УДК 004.056.5

СЕТЕВЫЕ ПРОТОКОЛЫ НОВОГО ПОКОЛЕНИЯ И ИХ АНАЛИЗ С ТОЧКИ ЗРЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

М. А. Селиванов
Донской государственный технический университет (г. Ростов-на-Дону, Российская Федерация)

Проведен разбор недостатков протокола HTTP/1.1. Представлен общий теоретический анализ безопасности протоколов нового поколения: SPDY, HTTP/2, QUIC, HTTP/3. Приведены актуальные статистические данные по внедрению данных протоколов в информационной среде. Проанализированы некоторые связанные с реализациями этих протоколов уязвимости. Дана оценка перспективам дальнейшего развития упомянутых протоколов в сетевой клиент-серверной среде на основе их характеристик и статистических данных.

Ключевые слова: протоколы TCP/IP, протокол HTTP, Интернет, сетевая производительность, уязвимости сетевых протоколов, уязвимости веб-серверов, целостность данных, доступность данных, информационная безопасность.

NEW GENERATION NETWORK PROTOCOLS AND THEIR ANALYSIS FROM THE POINT OF VIEW OF INFORMATION SECURITY

M. A. Selivanov
Don State Technical University (Rostov-on-Don, Russian Federation)

The paper provides the analysis of the shortcomings of the HTTP/1.1 protocol. It provides a general theoretical analysis of the security of the next generation protocols: SPDY, HTTP/2, QUIC, HTTP/3. Relevant statistics on the implementation of these protocols in the information environment are demonstrated. Some vulnerabilities associated with the implementations of these protocols are analyzed. The prospects for the further development of the mentioned protocols in a network client-server environment are estimated on the basis of their characteristics and the provided statistical data.

Keywords: TCP/IP protocols, HTTP protocol, Internet, network performance, vulnerabilities of network protocols, vulnerabilities of web servers, data integrity, data availability, information security.

Введение. В течение длительного времени основным протоколом обмена гипертекстовыми данными во Всемирной паутине (WWW) оставался базирующийся на TCP-соединении протокол прикладного уровня HTTP/1.1 (а также его расширение HTTPS, использующее HTTP через слой криптографических протоколов TLS/SSL для обеспечения конфиденциальности сетевых коммуникаций). Тем не менее, протоколы TCP и HTTP/1.1 содержат ряд особенностей, вызывающих задержки в доставке и получении данных, что в контрасте с ростом пропускной способности каналов обострило проблему оптимизации работы сетевых протоколов. Цель данной статьи — проанализировав новые протоколы для решения проблем задержек на прикладном и транспортном уровне, определить их недостатки и уязвимости.

Основная часть. Недостатки HTTP/1.1. Основной дискутируемый недостаток протокола HTTP/1.1 — отсутствие реализации мультиплексирования в запросах, т. е. асинхронного использования единого соединения для обмена информацией с обеспечением минимальных задержек. HTTP/1.1 с помощью заголовка Connection: keep-alive, который установлен по

умолчанию, может на определенное время устанавливать постоянное TCP-соединение с сервером, благодаря чему отпадает необходимость каждый раз нерационально устанавливать новые синхронизирующие TCP-рукопожатия. Также HTTP/1.1 может устанавливать конвейерное соединение (HTTP pipelining), позволяющее при постоянном TCP-соединении отправлять сразу несколько HTTP-запросов без ожидания ответа, что положительно влияет лишь на скорость процесса поочередной отправки пакетов. Однако конвейерные HTTP-соединения зачастую не имеют корректной поддержки во многих приложениях, и возможность их использования по умолчанию отключена (или не поддерживается) в большинстве браузеров. Данные методы оптимизации требуют потребления ресурсов на серверной стороне, что при высоконагруженном серверном трафике может приводить к DoS-угрозе (отказ в обслуживании). И самое главное — они не спасают от проблемы блокировки начала очереди (Head-of-line blocking, далее HOL) [1].

Суть проблемы HOL заключается в том, что ожидание доставки одного пакета блокирует отправку пакетов, следующих в очереди. Очевидно, особо остро эта проблема проявляется в том случае, когда ресурсы, передаваемые в начале очереди, имеют значительный размер.

