МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.057.4

ОТКАЗУЕМЫЕ ГРУППОВЫЕ КОММУНИКАЦИИ В МОДЕЛИ ГЛОБАЛЬНОГО НЕОГРАНИЧЕННОГО ЗЛОУМЫШЛЕННИКА[1]

В. Ф. Шейдаев*, Д. Ю. Гамаюнов**

*Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия
**Национальный исследовательский университет «Высшая школа экономики», г. Москва, Россия

Предлагается использовать децентрализованные отказуемые групповые коммуникации для обеспечения защищённости общения в модели нарушителя, имеющего доступ к инфраструктуре передачи данных, а также неограниченное финансирование. За основу взят существующий протокол групповых отказуемых коммуникаций multi-party Off-the-Record (mpOTR), допускающий работу на децентрализованном транспорте, его свойства совершенной прямой секретности улучшены за счёт введения процедуры продвижения ключевого материала (Key Ratcheting). Предложена полностью децентрализованная система защищённых групповых коммуникаций, обладающая свойствами отказуемости, согласованности текста переписки и улучшеными свойствами совершенной прямой секретности, а также способная противостоять Sybil-атакам. Соответствующее программное средство реализовано на языке JavaScript и обеспечивает защищённую передачу сообщений между браузерами в условиях отсутствия центрального сервера.

Ключевые слова: *безопасность коммуникаций, децентрализованные коммуникации, отказуемость.*

DOI 10.17223/20710410/40/6

DENIABLE GROUP COMMUNICATIONS IN THE PRESENCE OF GLOBAL UNLIMITED ADVERSARY

V. F. Sheidaev*, D. Y. Gamayunov**

*Lomonosov Moscow State University, Moscow, Russia
**National Research University Higher School of Economics, Moscow, Russia

E-mail: enr1g@seclab.cs.msu.su, dgamaunov@hse.ru

With this paper, we provide our research into the problem of secure communications in the presence of a global unlimited adversary. As a solution, we propose to use decentralised deniable communications. We have made use of our existing multi-party Off-the-Record protocol's modification which is able to function over decentralised

[1]Работа поддержана грантом УМНИК Фонда содействия развитию малых форм предприятий в научно-технической сфере.

transport. Its Perfect Forward Secrecy (PFS) features were improved by adding Key Ratcheting procedure to the protocol's flow. As a result, we propose a fully decentralised cryptosystem which has deniability and transcript consistency features, improved PFS and ability to resist the Sybil attack. We also give a detailed overview of the protocol model implementation in JavaScript designed to function in conditions of centralized server's absence.

Keywords: *secure communications, decentralised communications, deniability.*

Введение

В 2004 г. была опубликована работа Яна Голдберга и Никиты Борисова о протоколе отказуемых коммуникаций OTR — Off-The-Record (Communications) [1]. Цель протокола — устранить недостатки традиционных средств криптографически защищённого общения, которые, по мнению авторов [1], уже не способны обеспечить должного уровня защищённости. В частности, традиционные криптографические протоколы не обладают свойствами отказуемости и совершенной прямой секретности, например, протокол PGP (Pretty Good Privacy) и другие построенные на аналогичных концепциях протоколы используют сильную криптографическую подпись. Это позволяет подтвердить авторство сообщения не только получателю, но и произвольному третьему лицу. Таким образом, участие отправителя в переписке и авторство конкретного сообщения может быть доказано. PGP не способен обеспечить участникам коммуникации возможность подтверждения авторства ограниченным кругом лиц и в течение ограниченного времени (непосредственно во время коммуникации). Если Alice является информатором и не доверяет Bob'у, PGP не позволит ей надёжно аутентифицировать себя и при этом иметь возможность отрицать участие в беседе. Свойство конфиденциальности PGP опирается также на сохранение закрытого ключа в секрете. В случае компрометации долговременного ключевого материала будет скомпрометирована и вся зашифрованная при его помощи корреспонденция. В эпоху распространения мобильной техники потеря или кража ключевого материала особенно актуальны.

В последние годы особое внимание уделяется проблеме защиты частных коммуникаций в присутствии глобального злоумышленника (Global Passive Adversary), то есть в условиях асимметрии возможностей пользователя по обеспечению конфиденциальности своих данных и возможностей владельцев телекоммуникационной инфраструктуры по сбору и анализу данных всех пользователей [2 – 4]. В значительной степени увеличилось количество средств защищённого общения (мессенджеров), в том числе с поддержкой свойств совершенной прямой секретности и отказуемости. Начали проводиться сравнительные исследования в предметной области, в том числе академические [5, 6]. Анализ приведённых работ показывает, что на сегодняшний день среди защищённых мессенджеров отсутствуют практически применимые протоколы групповых отказуемых коммуникаций. Более того, большинство представленных решений являются централизованными. С точки зрения защиты коммуникаций в присутствии глобального злоумышленника любой централизованный сервис обмена сообщениями является возможной точкой компрометации, а в некоторых случаях, когда поставщиком сервиса является крупная глобальная компания (такая, как Facebook, Apple или Microsoft), сам поставщик сервиса является глобальным злоумышленником в рассматриваемой модели нарушителя. Отметим, что использование одного лишь end-to-end шифрования недостаточно. Центральные сервера занимаются пересылкой сообщений между абонентами, хранят метаданные либо владеют другой привлекательной

для злоумышленников информацией, например списками контактов пользователей и их телефонными номерами. Наконец, провайдеры централизованных коммуникаций зачастую используют обезличенные данные пользователей для маркетинговых целей. Всё это делает централизованную инфраструктуру средств общения потенциальной целью для злоумышленников разного масштаба.

