Криптографический протокол обмена ключами децентрализованного характера в среде централизованных сервисов связи

Коваленко Геннадий Александрович

Аннотация. Разработка протоколов обмена ключами между собеседниками / узлами в сети всегда является крайне важной и сложной задачей в проблематике безопасных коммуникаций, потому как исходит из возможных нападений и специфики самой системы, в которой обмен должен корректно происходить. Централизованные и децентрализованные способы обмена ключами представляют собой два разнородных способа функционирования таковых механизмов. В то время как централизованные механизмы уже действует и функционируют в повседневности, децентрализованные же механизмы, за счёт этого, кажутся более эзотерическими и мало изученными, что не всегда является таковым. Децентрализованные системы также имеют возможность обмениваться ключами, но исходя лишь из своей ризоморфной парадигмы.

Ключевые слова: протокол обмена ключами; криптографический протокол; децентрализованные сети; централизованные системы; сеть доверия; MITM-атака;

Содержание

1. Введение

При разработке децентрализованных систем всегда остро стоит проблема аутентификации субъектов при обмене публичными ключами как между непосредственными собеседниками, так и между узлами в самой сети. Проблема исходит из атак типа MITM (man in the middle), осуществление которых приводит не только к череде пассивных нападений в лице прослушивания транслируемого трафика, но и к активным нападениям, в лице его подмены. Такой исход вызывает сопутствующие проблемы доверия безопасности всех накладываемых коммуникаций.

2. Проблематика

Проблема обмена ключами исходит из классической криптографии, когда не существовало какого бы то ни было раздела асимметричной криптографии. Проблема звучала достаточно просто: возможно ли передать симметричный ключ шифрования безопасно так, чтобы третья сторона не смогла его перехватить и прочитать?

Рисунок 1. Использование небезопасного канала связи при передаче ключа

За четыре тысячелетия классической криптографии так и не был дан корректный ответ на поставленный вопрос. Лишь с приходом современной криптографии, когда классическая форма постепенно приобретала новый облик науки, появлялся и новый раздел криптографии - асимметричная криптография [1], сутью которой стала возможность безопасной передачи симметричного ключа, использовав при этом небезопасный канал связи.

Головоломки Меркла (1974), протокол Диффи-Хеллмана [DH] (1976), алгоритм RSA (1976), протокол Месси-Омуры [MO] (1978), ранцевая криптосистема Меркла-Хеллмана (1978), криптосистема Рабина (1979), схема Эль-Гамаля [EG] (1985), и последующие вариации DH, EG, MO на эллиптических кривых породили возможность решения огромного спектра прикладных задач, которым важна была безопасность коммуникаций, начиная с обычной потребности в общении, и заканчивая банковскими транзакциями.

Внешне может показаться, что при таком развитии криптографии, и в частности её асимметричного раздела, ранее существовавшая и старая проблема классической криптографии более становится не актуальной. Но на самом деле, асимметричная криптография не решает окончательно первоначальную проблему, а лишь сдвигает "ареал обитания атакующих" с пассивных до активных нападений, заменяя проблему передачи симметричного ключа на проблему передачи публичного ключа.

Тем не менее, даже такой результат уже является достаточно позитивным, потому как злоумышленнику становится необходимо совершать дополнительные действия и трудозатраты, которые, по истечению времени, могут ещё и выдать факт его существования субъектам коммуникации (если таковые смогут лично встретиться и проверить передаваемые значения), но лишь как постфактум.