Распространенным методом сокращения HOL-блокировок в браузерах является открытие нескольких параллельных TCP-соединений для каждого домена (в большинстве браузеров число этих соединений равно шести [2]). Это привело к использованию практики доменного шардинга (распределения ресурсов по разным доменам), спрайтинга (объединения изображений) и конкатенации (объединение) файлов [3]. Однако данные практики с учетом роста повсеместного распространения TLS/SSL-шифрования также приводят к значительному потреблению вычислительных ресурсов и временным задержкам.

Данные проблемы привели к разработке новых протоколов прикладного и транспортного уровня, призванных устранить недостатки в протоколах HTTP/TCP и обеспечить более рациональное, оптимальное и быстрое сетевое взаимодействие.

Далее будут рассмотрены и проанализированы некоторые из таких протоколов.

SPDY и HTTP/2. В 2009 году компания Google представила свой протокол прикладного уровня SPDY. Всего было выпущено четыре экспериментальные версии данного протокола [4]. Активная разработка и поддержка велась вплоть до 17 февраля 2015 года, когда IESG стандартизировала протокол HTTP/2. После стандартизации HTTP/2, спецификация которого была основана на SPDY, компания Google приняла решение упразднить разработку и поддержку SPDY в пользу HTTP/2 [5].

Данный протокол содержит следующие основные технологии:

— мультиплексирование, позволяющее клиенту отправлять несколько пакетов по разным асинхронным потокам при одном TCP-соединении (в управляющем фрейме пакета указывается его принадлежность к определенному потоку);

— используется одно соединение на домен;

— данные передаются в бинарном виде;

— реализуется контроль потока данных (управление размерами окна буфера) и обработки ошибок;

— происходит клиентская приоритизация запросов, с помощью которой клиент может выбирать приоритет доставки ресурсов;

— push-технология позволяет серверу отправлять несколько ответов на один запрос клиента (для предотвращения push-атак с неадекватным объемом информации предлагалась реализация контроля передаваемых push-данных на стороне браузера/клиента);

— происходит сжатие заголовков с помощью zlib (deflate);

— используется TLS-расширение NPN (Next Protocol Negotiation) для согласования используемого типа протокола прикладного уровня при TLS/SSL-подключении [6].

Летом 2012 года была опубликована информация об уязвимости CRIME-атаки (Compression Ratio Info-leak Made Easy), нацеленной на алгоритмы сжатия в SPDY и TLS/SSL, позволяющей злоумышленнику получить доступ к зашифрованным заголовкам (в частности, cookies-сессиям) [7].

После завершения поддержки SPDY компанией Google большинство веб-серверов прекратило поддержку модулей для данного протокола, заменив их на модули для HTTP/2 [8].

На сегодняшний день, согласно статистике W3Techs.com, протокол SPDY используют менее 5% веб-сайтов, и этот показатель продолжает снижаться (рис. 1) [9].