Отметим, что разработка протокола децентрализованного защищённого группового общения является нетривиальной задачей, особенно в контексте отказуемых коммуникаций. Во-первых, задача обеспечения отказуемости для группы участников на сегодняшний день не имеет удовлетворительного по эффективности решения [7]. Во-вторых, децентрализация коммуникаций порождает необходимость в обеспечении согласованности текста переписки. Наконец, открытым остаётся вопрос распределения ключевого материала среди участников. PKI не подходит ввиду своей централизованной природы. В свою очередь, Web of Trust предполагает, что доверенные отпечатки ключей будут подписываться долговременными ключами, что не согласуется с концепциями отказуемых коммуникаций. Ранее авторами работы [8] был предложен протокол, являющийся развитием концепции протокола mpOTR (multi-party OTR) [9], и разработана его централизованная модельная реализация [10]. Отметим, что протокол поддерживал и децентрализованный режим функционирования, поэтому дальнейшие исследования были направлены на создание практически применимого протокола распределённого отказуемого общения, а также на улучшение свойств протокола.

В данной работе представлены результаты в области исследования и разработки защищённых групповых коммуникаций, а также децентрализованная модельная реализация, демонстрирующая работоспособность концепции [11].

1. Состояние предметной области

Область отказуемых средств общения является сравнительно молодой и не представлена большим количеством протоколов и прикладных реализаций. Наиболее удачными, с нашей точки зрения, являются Signal, GOTR и RetroShare, однако все они имеют те или иные недостатки.

Signal — протокол защищённого общения, создание которого было вдохновлено оригинальным OTR [12]. Он имеет множество улучшений в сравнении с OTR, связанных с ориентацией Signal на асинхронную среду передачи сообщений: Signal имеет более эффективную процедуру аутентификации [13] и улучшенный механизм продвижения ключевого материала (Key Ratcheting) [14]. Разработчиками Signal также были предприняты попытки создания группового протокола отказуемых коммуникаций [15]. Идея группового протокола Signal — установление попарных соединений между участниками, что порождает накладные расходы при передаче и хранении контекстов коммуникации со всеми участниками. Протокол Signal используется в одноимённом мессенджере, а также в WhatsApp, Facebook Messenger и Allo [16 – 18]. Из недостатков протокола отметим его централизованность, а также то, что учётные записи Signal привязаны к номерам мобильных телефонов, т. е. если Alice и Bob хотят поговорить друг с другом, то они должны раскрыть друг другу свои номера. Наконец, Signal не поддерживает согласованности передачи сообщений при групповых коммуникациях.

GOTR является одной из академических попыток разработки Multi-party Off-the-Record протокола [19]. При создании GOTR авторы ориентировались на устранение недостатка оригинального mpOTR: для изменения состава участников в mpOTR необходимо завершить текущий сеанс коммуникации и запустить новый [9]. В случае с GOTR применяются так называемые кольцевые ключи. Используемый протокол

сложен в описании и реализации и требует повышенного количества ресурсов в связи с тем, что каждый участник коммуникации создаёт по два виртуальных узла на каждое попарное соединение. Авторами [19] заявлено наличие модельной реализации протокола в виде плагина к известному мессенджеру, однако она не обнаружена в публичном доступе. Протокол поддерживает децентрализованную передачу данных, но вопросы обеспечения согласованности не поднимаются.

Наконец, RetroShare является полностью децентрализованным решением [20]. Он построен на принципах F2F (Friend-to-Friend — пиринговая сеть, в которой пользователи соединены напрямую только с доверенными узлами) и поддерживает работу через DHT (Distributed Hash Table — распределённая хеш-таблица, используется при построении распределённых сетей без трекеров). RetroShare применяет хорошо известные криптографические примитивы. Аутентификация в нём производится посредством PGP, а защищённая передача данных — при помощи последних версий TLS. Из последнего утверждения следует, что RetroShare поддерживает совершенную прямую секретность, но в то же время не поддерживает свойств отказуемости. Разработчиками не заявлена поддержка какой-либо модели консистентности передачи сообщений.

Подводя итог, стоит отметить, что среди рассмотренных протоколов нет практически применимого протокола децентрализованного отказуемого общения со свойствами согласованности текста переписки. Это подтверждает актуальность данной работы, где приведены основные результаты проекта, целью которого является создание протокола, удовлетворяющего этим требованиям.

2. Описание протокола

2.1. Модель нарушителя

Рассматривается модель глобального злоумышленника, основными целями которого являются нарушение конфиденциальности передаваемых данных и идентификация участников коммуникации. К нарушителям такого класса можно отнести, например, провайдеров облачных услуг и средств централизованных коммуникаций, а также организации, занимающиеся реализацией персональных данных пользователей.

Особенность подобной модели нарушителя в том, что злоумышленник может попытаться скомпрометировать ключевой материал участников коммуникации путём перебора, так как обладает значительными техническими и финансовыми возможностями, а также может записывать трафик коммуникаций на протяжении длительного времени. Например, в [21] продемонстрирована атака на протокол Diffie — Hellman Key Exchange (DHKE), позволяющая существенно сократить время нахождения дискретного логарифма. Стоит заметить, что многие реализации DHKE из соображений производительности используют фиксированные значения порядка группы и генератора, что позволяет злоумышленнику предвычислить этапы алгоритма дискретного логарифмирования, зависящие только от порядка группы. По мнению авторов [21], уже сегодня организация с неограниченным финансированием способна производить успешные атаки на DHKE с длиной ключа 1024 бит. Если в будущем по каким-то причинам (компрометация, злой умысел участников коммуникации) злоумышленник получит доступ к открытым текстам переписки, он также сможет попытаться привязать сеанс коммуникации к участникам.

