МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

УДК 519.7

О БЕЗОПАСНОСТИ ПРОТОКОЛА SESPAKE

Е. К. Алексеев, С. В. Смышляев

ООО «КРИПТО-ПРО», г. Москва, Россия

Протокол SESPAKE описан в Рекомендациях по стандартизации Р 50.1.115-2016 «Информационная технология. Криптографическая защита информации. Протокол выработки общего ключа с аутентификацией на основе пароля». В настоящей работе приводятся результаты криптографического анализа данного протокола в релевантных моделях противника. Вводится новая модель противника с угрозой ложной аутентификации, являющаяся расширением базовой модели, применяемой для протоколов с внутренней диверсификацией ключей. Стойкость протокола оценивается в двух моделях: в модели на основе задачи отличения и в расширенной.

Ключевые слова: *модели и методы защиты информации, криптографические протоколы.*

DOI 10.17223/20710410/50/1

ON SECURITY OF THE SESPAKE PROTOCOL

E. K. Alekseev, S. V. Smyshlyaev

CryptoPro, Moscow, Russia

E-mail: alekseev@cryptopro.ru, svs@cryptopro.ru

The Security Evaluated Standardized Password Authenticated Key Exchange (SESPAKE) protocol is standardized in Russia as R 50.1.115-2016. The current paper provides analysis of the protocol in relevant adversary models. We define new indistinguishability-based adversary model with a threat of false authentication that is an extension of the original indistinguishability-based model up to the case of protocols with authentication step without key diversification. We prove the protocol security in two adversary models with a classic threat of distinguishing a generated session key from a random string and with a threat of false authentication.

Keywords: *models and methods in information security, cryptographic protocols.*

Введение

Суть протоколов, которым посвящена данная работа, состоит в том, что стороны, изначально разделяя лишь малоэнтропийный ключ (который обычно называют паролем), в процессе взаимодействия вырабатывают высокоэнтропийный общий ключ. Специфичное требование, предъявляемое к таким протоколам, таково: даже активный противник, имеющий доступ к каналу связи, не должен получить информацию,

достаточную для подбора пароля «off-line», то есть без взаимодействия с легитимными участниками. Являясь по сути протоколами выработки общего ключа с аутентификацией на основе пароля, протоколы такого типа часто объединяют под общим названием «PAKE» — «Password Authenticated Key Exchange».

Первый протокол, получивший название EKE, предложен S. Bellovin и M. Merritt в 1992 г. [1]. На сегодняшний день существует множество протоколов данного типа (см., например, [2 – 6]). В 1993 г. M. Bellare и P. Rogaway в работе [7] предложили *модель на основе неотличимости* для оценки стойкости протоколов аутентифицированной выработки общего ключа. Эта модель была расширена в [8] для анализа PAKE-протоколов (обычно её называют BPR-моделью). Она позволила получить оценки стойкости для некоторых протоколов данного типа [9].

Структурно протоколы типа AKE (и PAKE в частности) обязательно содержат этап выработки ключа и опционально — этап его подтверждения. Нижние оценки стойкости получены как для протоколов без этапа подтверждения [9], так и для ряда протоколов, содержащих этот этап [4]. В обоих случаях доказывается стойкость относительно угрозы отличения выработанного общего ключа от случайной строки. Однако не для всех протоколов, содержащих этап подтверждения ключа, удаётся корректно определить такую угрозу. В случаях, когда на этапе подтверждения доказывается владение непосредственно выработанным ключом, у противника естественным образом появляется критерий, позволяющий проверить, является ли целевая строка случайной. На это явно указывают J. Bender, M. Fischlin и D. Kugler в [10] (п. «The final authentication step» в разделе с описанием протокола PACE), такая же проблема возникает при оценке стойкости протокола TLS 1.2 (см. аннотацию работы [11]). В работах, в которых получены оценки стойкости для протоколов с подтверждением ключа, проблема корректного определения угрозы отличения решается на уровне строения протокола путём введения дополнительного «ветвления» полученных ключей. Это означает, что ключ для этапа подтверждения производится из общего секрета одним необратимым образом, а основной ключ производится из этого же секрета другим необратимым образом.