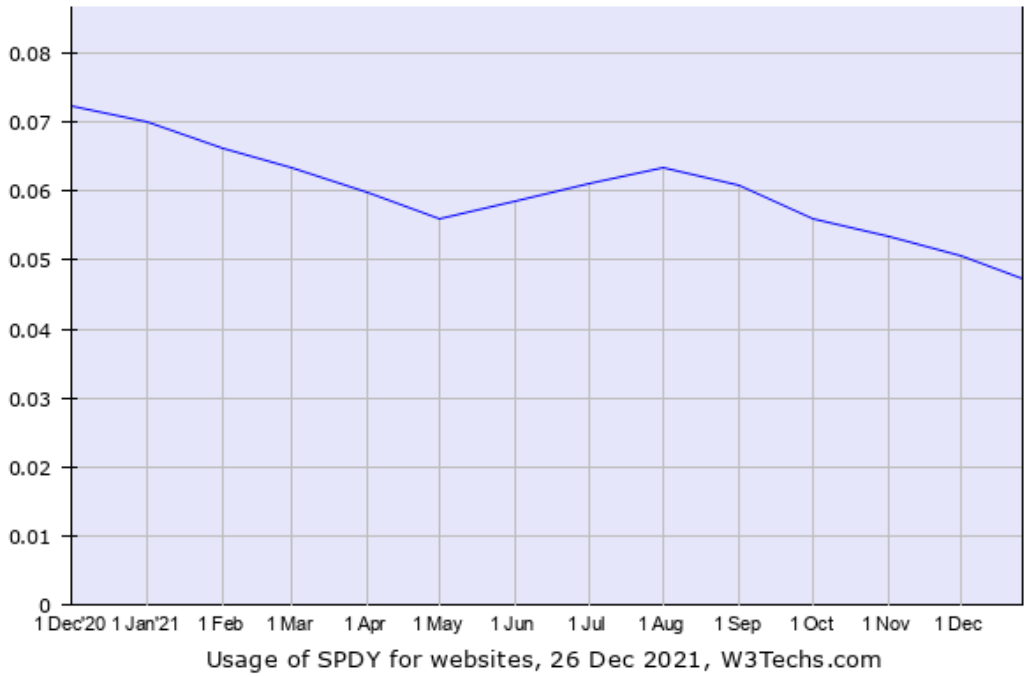

Рис. 1. Статистика использования SPDY веб-сайтами на конец 2021 года

Протокол HTTP/2, являющийся фактическим наследником протокола SPDY, был разработан HTTP Working Group (httpbis) и выпущен 14 мая 2015 года. Уже к концу 2015 года его поддержка была реализована в большинстве браузеров [5].

Основные особенности протокола HTTP/2:

— развитие на основе заложенных SPDY технологий;

— собственный алгоритм сжатия заголовков HPACK, использующий код Хаффмана и двустороннюю (клиент и сервер) таблицу заголовков, что должно предотвратить нерациональную отправку не изменяющихся заголовков;

— в качестве замены NPN используется TLS-расширение ALPN (Application Layer Protocol Negotiation);

— большинство браузеров де-факто поддерживают лишь HTTPS-подключение (TLSv1.2+) по HTTP/2, хотя HTTP/2 по стандарту допускает использование незашифрованного соединения [10].

Ряд реализаций протокола HTTP/2 в популярных серверах содержит множество уязвимостей, подвергающих непропатченное или некорректно настроенное серверное ПО различным угрозам, что ставит общую безопасность использования HTTP/2 под сомнение:

— множественные уязвимости типа DoS [11];

— Request Smuggling (внедрение запросов в backend-серверы через frontend-серверы) [12];

— HPACK-бомбы, приводящие к избыточному потреблению памяти в nghttp2 и Wireshark [13];

— HEIST-атака [14] и т. д.

На сегодняшний день, согласно статистике W3Techs.com, протокол HTTP/2 используется в ~45–50 % веб-сайтов (рис. 2) [15].

QUIC и HTTP/3. Протокол HTTP/2 обеспечивает предотвращение HOL-блокировок, однако это не касается транспортного уровня. В связи с тем, что протокол TCP должен гарантированно доставить пакеты по порядку, в случае потерь или задержек при передаче также создается блокировка. Таким образом, для предотвращения блокировок на транспортном уровне необходима разработка иных протоколов транспортного уровня.

В качестве решения компания Google выпустила в 2013 году протокол транспортного уровня QUIC (Quick UDP Internet Connections), позволяющий использовать мультиплексирование на основе протокола UDP.

Основные особенности QUIC:

— интегрированный TLS;

— принадлежность пакета потоку указывается на транспортном уровне в STREAM-фрейме;

— улучшенные механизмы контроля перегрузки;

— наличие алгоритма прямой коррекции ошибок (Forward Error Correction), заключающегося в том, что передача в пакете избыточных данных позволяет восстанавливать данные в случае ошибок и потерь (упразднено во второй версии);

— использование идентификатора соединения UUID, позволяющего не пересоздавать соединение при смене IP [16].