Помимо того, что подобный злоумышленник может записывать и хранить транскрипт переписки (пусть даже шифрованный), он же может выполнять задачу взаимной идентификации пользователей. Например, пользователи будут идентифицировать друг друга при помощи номеров мобильных телефонов, которые существенно привя-

заны к своим владельцам, в частности, в большинстве стран номера телефонов регистрируются по удостоверяющим личность документам. Таким образом, злоумышленник в нашей модели может неотказуемо привязать ключевой материал пользователей к ним самим, например через номер телефона.

Подытоживая, сформулируем модель нарушителя следующим образом: в качестве нарушителя будем рассматривать организацию, обладающую существенными вычислительными мощностями и/или доступом к инфраструктуре передачи данных.

2.2. Краткое описание базового протокола

В [8] сформулированы требования к протоколу криптографически защищённых групповых коммуникаций, предложен удовлетворяющий этим требованиям протокол, а также проведена его формальная верификация (model checking). Приведём эти требования:

1) конфиденциальность;
2) целостность;
3) аутентичность;
4) отказуемость;
5) совершенная прямая секретность;
6) согласованность транскрипта.

Первые три свойства являются традиционными в криптографии, гораздо больший интерес представляют остальные — именно они обеспечивают достаточный уровень безопасности в контексте рассматриваемой модели нарушителя. Протокол [8] основан на multi-party Off-the-Record из работы [9]. Актуальность его модификации была вызвана тем, что оригинальный mpOTR имел лишь высокоуровневое описание и многие существенные вопросы реализации остались открытыми. В частности, в [9] не было предложено масштабируемой процедуры групповой аутентификации и не были решены проблемы, связанные с обеспечением согласованности транскрипта во время коммуникации. Предложенный в [8] протокол решает обе эти проблемы. Во-первых, в нём использована улучшенная процедура отказуемой групповой аутентификации IDSKE, имеющая сложность $4N$ по числу сообщений, в сравнении с $12N$ у mpOTR [22]. Во-вторых, для обеспечения согласованности транскрипта используется протокол OldBlue, позволяющий добросовестным участникам поддерживать причинную консистентность передаваемых сообщений [23]. Не будем подробно останавливаться на описании всего протокола, отметим лишь, что протокол описывает полный цикл фаз защищённого общения:

1) установление канала: участники устанавливают соединение между собой, вообще говоря, не защищённое;
2) аутентификация: участники совершают процедуру отказуемой групповой аутентификации. В результате участники вырабатывают сессионный ключ шифрования и отказуемые подписные ключи;
3) коммуникация: на данном этапе участники непосредственно пересылают друг другу сообщения;
4) завершение: безопасное удаление из памяти сессионного ключа и публикация отказуемых подписных ключей.

2.3. Продвижение ключевого материала

Ещё одним недостатком протокола mpOTR является то, что он не обладает свойством совершенной прямой секретности во время коммуникации. Подобное свойство полезно для протокола в случае, если по каким-то причинам сессионный ключ шиф-

рования был скомпрометирован во время коммуникации, т. е. пока он ещё существовал в памяти устройств участников сеанса общения. Факт использования одного ключа на весь сеанс коммуникации отличает mpOTR от OTR [1] или его аналогов [14], в которых новый эфемерный ключ шифрования вырабатывается каждые несколько сообщений, что существенно уменьшает последствия возможной компрометации.

Данный процесс в литературе часто называют Key Ratcheting [5, 14]. Мы будем называть его *продвижением ключевого материала*. Будем обозначать i-й сессионный ключ sk_i. Множество сообщений, которые компрометируются при компрометации ключа sk_i, будем называть *окном компрометации*. Отметим, что в окно компрометации могут входить не только сообщения, которые непосредственно были зашифрованы ключом sk_i, но и сообщения, зашифрованные предыдущими и/или последующими ключами. Это связано с тем, что компрометация sk_i может повлечь за собой компрометацию других ключей в зависимости от выбранной стратегии генерации ключей.

Во введённых терминах цель процедуры продвижения ключевого материала можно сформулировать как минимизацию окна компрометации. В случае с общением двух собеседников задача имеет эффективное решение. В частности, в [5, 14] описываются различные подходы к организации продвижения ключевого материала. Их сравнение можно найти на рис. 1: Naïve KDF Ratchet соответствует алгоритму Silent Circle, Double Ratchet — протоколу Axolotl из [14], а OTR Ratchet — продвижению ключевого материала в оригинальном OTR [1]. Как можно видеть, наиболее удачным с точки зрения минимизации окна компрометации является смешанный подход Double Ratchet, так как при этом так же, как и в случае Naïve KDF Ratchet, каждое сообщение шифруется уникальным ключом и при первой же возможности вырабатывается новый сессионный ключ, независящий от предыдущих.

При описании алгоритма продвижения ключевого материала будем использовать следующие понятия:

— sk_i — мастер-ключ, вырабатываемый в результате групповой процедуры генерации ключа. В качестве неё выбран прокол Бурместера — Десмедта [24], так как в сравнении с другими известными протоколами для симметричной групповой выработки общего секрета, например групповым протоколом Диффи — Хеллмана, он имеет наименьшее число раундов;
— sk_i^n — ключ n-го сообщения, отправленного после порождения мастер-ключа sk_i.

В предложенной реализации для порождения ключа sk_i^n использована схема $Hash(sk_i + Hash(n))$. Она отличается от, например, применяемой в Axolotl и Silent Circle, где новый ключ зависит только от предыдущего, то есть $sk_i^n = Hash(sk_i^{n-1})$, $sk_i^0 = sk_i$. В выбранной схеме при компрометации ключа sk_i^n злоумышленнику становится доступно только одно сообщение, в то время как во второй — все последующие до очередной процедуры групповой выработки ключа.