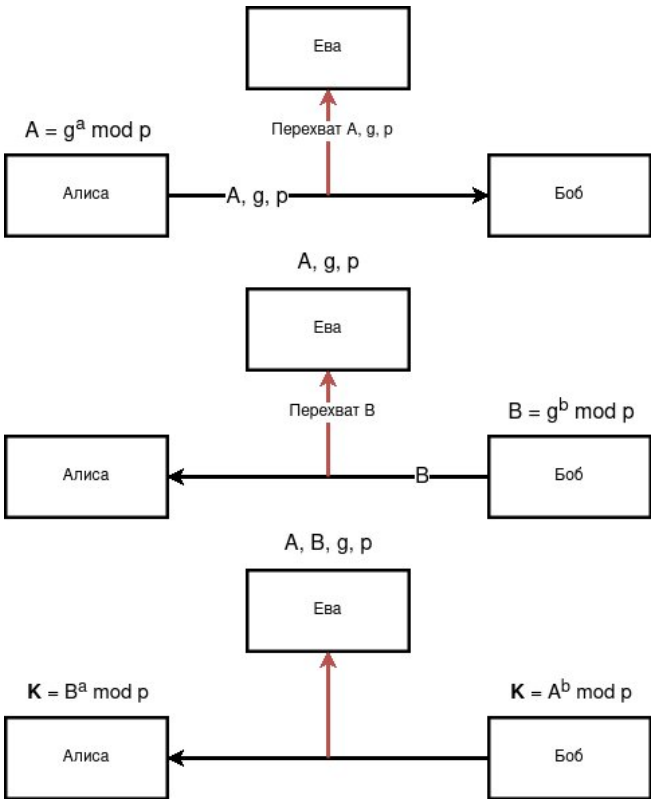

Рисунок 2. Использование небезопасного канала связи при генерации ключа (протокол Диффи-Хеллмана)

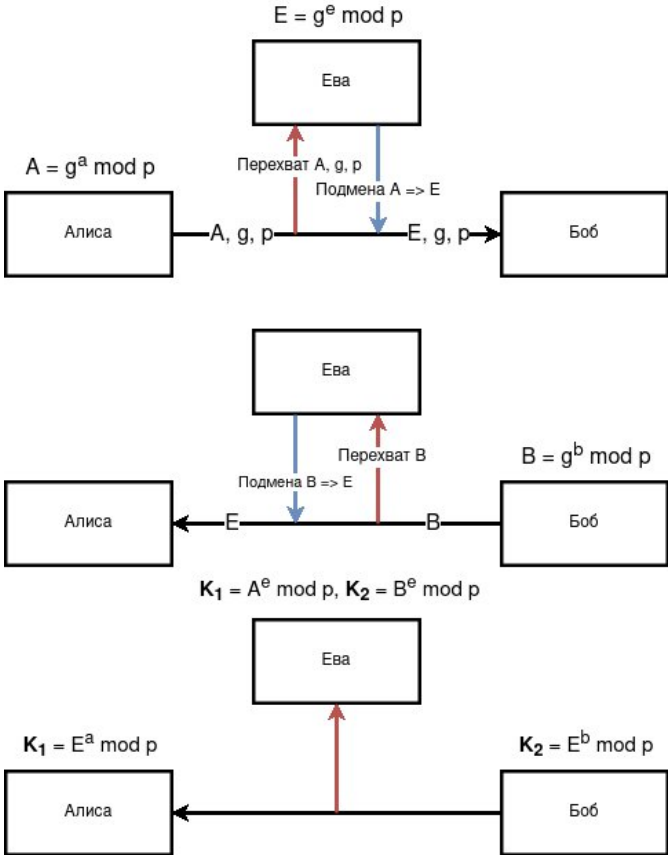

Рисунок 3. Атака MITM на примере протокола Диффи-Хеллмана

3. Решение централизации

Современный Интернет со всей текущей рекламой, коммерцией, онлайн покупками и продажей просто не мог бы адекватно существовать, если бы постоянно совершались злоумышленниками сопутствующие MITM-атаки. И действительно, если бы такое существовало в реальности, то можно было бы пересчитать на пальцах компании, которые были бы готовы смириться с рисками кражи передаваемых финансов и с постоянным снижением уровня доверия их же клиентов. Да и сами клиенты, принимая отрицательную сторону платежей через Интернет, просто бы продолжали пользоваться наличными деньгами. В итоге, единственной безотказно рабочей бизнес-моделью в Интернете оставались бы сами MITM-атаки.

Тем не менее, настоящая реальность показывает нам, что проблема MITM как-то решается и по ощущениям довольно успешно. Когда мы пользуемся браузером, то он нам может показывать три возможных состояния коммуникаций: небезопасно (http://), безопасно (https://) и ещё раз небезопасно (~~https~~://). Первое говорит просто о том, что не существует вовсе шифрования. Второе утверждает, что всё безопасно шифруется и подтверждается. И как раз последнее свидетельствует о том, что возможно осуществление MITM-атаки.