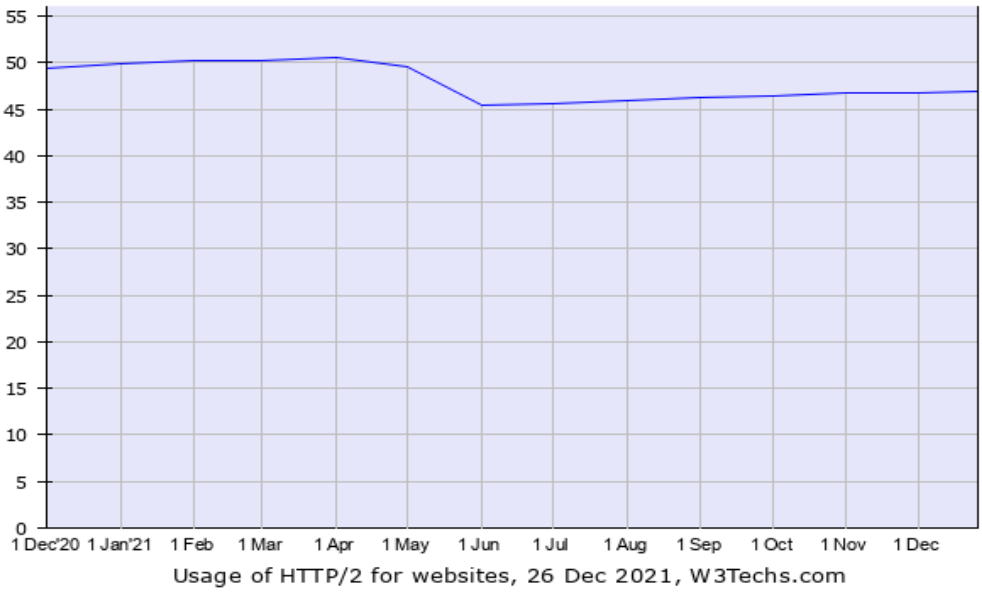

Рис. 2. Статистика использования HTTP/2 веб-сайтами на конец 2021 года

QUIC поныне является экспериментальным протоколом и содержит ряд очевидных недостатков [17]:

— использование UDP приводит к подверженности DoS-атакам;

— принудительность и встроенность шифрования заголовков и данных, что сильно усложняет процесс отладки;

— механизмы контроля перегрузки могут приводить к снижению скорости передачи.

На сегодняшний день, согласно статистике W3Techs.com, протокол QUIC используется в ~7–8% веб-сайтов и имеет тенденцию роста статистических показателей (рис. 3) [18].