Процедура продвижения работает следующим образом: в начале переписки сразу после процедуры аутентификации счётчики текущего сообщения n и текущего мастер-ключа i инициализируются нулём, а начальный мастер-ключ sk_0 — общим секретом, выработанным в ходе работы IDSKE [22].

При отправке сообщение шифруется ключом sk_i^n. В метаданные сообщения добавляются значения nr и ir, соответствующие текущим значениям n и i, после чего счётчик n увеличивается на единицу. При получении сообщения из его метаданных извлекаются значения счётчиков nr и ir, формируется ключ sk_{ir}^{nr} и сообщение расшифровывается.

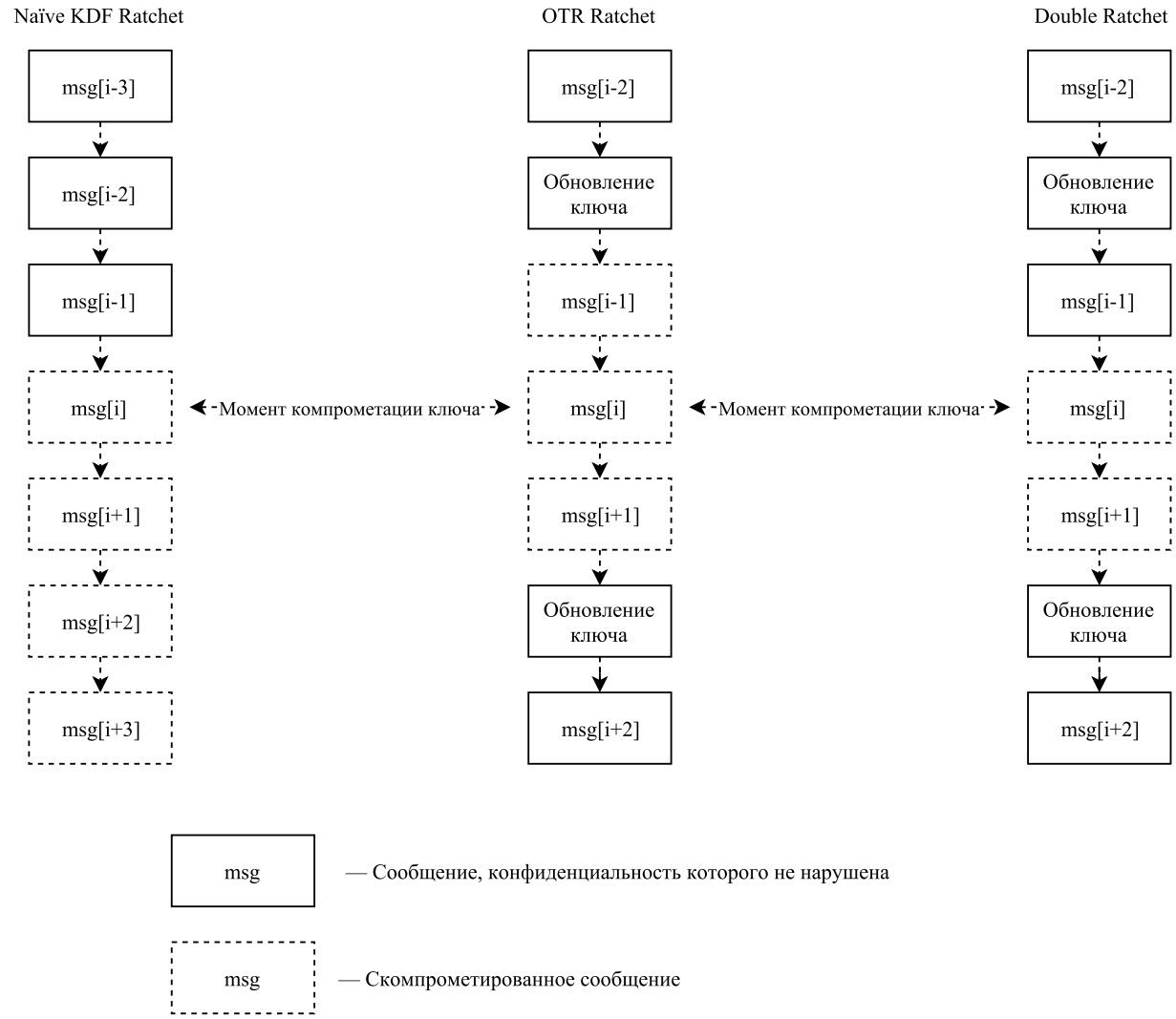

Рис. 1. Сравнение различных подходов к продвижению ключевого материала

Каждые N сообщений или T секунд производится процедура обновления текущего сессионного ключа при помощи алгоритма Бурместера — Десмедта и счётчик сообщений n сбрасывается в ноль. Предыдущие мастер-ключи sk_i хранятся в памяти до тех пор, пока не будет доставлено последнее зашифрованное при их помощи сообщение. В протоколе поддерживается свойство согласованности текста переписки, что позволяет знать, когда пришло последнее сообщение, зашифрованное старым ключом.

2.4. Проблемы аутентификации в условиях децентрализованной среды

При построении полностью децентрализованной инфраструктуры передачи данных для защищённого мессенджера возникает множество проблем. Во-первых, важно рассмотреть вопрос устойчивости предлагаемой схемы к sybil-атакам [25]. Для решения данной задачи ранее предлагались решения, основанные на использовании удостоверяющих центров либо установлении лимитов на число присоединяемых узлов, чтобы ограничить возможности атакующего по внедрению своих узлов. Оба подхода неприменимы в рассматриваемой модели нарушителя, так как требуют высокой степени централизации. Более поздние работы, нацеленные на построение P2P-сетей на DHT, используют в качестве вспомогательной меры знания о социальных связях участников

сети [26, 27]. Однако и эти подходы имеют недостатки, так как предполагают наличие свойств сильной связности у графа социальных связей, а также известную оценку размеров децентрализованной сети. При реализации мы предлагаем использовать модификацию классического DHT протокола Chord [28]. Данный протокол примечателен тем, что имеет минимальные предположения о сети, прост в реализации, а также достаточно эффективен при большом количестве зловредных узлов.