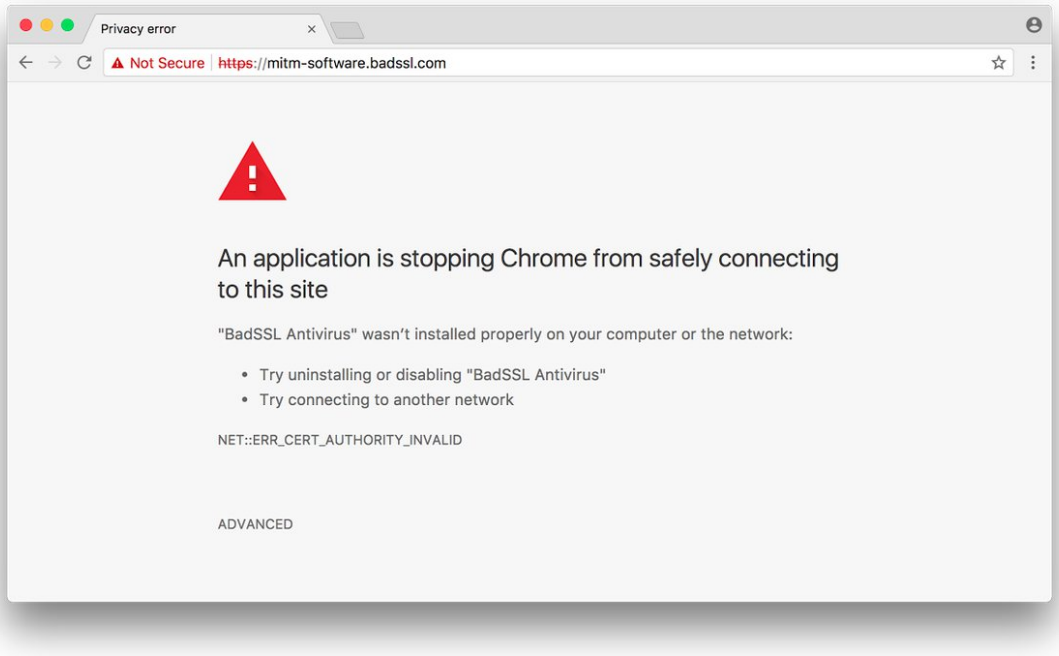

Рисунок 3. Атака MITM на примере протокола Диффи-Хеллмана

Так в итоге, как современный мир смог решить проблему MITM? Ответ: достаточно просто — делегировав возможность совершения MITM атаки ограниченному кругу сервисов, выдвигаемых в роли центров сертификации (доверенных узлов). Схема крайне проста, но на своих первоначальных этапах она была также уязвима к MITM со стороны сторонних злоумышленников.

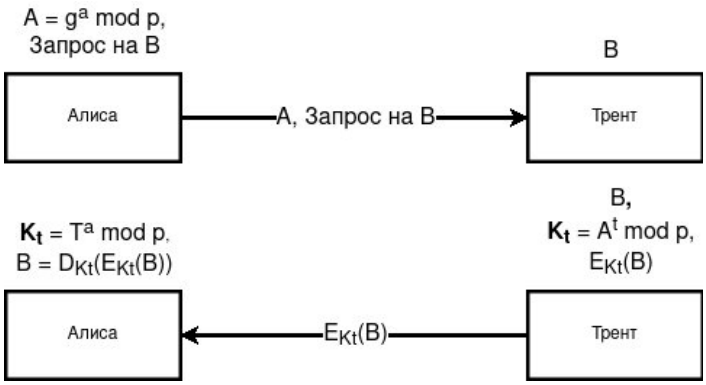

Рисунок 4. Трент - доверенный узел. Предполагается, что Алиса уже знает Трента, а потому и знает **T**. Доверие абсолютно, и Алиса предполагает, что Трент не будет совершать MITM. Под E_{Kt} понимается алгоритм, способный доказать Алисе, что само сообщение отправил и подтвердил именно Трент.

Иными словами, при работе за компьютером с доступом в Интернет, уже предполагается, что на таковом компьютере, на уровне ОС, браузера или конкретного приложения установлены сертификаты (публичные ключи) центров сертификации, за счёт которых и совершаются дальнейшие безопасные коммуникации. Но стоит понимать, что как-то ОС, браузер, приложение, должны были попасть на итоговый компьютер без доверенного узла. Здесь может быть несколько ветвлений развития со стороны централизации.

1. Первый и самый очевидный вектор — просто плыть по течению и рано или поздно, даже с учётом жертв MITM, большая часть пользователей наконец установит обновлённую ОС, браузер или приложение с вложенными в них публичными ключами центров сертификации.