В настоящей работе исследуется протокол SESPAKE аутентифицированной выработки общего ключа на основе пароля, содержащий этап подтверждения ключа. При этом на этапе подтверждения используется непосредственно сам выработанный в процессе взаимодействия сторон секрет (то есть отсутствует упомянутое ранее «ветвление»). Этап выработки ключа является модификацией протокола, предложенного в [9], этап подтверждения ключа является оригинальным. Для данного протокола предложена модель на основе неотличимости с угрозой ложной аутентификации, которая формализуется с помощью задачи отличения настоящей аутентификационной информации от случайной. Доказательство оценки стойкости протокола SESPAKE в этой модели делится на три этапа, первый из которых существенно основывается на идеях, представленных в работе [9]. Второй и третий этапы являются оригинальными. На втором и третьем этапах доказательства использованы приёмы, которые могут помочь при анализе и обосновании стойкости других протоколов типа PAKE. Насколько известно авторам, представленный протокол является первым протоколом без ветвления ключей на этапе подтверждения, для которого предложена модель и в этой модели получена оценка стойкости.

Обзор уязвимостей протоколов данного семейства, а также принципы построения протокола SESPAKE, стандартизированного в России [12], представлены в работе [13].

1. Модель противника и оценки стойкости известных протоколов

Опишем модель BPR, которая используется в настоящей работе для оценки стойкости этапа выработки ключа протокола SESPAKE, и сделаем краткий обзор известных результатов, касающихся оценок стойкости PAKE-протоколов. Рассмотрим также общие принципы моделирования противника, применяемые в области практически ориентированной доказуемой стойкости.

В [14] выделены следующие типы моделей противника: на основе неотличимости, на основе универсальной композиции и на основе симулирования. В настоящей работе стойкость протокола исследована в модели на основе неотличимости; рассмотрим эту модель противника подробнее.

Все результаты относительно стойкости протоколов типа PAKE, известные авторам, относятся к области так называемой практически ориентированной доказуемой стойкости, принципы которой описаны в [15]. Особенность этого подхода состоит в том, что он позволяет анализировать стойкость конкретных схем, не обладающих бесконечно возрастающим параметром безопасности. Данный подход использует классическую для теории сложности технику сведения одной задачи к другой, а результатом проведённого анализа является верхняя оценка возможностей (например, вероятности успешного осуществления угрозы) противника, зависящая от доступных ему вычислительных и информационных ресурсов (подробнее процесс получения оценок описан, например, в [16, разд. 8.3.1]). Ограничения этих ресурсов, продиктованные конкретными условиями эксплуатации оцениваемой криптосистемы, позволяют получить численные оценки её стойкости. Таким образом, вопросы выбора целевого порогового значения стойкости и, как следствие, допустимости использования криптосистемы могут быть решены исключительно на этапе её встраивания в более высокоуровневую систему. Вне такого контекста получаемые оценки можно прокомментировать с точки зрения предполагаемых границ применения протокола.

1.1. Общие принципы формализации противника

Под *противником* будем понимать вероятностную интерактивную машину Тьюринга. Неформально это программа, которая может делать запросы к другим программам и отвечать на запросы, сделанные к ней самой. Формальное определение с подробными пояснениями можно найти в [17, Definitions 4.2.1, 4.2.2].

Мера успешности противника при решении задач, определяемых моделью, называется *преимуществом противника* и обозначается $\mathbf{Adv}(\mathcal{A})$. Обычно это обозначение снабжается верхним и нижним индексами, которые указывают соответственно на используемую модель и конкретный анализируемый в её рамках криптографический объект.

Задачи противника, которые подразумевают бинарный ответ, будем называть *задачами распознавания*. В таких задачах SUCC — это событие, состоящее в том, что результат работы противника совпал со значением неизвестного ему бита b, выбранного случайно в процессе его модельного взаимодействия с анализируемой криптосистемой. Для противника $\mathcal{A}$, решающего задачу распознавания $Task$, принято оценивать не вероятность успеха, а величину $\mathbf{Adv}^{Task}(\mathcal{A})$, которая определяется следующим образом:

$$\mathbf{Adv}^{Task}(\mathcal{A}) = 2\Pr[\mathrm{SUCC}] - 1.$$

Отсутствие в данном определении модуля объясняется тем, что для любого противника с отрицательным значением $\mathbf{Adv}^{Task}(\mathcal{A})$ существует противник $\mathcal{A}'$, который работает точно так же, как $\mathcal{A}$, но возвращает противоположные ему значения.