Во-вторых, встаёт вопрос распределения ключей между участниками коммуникации. Рассмотрены три подхода: Public Key Infrastructure (PKI), Web Of Trust (WoT) и Trust On First Use (TOFU). Ввиду стремления максимально отказаться от узких мест доверия (так называемых trust bottleneck), решено не рассматривать PKI из-за его централизованной природы. Что касается WoT, то в нём предполагается, что пользователи будут использовать неотказуемые подписи публичных ключей своих собеседников. Это не вступает в прямое противоречие со свойствами безопасности нашего протокола, однако наличия подобных подписей всё же хотелось бы избежать, когда речь идёт о протоколе отказуемого общения (не столько с точки зрения математической, сколько с точки зрения практического использования протокола). TOFU видится наиболее перспективным способом обмена ключами. Суть протокола заключается в том, что стороны обмениваются отпечатками ключей по третьему каналу, а позже при первом установлении соединения сравнивают предъявленные отпечатки с полученными ранее и запоминают свой выбор на соединяющихся устройствах. Таким образом, участники хранят у себя на устройствах список доверенных публичных ключей. Данный подход хорош минимальностью инфраструктуры и отсутствием какой-либо централизации. Наибольшее распространение TOFU получил в протоколе SSH. При использовании TOFU немаловажным аспектом является используемый тип отпечатка ключа. Существуют исследования, посвящённые разработке удобных в использовании (чтении и сравнении «на глаз») отпечатков ключей [29 – 33].

Мы решили остановиться на подходе [33], так как он не снижает уровня безопасности в сравнении с классическими отпечатками ключей, но при этом является более читаемым, потому что содержит базовые слова английского языка. Оригинальный OTR предоставляет также возможность проверить личность собеседника при помощи секретного вопроса. Для этого используются схемы доказательства с нулевым разглашением, позволяющие проверить знание стороной общего секрета без раскрытия информации о нём. В OTR с этой целью применяется протокол SMP (Socialist Millionaires Protocol) [34]. Мы решили не отходить от этой практики и также предоставлять пользователю возможность подтверждения личности собеседника при помощи SMP без каких-либо изменений в протоколе.

3. Описание реализации

Средство коммуникации, в котором реализован предложенный протокол, представляет собой децентрализованное веб-приложение на языке JavaScript. Узлы устанавливают соединение друг с другом при помощи сервера-трекера. Он используется только для трансляции идентификаторов пользователей в IP-адреса, сообщения пользователей передаются напрямую. Особый случай представляют пользователи, использующие протокол NAT (Network Address Translation — «преобразование сетевых адресов» — механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов). Для них установление прямого соединения не всегда возможно, однако для подобных ситуаций WebRTC поддерживает протокол TURN, позволяющий воспользоваться сервером для ретрансляции.

Узлы соединяются друг с другом в топологию «каждый с каждым». При необходимости выполнить какой-нибудь централизованный алгоритм, например перезапустить коммуникацию в случае отсоединения одного из участников, лидер (т. е. участник, координирующий выполнение централизованного алгоритма) выбирается следующим образом: каждый узел сортирует уникальные идентификаторы всех участников коммуникации и выбирает наибольший из них. Узел с максимальным идентификатором объявляется лидером, при этом для его определения не потребовалась отправка ни одного сообщения.

Для работы с WebRTC используется библиотека с открытым исходным кодом peerjs [35]. Вместе с ней поставляется публичный сервер-трекер, который при необходимости можно развернуть самостоятельно, в том числе в локальной сети.

3.1. Характеристики модельной реализации

На данный момент разработана модельная реализация протокола на языке JavaScript с использованием пиринговой технологии WebRTC [11]. Реализация является децентрализованной, однако для простоты транспортная часть реализована с использованием трекера [35], а не DHT. Алгоритм распределённого транспорта [28] планируется реализовать в будущем.

Проведено исследование свойств модельной реализации в различных сценариях использования. Во время исследования производилось профилирование на предмет того, сколько времени затрачено на криптографические вычисления, а сколько на сетевые операции.

Установление соединения: n пользователей устанавливают первичное соединение друг с другом и проводят процедуру взаимной аутентификации. При этом между участниками используется тип соединения каждый-с-каждым, то есть не предусмотрена ретрансляция. Результаты приведены в табл. 1.

Таблица 1

Время установления соединения, мс

n	3	5	7	10
Установление первичного соединения	153	609	700	1005
Фаза аутентификации	1712	1987	2357	3823
Фаза аутентификации: сетевые операции	1011	1254	1437	2761
Фаза аутентификации: криптографические вычисления	701	733	920	1062

Время приёма-передачи: измеряется время, за которое участник U_i рассылает всей группе сообщение, а другой участник U_j отвечает ему (табл. 2). Время окончания фиксируется после доставки ответного сообщения всем участникам. При этом предполагается отсутствие потерь сообщений на каналах связи. Время между узлами синхронизируется при помощи локального NTP-сервера (Network Time Protocol — сетевой протокол для синхронизации внутренних часов компьютера с использованием сетей с переменной задержкой).