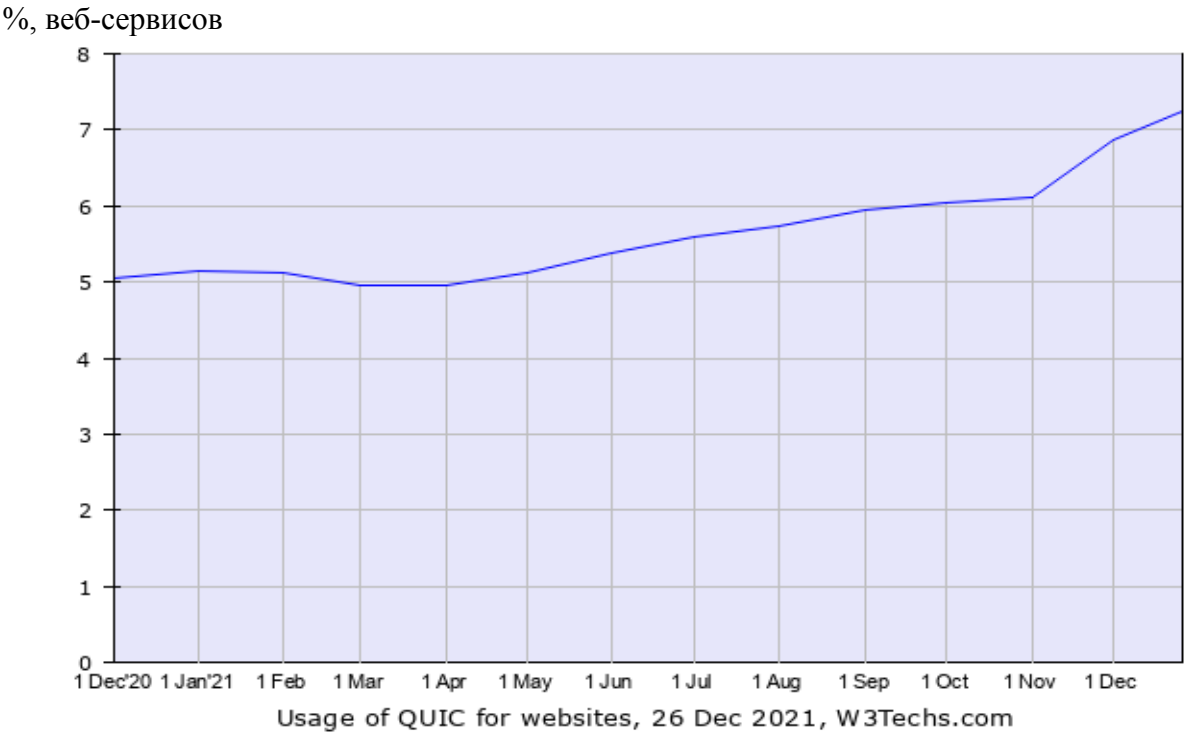

Рис. 3. Статистика использования QUIC веб-сайтами на конец 2021 года

В настоящее время ведется разработка протокола HTTP/3 (ранее известного как HTTP-over-QUIC), который должен обеспечить функционирование HTTP-технологий с использованием QUIC вместо TCP на транспортном уровне.

Большинство браузеров уже к 2021 году включило в свои спецификации поддержку HTTP/3, несмотря на не0стандартизированность протокола [19].

Уже доступны такие реализации HTTP/3, как quiche (API от Cloudflare, написанный на C и Rust), aioquic (API на Python), ведется разработка поддержки для веб-сервера nginx и др.

На сегодняшний день, согласно статистике W3Techs.com, протокол HTTP/3 используется в ~24% веб-сайтов и имеет тенденцию роста статистических показателей (рис. 4) [20].

%, веб-серверов

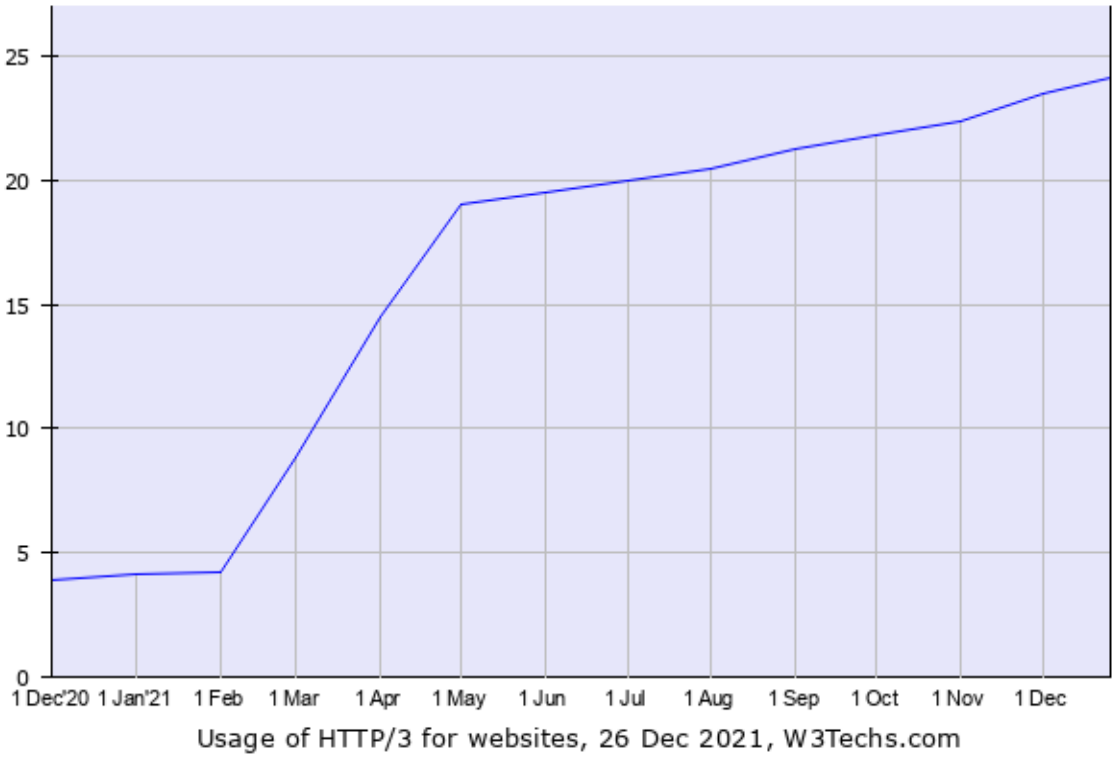

Рис. 4. Статистика использования HTTP/3 веб-сайтами на конец 2021 года

Заключение. Таким образом, можно сделать вывод, что на сегодняшний день активно ведутся разработка и внедрение протоколов нового поколения, призванных решить проблемы задержек на прикладном и транспортном уровне. Однако концепции новых протоколов неидеальны и имеют ряд просчетов и уязвимостей. Прежде чем использовать новые протоколы (особенно нестандартизированные), рекомендуется ознакомиться с принципами и особенностями работы этих протоколов, а также существующими в них уязвимостями.