2. Второй вектор — использовать кооперацию с производителями устройств, чтобы таковые, устанавливали ОС с уже вложенными публичными ключами центров сертификации. Таковой способ вполне удовлетворителен, но не действителен для клиентов у которых уже присутствуют устройства.

Как только информация о центрах сертификации будет установлена на клиентском устройстве, то все дальнейшие MITM атаки начинают делегироваться. Общий вид MITM со сторонними злоумышленниками теряет свой первоначальный смысл. Теперь MITM могут совершать лишь "законно установленные" злоумышленники в роли ЦС.

4. Решения децентрализации

Децентрализованные системы часто решают проблемы MITM атак более изощрёнными, эзотерическими способами за счёт невозможности создания центров сертификации, которые бы явно противоречили ризоморфной структуре. Тем не менее, как далее мы опишем, децентрализованные сети часто на своих первоначальных стадиях запуска будут использовать централизованные системы, как и до этого, сами централизованные системы использовали небезопасные каналы связи при установке ЦС.

1. Первый метод обмена ключами в децентрализованных сетях — это обмен ключами лично в оффлайне. Иными словами, метод сводится к использованию простого, старого,

надёжного и проверенного временем в тысячелетия способа обмена. Безусловно это не самый удобный способ, потому как может существовать большое количество его ограничений — невозможность передачи ключа явно, географическое расположение не позволяющее осуществить передачу, небезопасность использования посредника при транспортировании и прочее. В любом случае, такой способ хоть и не имеет технической базы, но при это является одним из самых надёжных методов передачи.

2. Второй метод сводится к использованию централизованных сервисов в роли площадок для размещения публичных ключей. Часто создатели децентрализованных систем сами создают сервер, на котором выкладывают и постоянно редактируют список публичных ключей, выступая тем самым в роли некого ЦС, хоть и на более низком уровне по иерархии доверия. Как только первоначальная децентрализованная сеть была построена и сервер, в роли ЦС, сыграл свою основную роль *корректно*, сами участники могут обмениваться внутри сети публичными ключами, в некой степени эмулируя применение *первого* способа обмена. Подменить публичные ключи становится возможным на трёх уровнях: 1) на уровне оригинального ЦС, где таковой должен сотрудничать с провайдерами связи для успешных редиректов на копию сервера с другими публичными ключами, 2) на уровне выдвигаемого сервера в роли ЦС, где таковой может самолично изменять публичные ключи, 3) на уровне децентрализованной сети, если произошёл либо первый, либо второй пункт, либо если сам узел является злоумышленником, выдающим ложные ключи.

3. Третий метод сводится к использованию сети доверия [2]. Данный способ успешно может функционировать в роли продолжения второго, после того как клиенты получили первоначальные публичные ключи от сервера. Каждый указанный публичный ключ на сервере сам становится своеобразным ЦС, который перенаправляет публичные ключи от одного узла к другому. Далее, как только сами клиенты начинают обладать определённым количеством ключей, они также автоматически становятся ЦС, устанавливающими и выстраивающими на своей стороне дальнейшие коммуникации. Указанный механизм интересен тем, что уровень централизованного доверия к узлам постоянно "разлагается". Изначально существовавший один сервер обладает *100%* уровнем доверия, далее список публичных ключей в *N* уменьшает, в лучшем случае, первоначальное доверия до *100%/N*, и далее, сами клиенты подключившись к узлам в *Q-ом* количестве, продолжают уменьшать необходимый уровень доверия, в лучшем случае, до *100%/N/Q* и т.д. Сеть доверия будет работать лишь при условии, что *N* и *Q* не приводят всё к тому же *100%* уровню доверия, иначе говоря, если *N* и *Q* не равны единице. Поэтому следует понимать под *N и Q* не просто количество узлов, а количество узлов не подменяющих информацию. Если количество узлов не подменяющих информацию равно нулю, то следует установить *N или Q = 1*.

4. Четвёртый метод сводится к использованию уже существующих децентрализованных сервисов в роли площадок для размещения публичных ключей. Как пример, используя некий блокчейн *X*, мы можем внести криптовалюту в свой аккаунт, вставив публичный ключ в определённо договорённый (с абонентом) интервал времени *T* или блок *B*. Далее, мы ссылаемся на конкретный блок *B* и интервал времени *T* для идентификации своего публичного ключа. Злоумышленнику в таком сценарии становится необходимо успеть поместить публичный ключ в интервал *T* или блок *B*, заменив до этого сообщение о публичном ключе на стороне абонента.