Таблица 2

Время приёма-передачи (RTT), мс

n	3	5	7	10
RTT	76	98	112	120
RTT: сетевые операции	74	95	109	117
RTT: криптографические вычисления	2	3	3	3

Время повторного соединения после разрыва/отключения: в первоначальной конфигурации имеется n участников, далее участник U_n штатно отсоединяется от остальных либо имитируется сетевой разрыв с другими участниками. Измеряется время, через которое коммуникация возвращается в консистентное состояние (табл. 3).

Таблица 3

Восстановление коммуникации после разрыва, мс

n	3	5	7	10
Обнаружение разрыва	4119	5964	4712	5001
Обнаружение отсоединения	5	12	17	19
Перезапуск чата	1472	1719	2374	3773
Сумма	1477	1731	2391	3792
Сумма (разрыв)	5591	7683	7086	8774

3.2. Особенности платформы

Браузеры зачастую воспринимаются как среда, непригодная для построения криптографически защищённых приложений. Подобное мнение сформировано, в частности, из-за того, что JavaScript не имел доступа к источникам достаточной энтропии, необходимых типов данных для работы с большими числами, а также не мог обеспечить константного выполнения некоторых операций для защиты от атак по сторонним каналам; наконец, веб-приложения нельзя считать безопасным способом доставки контента ввиду их динамичности [36, 37]. Поэтому эта тема длительное время обходилась академическими исследованиями стороной.

Справедливо будет отметить, что в последние несколько лет ситуация существенно изменилась в лучшую сторону, как и мнение исследователей по данному вопросу [38, 39]. Во-первых, JavaScript получил поддержку криптографически стойкой генерации псевдослучайных чисел, а также API для работы с криптографическими примитивами [40]. Во-вторых, увеличилось количество библиотек, реализующих различные криптографические примитивы, в том числе академических и активно сопровождаемых [41]. Наконец, на вторую половину 2017 г. более 70 % браузеров в мире поддерживают технологию WebRTC [42]. Это сложный сетевой протокол, позволяющий браузерам устанавливать друг с другом end-to-end соединение, свойства безопасности которого высоко оценены [43].

Что касается атак по сторонним каналам, то данный вопрос исследован в работах [44, 45]. Показано, что контроль злоумышленником одной из вкладок браузера может позволить определить, какой сайт из списка известных был загружен в другой вкладке, какие клавиши были нажаты, какой документ отрисовывался в соседнем окне, наконец, даже определить, что в подключенном шифрованном диске есть скрытый раздел. Атаки проводились при помощи профилирования состояния L3-кэша процессора, что стало возможным благодаря поддержке High Resolution Timer в JavaScript. Это позволяет атаковать не только другие вкладки и окна браузера, но также и приложения, запущенные, например, в виртуальной машине. То есть данная проблема охватывает вычислительные устройства в целом, а не только браузеры. Её решение возможно при помощи модификации аппаратного обеспечения либо, что более вероятно, введения ограничения на использование High Resolution Timer в JavaScript-приложениях.

Специфических атак по сторонним каналам, позволяющих злоумышленнику получить данные из другой вкладки браузера, нами не обнаружено. В целом, это позволяет

сделать вывод о жизнеспособности криптографических приложений на базе стека Web-технологий.

Заключение

Представлено описание протокола защищённых групповых коммуникаций со свойством отказуемости, а также его улучшения, связанные с обновлением ключевого материала. Протокол реализован на языке JavaScript с использованием пиринговой технологии WebRTC, результаты исследования базовых сценариев представлены в работе.

Мы связываем дальнейшее развитие свойств протокола с улучшением свойств удобства пользования, решением вопроса взаимной аутентификации пользователей в контексте отказуемых коммуникаций и переходом на использование децентрализованного транспорта без использования трекера, например на основе DHT.

ЛИТЕРАТУРА

1. *Borisov N., Goldberg I., and Brewer E.* Off-the-record communication, or, why not to use PGP // Proc. Workshop on Privacy in the Electronic Society. ACM, 2004. P. 77–84.
2. `http://www.bbc.com/news/world-us-canada-22837100` — Profile: Edward Snowden, BBC News.
3. `http://www.globalissues.org/article/802/surveillance-state` — Surveillance State: NSA Spying and more.
4. `http://www.theguardian.com/media/2015/jul/02/wikileaks-us-spied-on-angela-merkels-ministers-too-says-german-newspaper` — WikiLeaks: US spied on Angela Merkel's ministers too, says German newspaper, The Guardian.
5. *Unger N. et al.* SoK: secure messaging // Proc. IEEE Symp. Security and Privacy (SP). 2015. P. 232–249.
6. `https://www.eff.org/secure-messaging-scorecard` — Electronic Frontier Foundation. Secure Messaging Scorecard.
7. *Unger N. and Goldberg I.* Deniable key exchanges for secure messaging // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 1211–1223.
8. *Коростелева М. В., Гамаюнов Д. Ю.* Обеспечение криптографически защищенных групповых коммуникаций с функцией отказуемости // Проблемы информационной безопасности. Компьютерные системы. 2014. №. 3. С. 74–79.
9. *Goldberg I. et al.* Multi-party off-the-record messaging // Proc. 16th Conf. Computer and Communications Security. ACM, 2009. P. 358–368.
10. `https://github.com/maria-msu-seclab/mpotrDevelopment` — Moscow State University Seclab mpOTR.
11. `https://bitbucket.org/Enr1g/p2p_mpotr.js` — p2p_mpOTR.js.
12. `https://whispersystems.org` — Open Whisper Systems.
13. `https://whispersystems.org/blog/simplifying-otr-deniability/` — Simplifying OTR deniability.
14. `https://whispersystems.org/blog/advanced-ratcheting/` — *Marlinspike M.* Advanced Cryptographic Ratcheting.
15. `https://whispersystems.org/blog/private-groups/` — *Marlinspike M.* Private Group Messaging.
16. `https://whispersystems.org/blog/facebook-messenger/` — Facebook Messenger deploys Signal Protocol for end-to-end encryption.
17. `https://whispersystems.org/blog/allo/` — Open Whisper Systems partners with Google on end-to-end encryption for Allo.