Библиографический список

1. Connection_management_in_HTTP_1.x // MDN Web Docs: [web page]. URL:https://developer.mozilla.org/ru/docs/Web/HTTP/Connection_management_in_HTTP_1.x (accessed: 26.12.2021).

2. Browser connection limitations // Push Technology: [web page]. — URL: https://docs.pushtechnology.com/cloud/latest/manual/html/designguide/solution/support/connection_limitations.html (accessed: 26.12.2021).

3. Optimizing Application Delivery // High Performance Browser Networking: [web page]. — URL: https://hpbn.co/optimizing-application-delivery/#optimizing-for-http1x (accessed: 26.12.2021).

4. SPDY Protocol // The Chromium Projects: [web page]. — URL: https://dev.chromium.org/spdy/spdy-protocol (accessed: 26.12.2021).

5. HTTP/2 // Wikipedia (eng.): [web page]. — URL: https://en.wikipedia.org/wiki/HTTP/2 (accessed: 26.12.2021).

6. SPDY Protocol — Draft 3.1 // The Chromium Projects: [web page]. — URL: https://www.chromium.org/spdy/spdy-protocol/spdy-protocol-draft3-1 (accessed: 26.12.2021).

7. CVE-2012-4930 // CVE: [web page]. — URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-4929 (accessed: 26.12.2021).
8. Module ngx_http_spdy_module // nginx documentation: [web page]. — URL: http://nginx.org/en/docs/http/ngx_http_spdy_module.html (accessed: 26.12.2021).
9. Usage statistics of SPDY for websites // W3Techs: [web page]. — URL: https://w3techs.com/technologies/details/ce-spdy (accessed: 26.12.2021).
10. HTTP/2 // High Performance Browser Networking: [web page]. — URL: https://hpbn.co/http2/ (accessed: 26.12.2021).
11. HTTP/2 can shut you down! // SecPod: [web page]. — URL: https://www.secpod.com/blog/http2-dos-vulnerabilities/ (accessed: 26.12.2021).
12. HTTP/2: The Sequel is Always Worse // PortSwigger: [web page]. — URL: https://portswigger.net/research/http2 (accessed: 26.12.2021).
13. HTTP/2: In-depth analysis of the top four flaws of the next generation web protocol // Imperva [web page]. — URL: https://www.imperva.com/docs/Imperva_HII_HTTP2.pdf (accessed: 26.12.2021).
14. CVE-2016-7153 // CVE: [web page]. — URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-7153 (accessed: 26.12.2021).
15. Usage statistics of HTTP/2 for websites // W3Techs: [web page]. — URL: https://w3techs.com/technologies/details/ce-http2 (accessed: 26.12.2021).
16. QUIC, a multiplexed transport over UDP // The Chromium Projects: [web page]. — URL: https://www.chromium.org/quic (accessed: 26.12.2021).
17. QUIC: What is behind the experimental Google Protocol? // Digital Guide IONOS: [web page]. — URL: https://www.ionos.com/digitalguide/hosting/technical-matters/quic-the-internet-transport-protocol-based-on-udp/ (accessed: 26.12.2021).
18. Usage statistics of QUIC for websites // W3Techs: [web page]. — URL: https://w3techs.com/technologies/details/ce-quic (accessed: 26.12.2021).
19. HTTP/3 // Wikipedia (eng.): [web page]. URL:https://en.wikipedia.org/wiki/HTTP/3 (accessed: 26.12.2021).
20. Usage statistics of HTTP/3 for websites // W3Techs: [web page]. — URL: https://w3techs.com/technologies/details/ce-http3 (accessed: 26.12.2021).

Об авторе:

Селиванов Максим Александрович, студент кафедры «Вычислительные системы и информационная безопасность» Донского государственного технического университета (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), sog7698@icloud.com

About the Author:

Selivanov, Maksim A., Student, Department of Computing Systems and Information Security, Don State Technical, sog7698@icloud.com