5. Пятый метод сводится к использованию уже существующих централизованных сервисов как ретрансляторов публичных ключей от точки *A* до точки *B*. Внешне пятый способ может быть схож со вторым, по причине использования централизованных механизмов, но внутренне он отличается достаточно сильно. Так например, во втором способе публичные ключи размещаются на централизованном сервисе, а далее по запросу таковой список просто выгружается уже в качестве ответа. В пятом же способе, передача публичных ключей связана непосредственно с криптографическим протоколом, позволяющим с определённой степенью регулируемой вероятности передать публичный ключ не только без последующей подмены централизованными сервисами, но и при этом не раскрывая сам передаваемый публичный ключ, что может быть необходимо / полезно при условии, что децентрализованная сеть является анонимной.

4.1. Протокол пятого метода

Использование централизованных сервисов при обмене публичными ключами звучит противоречиво, потому что сами же централизованные сервисы могут с бо́льшим шансом / успехом подменять публичные ключи. Поэтому, чтобы снизить риски подмены, разработчики децентрализованных систем создают собственные серверы, как это было описано во *втором* методе. Тем не менее, весь подход сводится к двум моментам.

1. Во-первых, связь с централизованными сервисами уже по умолчанию защищена ЦС, иными словами мы начинаем искоренять всеразличных злоумышленников стоящих между отправителем и получателем в децентрализованных сетях. В таком случае мы просто сужаем спектр всех возможных атакующих до централизованных сервисов.

2. Во-вторых, должно быть выбрано несколько централизованных сервисов наиболее несвязанных между собой. Например, facebook и vkontakte, telegram и signal и т.д. Иными словами, необходимо "разложить" централизацию индивидуальных сервисов на децентрализацию множества сервисов.

Далее, механизм связи сводится к передаче одного публичного ключа сразу по нескольким централизованным сервисам. Если ключи все пришли одинаковые, то велика вероятность того, что ключ не был подменён. Если малая часть ключей была подменена, то можно взять в качестве правды - ключ с бо́льшим процентом однотипности, либо совершить вновь процедуру отправления ключа, но исключая предыдущие сервисы (которые выдали ключ в ме́ньшей пропорции) и заменяя их другими сервисами.

Плюс такой схемы в том, что мы используем уже готовую (централизованную) инфраструктуру для обмена ключами, в отличие от Web of Trust, требующей создавать собственную инфраструктуру открытых ключей.

Предполагается, что у *A* и *B* есть своя пара открытый/закрытый ключ. Для *A* - это (*PubA/PrivA*), для *B* - это (*PubB/PrivB*),

1. Пользователь *B* генерирует временную пару открытый/закрытый ключ - (*PubT/PrivT*) и отправляет открытый ключ *PubT* пользователю *A* через *N* централизованных сервисов прямо несвязанных[1] между собой.

[1] Под несвязанностью понимается отсутствие общих директоров, инфраструктуры и прочего,

2. Пользователь *A* получает открытые ключи $PubT_1$, $PubT_2$, $PubT_3$, ..., $PubT_N$ с централизованных сервисов соответственно *1, 2, 3, ..., N* и сравнивает, все ли они одинаковые. Если больше половины ключей расходится ($\lim_{C \to N/2}$) - пользователи *A* и *B* выбирают совершенно новые *N* централизованных сервисов и начинают процедуру заново.

3. Если меньше половины ключей расходится ($\lim_{C \to N}$) - пользователи *A* и *B* выбирают под несогласованный публичный ключ новые сервисы связи и повторяют процедуру конкретно с ними.

4. Если результат выбранных сервисов расходится с результатом предыдущего большинства, то следует вновь поменять сервисы. Если ряд выбираемых сервисов выдают результат отличный от предыдущего большинства и в количестве данный ряд превышает предыдущее большинство, то следует вернуться на условие *процедуры 2*.