18. https://whispersystems.org/blog/whatsapp-complete/ — WhatsApp's Signal Protocol integration is now complete.
19. *Liu H., Vasserman E. Y., and Hopper N.* Improved group off-the-record messaging // Proc. 12th Workshop on Privacy in the Electronic Society. ACM, 2013. P. 249–254.
20. https://retroshare.readthedocs.io/en/latest/concept/topology/ — RetroShare Docs. Topology.
21. *Adrian D. et al.* Imperfect forward secrecy: How Diffie — Hellman fails in practice // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 5–17.
22. *Van Gundy M.* Improved Deniable Signature Key Exchange for mpOTR. http://matt.singlethink.net/projects/mpotr/improved-dske.pdf. 2013.
23. *Van Gundy M. D. and Chen H.* OldBlue: Causal Broadcast in a Mutually Suspicious Environment (Working Draft). http://matt.singlethink.net/projects/mpotr/oldblue-draft.pdf. 2012.
24. *Burmester M. and Desmedt Y.* A secure and scalable group key exchange system // Inform. Proc. Lett. 2005. V. 94. No. 3. P. 137–143.
25. *Douceur J. R.* The sybil attack // Intern. Workshop on Peer-to-Peer Systems. Berlin; Heidelberg: Springer, 2002. P. 251–260.
26. *Lesniewski-Laas C.* A Sybil-proof one-hop DHT // Proc. 1st Workshop on Social Network Systems. ACM, 2008. P. 19–24.
27. *Lesniewski-Laas C. and Kaashoek M. F.* Whanau: A sybil-proof distributed hash table // Proc. 7th USENIX Conf. on Networked Systems Design and Implementation (NSDI'10). 2010. P. 111–126.
28. *Danezis G. et al.* Sybil-resistant DHT routing // Europ. Symp. Research in Computer Security. Berlin; Heidelberg: Springer, 2005. P. 305–318.
29. *Loss D., Limmer T., and von Gernler A.* The Drunken Bishop: An Analysis of the OpenSSH Fingerprint Visualization Algorithm. http://dirk-loss.de/sshvis/drunken_bishop.pdf. 2009.
30. https://github.com/trevp/keyname — Keyname format for public-key fingerprints.
31. https://github.com/tomrittervg/crypto-usability-study/ *Ritter T. et al.* Crypto Usability Study.
32. https://tools.ietf.org/html/draft-miers-tls-sas-00 — Short Authentication Strings for TLS, Internet Draft.
33. http://philzimmermann.com/docs/PGP_word_list.pdf — PGP word list.
34. *Alexander C. and Goldberg I.* Improved user authentication in off-the-record messaging // Proc. Workshop on Privacy in Electronic Society. ACM, 2007. P. 41–47.
35. http://peerjs.com — The PeerJS Library.
36. https://rdist.root.org/2010/11/29/final-post-on-javascript-crypto/ *Lawson N.* Final post on Javascript crypto.
37. https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/ *Ptacek T.* Javascript Cryptography Considered Harmful.
38. https://nadim.computer/2013/05/23/critique-javascript-cryptography.html. *Kobeissi N.* Thoughts on Critiques of JavaScript Cryptography.
39. http://blog.kotowicz.net/2014/07/js-crypto-goto-fail.html?m=1 — JS crypto goto fail?
40. https://www.w3.org/TR/WebCryptoAPI/#security-considerations — Web Cryptography API, W3C Recommendation.

41. *Stark E., Hamburg M., and Boneh D.* Symmetric cryptography in javascript // Proc. Computer Security Applications Conf. (ACSAC'09). IEEE, 2009. P. 373–381.
42. http://caniuse.com/#search=webrtc — Can I Use? WebRTC Peer-to-peer connections.
43. http://webrtc-security.github.io — A Study of WebRTC Security.
44. *Hornby T.* Side-Channel Attacks on Everyday Applications: Distinguishing Inputs with FLUSH+RELOAD. https://www.semanticscholar.org/paper/Side-Channel-Attacks-on-Everyday-Applications-Dist-Hornby/a5ea83ad9abffe6c44b93617728e5f06f73bb9be?tab=citations. 2016.
45. *Oren Y. et al.* The spy in the sandbox: Practical cache attacks in javascript and their implications // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 1406–1418.

REFERENCES

1. *Borisov N., Goldberg I., and Brewer E.* Off-the-record communication, or, why not to use PGP. Proc. Workshop on Privacy in the Electronic Society, ACM, 2004, pp. 77–84.
2. http://www.bbc.com/news/world-us-canada-22837100 — Profile: Edward Snowden, BBC News.
3. http://www.globalissues.org/article/802/surveillance-state — Surveillance State: NSA Spying and more.
4. http://www.theguardian.com/media/2015/jul/02/wikileaks-us-spied-on-angela-merkels-ministers-too-says-german-newspaper — WikiLeaks: US spied on Angela Merkel's ministers too, says German newspaper, The Guardian.
5. *Unger N. et al.* SoK: secure messaging. Proc. IEEE Symp. Security and Privacy (SP), 2015, pp. 232–249.
6. https://www.eff.org/secure-messaging-scorecard — Electronic Frontier Foundation. Secure Messaging Scorecard.
7. *Unger N. and Goldberg I.* Deniable key exchanges for secure messaging. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 1211–1223.
8. *Korosteleva M. V. and Gamayunov D. Yu.* Obespecheniye kriptograficheski zashchishchennykh gruppovykh kommunikatsiy s funktsiyey otkazuyemosti [Protocol for secure group communications with deniability features]. Problemy Informatsionnoy Bezopasnosti. Komp'yuternyye Sistemy, 2014, no. 3, pp. 74–79.
9. *Goldberg I. et al.* Multi-party off-the-record messaging. Proc. 16th Conf. Computer and Communications Security, ACM, 2009, pp. 358–368.
10. https://github.com/maria-msu-seclab/mpotrDevelopment — Moscow State University Seclab mpOTR.
11. https://bitbucket.org/Enr1g/p2p_mpotr.js — p2p_mpOTR.js.
12. https://whispersystems.org — Open Whisper Systems.
13. https://whispersystems.org/blog/simplifying-otr-deniability/ — Simplifying OTR deniability.
14. https://whispersystems.org/blog/advanced-ratcheting/ — *Marlinspike M.* Advanced Cryptographic Ratcheting.
15. https://whispersystems.org/blog/private-groups/ — *Marlinspike M.* Private Group Messaging.
16. https://whispersystems.org/blog/facebook-messenger/ — Facebook Messenger deploys Signal Protocol for end-to-end encryption.
17. https://whispersystems.org/blog/allo/ — Open Whisper Systems partners with Google on end-to-end encryption for Allo.

18. https://whispersystems.org/blog/whatsapp-complete/ — WhatsApp's Signal Protocol integration is now complete.
19. *Liu H., Vasserman E. Y., and Hopper N.* Improved group off-the-record messaging. Proc. 12th Workshop on Privacy in the Electronic Society, ACM, 2013, pp. 249–254.
20. https://retroshare.readthedocs.io/en/latest/concept/topology/ — RetroShare Docs. Topology.
21. *Adrian D. et al.* Imperfect forward secrecy: How Diffie — Hellman fails in practice. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 5–17.
22. *Van Gundy M.* Improved Deniable Signature Key Exchange for mpOTR. http://matt.singlethink.net/projects/mpotr/improved-dske.pdf. 2013.
23. *Van Gundy M. D. and Chen H.* OldBlue: Causal Broadcast in a Mutually Suspicious Environment (Working Draft). http://matt.singlethink.net/projects/mpotr/oldblue-draft.pdf. 2012.
24. *Burmester M. and Desmedt Y.* A secure and scalable group key exchange system. Inform. Proc. Lett., 2005, vol. 94, no. 3, pp. 137–143.
25. *Douceur J. R.* The sybil attack. Intern. Workshop on Peer-to-Peer Systems, Berlin, Heidelberg, Springer, 2002, pp. 251–260.
26. *Lesniewski-Laas C.* A Sybil-proof one-hop DHT. Proc. 1st Workshop on Social Network Systems, ACM, 2008, pp. 19–24.
27. *Lesniewski-Laas C. and Kaashoek M. F.* Whanau: A sybil-proof distributed hash table. Proc. 7th USENIX Conf. on Networked Systems Design and Implementation (NSDI'10), 2010, pp. 111–126.
28. *Danezis G. et al.* Sybil-resistant DHT routing. Europ. Symp. Research in Computer Security, Berlin, Heidelberg, Springer, 2005, pp. 305–318.
29. *Loss D., Limmer T., and von Gernler A.* The Drunken Bishop: An Analysis of the OpenSSH Fingerprint Visualization Algorithm. http://dirk-loss.de/sshvis/drunken_bishop.pdf. 2009.
30. https://github.com/trevp/keyname — Keyname format for public-key fingerprints.
31. https://github.com/tomrittervg/crypto-usability-study/ *Ritter T. et al.* Crypto Usability Study.
32. https://tools.ietf.org/html/draft-miers-tls-sas-00 — Short Authentication Strings for TLS, Internet Draft.
33. http://philzimmermann.com/docs/PGP_word_list.pdf — PGP word list.
34. *Alexander C. and Goldberg I.* Improved user authentication in off-the-record messaging. Proc. Workshop on Privacy in Electronic Society, ACM, 2007, pp. 41–47.
35. http://peerjs.com — The PeerJS Library.
36. https://rdist.root.org/2010/11/29/final-post-on-javascript-crypto/ *Lawson N.* Final post on Javascript crypto.
37. https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/ *Ptacek T.* Javascript Cryptography Considered Harmful.
38. https://nadim.computer/2013/05/23/critique-javascript-cryptography.html. *Kobeissi N.* Thoughts on Critiques of JavaScript Cryptography.
39. http://blog.kotowicz.net/2014/07/js-crypto-goto-fail.html?m=1 — JS crypto goto fail?
40. https://www.w3.org/TR/WebCryptoAPI/#security-considerations — Web Cryptography API, W3C Recommendation.

41. *Stark E., Hamburg M., and Boneh D.* Symmetric cryptography in javascript. Proc. Computer Security Applications Conf. (ACSAC'09), IEEE, 2009, pp. 373–381.
42. `http://caniuse.com/#search=webrtc` — Can I Use? WebRTC Peer-to-peer connections.
43. `http://webrtc-security.github.io` — A Study of WebRTC Security.
44. *Hornby T.* Side-Channel Attacks on Everyday Applications: Distinguishing Inputs with FLUSH+RELOAD. `https://www.semanticscholar.org/paper/Side-Channel-Attacks-on-Everyday-Applications-Dist-Hornby/a5ea83ad9abffe6c44b93617728e5f06f73bb9be?tab=citations`. 2016.
45. *Oren Y. et al.* The spy in the sandbox: Practical cache attacks in javascript and their implications. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 1406–1418.