5. Далее пользователь *A* зашифровывает открытым ключом *PubT* свой открытый ключ *PubA* и отправляет полученный результат также на выбранные ранее *N* сервисов:

$$CPubA = E(PubT, PubA).$$

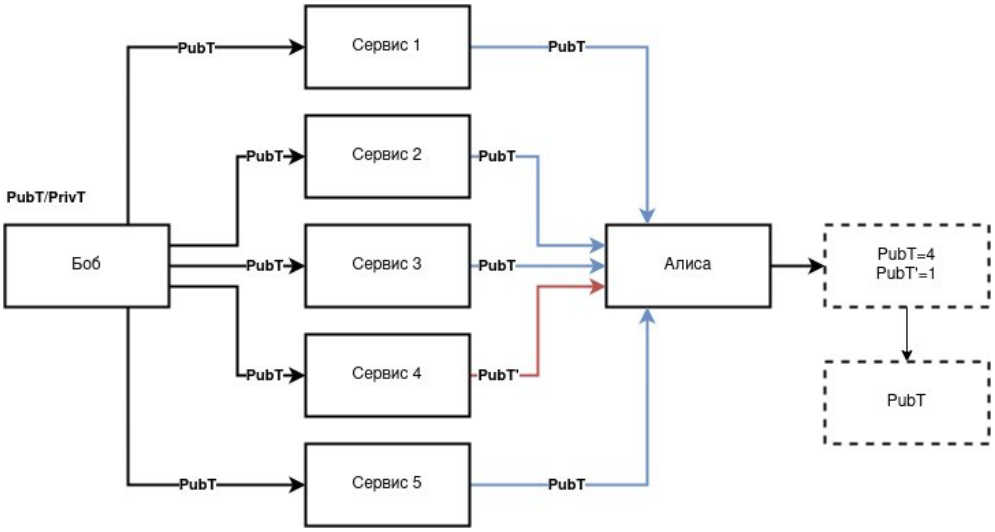

Рисунок 5. Схема получения временного публичного ключа PubT

6. Пользователь *B* просматривает корректность получения *CPubA*. Если шифрованные ключи расходятся, то применяются процедуры *2, 3, 4* направленные на уже шифрованную версию ключей.

7. При успешном получении шифрованного публичного ключа *CPubA*, пользователь *B* применяет временный закрытый ключ *PrivT* и расшифровывает *CPubA*, получая тем самым публичный ключ *PubA*:

$$PubA = D(PrivT, CPubA).$$

допустим vk.com и mail.ru считаются связанными между собой сервисами.

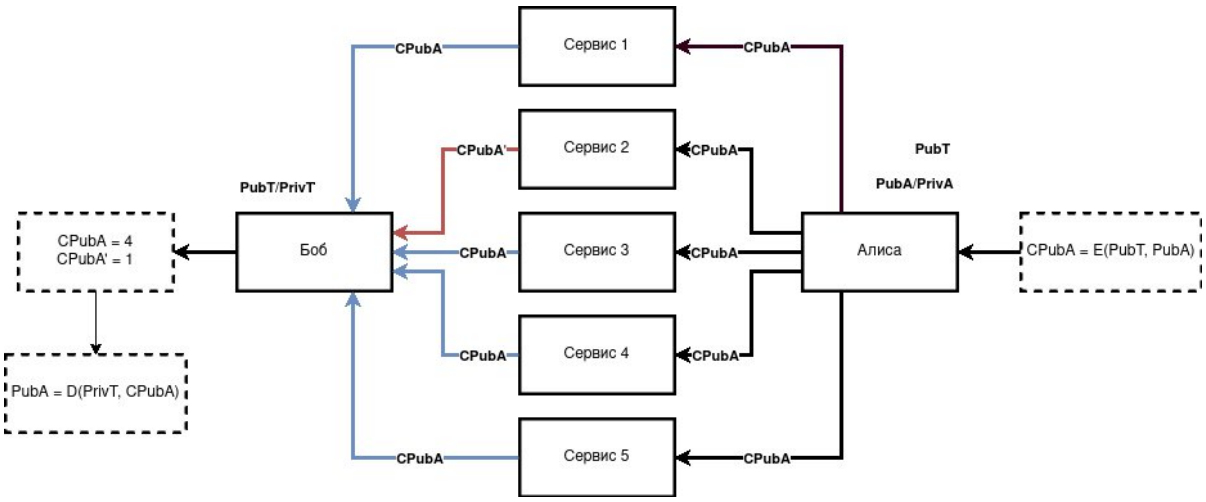

Рисунок 6. Схема получения публичного ключа PubA

8. Далее, пользователь *B* подписывает свой публичный ключ *PubB* временным закрытым ключом *PrivT*, и шифрует результат ранее полученным публичным ключом *PubA*. Результат подписания и шифрования пользователь *B* отправляет пользователю *A*. Пользователь *B* может использовать любой сервис связи, т.к. в данном случае уже нельзя будет корректно подменить информацию за счёт необходимости нарушения подписи для *PrivT*:

$$SPubB = S(PrivT, PubB),$$
$$CSPubB = E(PubA, SPubB).$$

9. Пользователь *A* принимает *CSPubB*, расшифровывает его своим публичным ключом *PubA*, получая тем самым *SPubB*. Далее пользователь *A* проверяет подпись, используя временный публичный ключ *PubT*, получая тем самым истинность публичного ключа *PubB*:

$$SPubB = D(PubA, CSPubB),$$
$$PubB = V(PubT, SPubB).$$

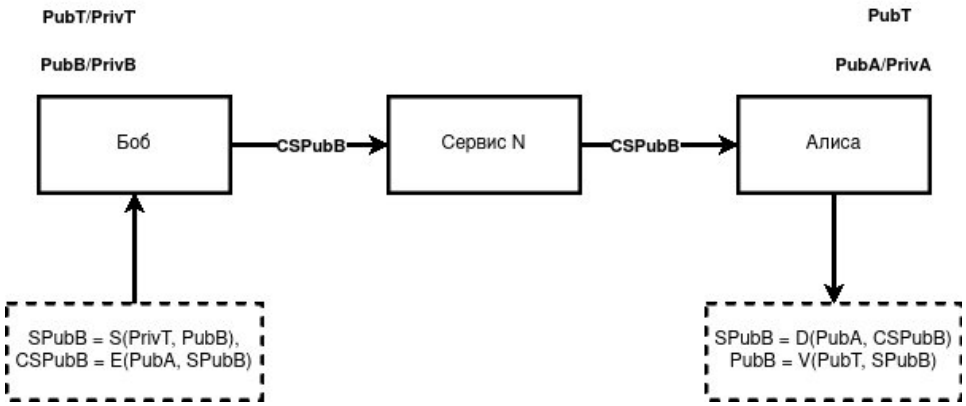

Рисунок 7. Схема получения публичного ключа PubB

Таким образом, пользователи *A* и *B* обменялись своими публичными ключами, воспользовавшись сторонними централизованными сервисами связи. При этом, в вышеописанном протоколе централизованные сервисы так и не узнали истинные публичные

ключи, которые будут применяться в качестве дальнейших идентификаторов *ID* в децентрализованной сети, а потому и не смогут их использовать для связывания с сетевым адресом *IP* (анонимные сети).

У централизованных сервисов в полномочии остались только *PubT, CPubA, CSPubB*. Этапы 2, *3, 4* защищают от активных атак за счёт однотипных действий на несвязанных между собой сервисах.

Протокол не лишён проблем:

1. Протокол является *вероятностным*. Существует вероятность, что централизованные сервисы смогут связаться и успешно подменять публичные ключи хотя бы для *4/1* сервисов, что будет достаточно, т.к. последний сервис будет считаться уже некорректным.

2. Если *A* или *B* является злоумышленником, а децентрализованная сеть является анонимной и чувствительной ко связи *ID=IP*, то злоумышленник, получив публичный ключ своего абонента, получит тем самыми и его идентификатор *ID* в сети. При сотрудничестве с сервисами связи, он сможет легко узнать и *IP* абонента. Таким образом, легко свяжет полученный публичный ключ с сетевым адресом.

3. Предполагается, что пользователи *A* и *B* уже друг друга идентифицируют на выбираемых ими централизованных сервисах. В противном случае, пользователи *A* и *B* не будут знать кому отправляют и от кого получают ключи.

5. Заключение

В работе были представлены способы обмена ключами как в централизованных системах, так и в децентрализованных. При разборе децентрализованных систем был подробно описан криптографический протокол, позволяющий обмениваться публичными ключами за счёт использования множества централизованных сервисов. Протокол может применяться при обмене ключами для последующего их использования в анонимных сетях за счёт сокрытия истинного публичного ключа при передаче. Недостатком протокола является его вероятностная структура, успешность результата которой зависит от количества централизованных сервисов связи и их несвязываемости между собой.

Список литературы

1. Diffie, W., Hellman, M. New Directions in Cryptography [Электронный ресурс]. — Режим доступа: https://ee.stanford.edu/~hellman/publications/24.pdf (дата обращения: 19.12.2020).
2. Hung-Yu C., Dynamic Public Key Certificates with Forward Secrecy [Электронный ресурс]. — Режим доступа: https://www.mdpi.com/2079-9292/10/16/2009 (дата обращения: 01.08.2023).