Если задача $Task$ вычислительная (результатом работы является значение из достаточно большого множества), то через $\mathbf{Adv}^{Task}(\mathcal{A})$ обозначается вероятность вычисления противником искомого значения (например, ключа криптосистемы).

Вычислительными ресурсами противника называют сумму максимального времени его работы и размера записи кода его программы. В зависимости от решаемой задачи противник может иметь возможность осуществлять обращения к одному или нескольким оракулам. Через $\mathcal{A}(t, q_1, q_2, \ldots, q_k)$ будем обозначать множество противников, которые обладают вычислительными ресурсами, не превышающими t, и делают не более $q_1, q_2, \ldots, q_k$ запросов к оракулам, доступным им в рамках используемой модели. Через $\mathbf{Insec}^{Task}(t, q_1, q_2, \ldots, q_k)$ будем обозначать следующую величину, называемую для краткости *нестойкостью задачи* $Task$:

$$\mathbf{Insec}^{Task}(t, q_1, q_2, \ldots, q_k) = \max_{\mathcal{A} \in \mathcal{A}(t, q_1, q_2, \ldots, q_k)} \mathbf{Adv}^{Task}(\mathcal{A}).$$

В том случае, когда требуется рассмотреть множество противников, ограниченное не по всем параметрам (или какие-то ограничения не имеют смысла для рассматриваемой задачи), лишние параметры не указываются.

Подробнее этот подход к формализации задачи оценки стойкости криптосистем обсуждается в работе [18].

1.2. Модель BPR

Опишем модель BPR для протоколов, предполагающих взаимодействие двух сторон (иногда мы также будем называть их участниками). Процесс реализации действий, предписываемых протоколом, будем называть сессией. В рамках различных сессий может взаимодействовать несколько сторон, совокупность которых будем называть сетью.

Традиционно для этого подхода в криптографии возможности противника по взаимодействию с криптосистемой в BPR-модели моделируются с помощью предоставления ему доступа к ряду *оракулов*, являющихся интерактивными машинами Тьюринга. Оракулы в BPR-модели можно разделить на два типа:

— Оракулы, моделирующие участников сети (далее — оракулы-участники): они носят технический характер и выделяются больше для ясности изложения и прозрачности проецирования модели на практику. Противник напрямую не может делать запросы к этим оракулам.

— Оракулы, моделирующие различные возможности противника: к этим оракулам у противника есть непосредственный доступ. При этом они задействуют оракулов из предыдущего пункта.

Если $\mathcal{O}$ — некоторый оракул, принимающий на вход запросы из множества R, то через $\mathcal{O}(r)$, $r \in R$, будем обозначать значение, возвращаемое оракулом $\mathcal{O}$ в качестве ответа на запрос r.

Заметим, что использование понятия «оракул» для моделирования в математической криптографии является стандартным приёмом, позволяет не вводить лишних конструкций и не приводит к внутренним противоречиям.

Оракулы-участники. Прежде чем переходить к описанию оракулов первого типа, введём некоторые сопутствующие понятия.

Множество участников сети является объединением непустых и непересекающихся множеств Client и Server, состоящих из строк-идентификаторов клиентов и серверов соответственно. Клиент может взаимодействовать с одним сервером, для этого используется секретный параметр, порождённый независимо от секретных параметров

других участников сети. Сервер может взаимодействовать с несколькими клиентами и хранит список своих секретных параметров, необходимых для такого взаимодействия в соответствии с протоколом. Заметим, что секретные параметры клиента и сервера не обязательно равны друг другу. Если они связаны таким образом, что могут быть эффективно получены друг из друга, то говорят, что протокол является протоколом *симметричного типа*, иначе — протоколом *асимметричного типа*.

Оракулы первого типа моделируют работу участников сети. Потенциально количество таких оракулов в модели не ограничено. Причина этого в том, что каждый из участников может осуществлять взаимодействие в рамках многих сессий, поэтому каждому оракулу соответствует пара (U, i), где U — идентификатор участника сети; i — номер сессии участника U, взаимодействие в которой моделируется этим оракулом. Часто можно встретить аналогию, что каждый такой оракул соответствует процессу операционной системы, реализующему протокольное взаимодействие некоторого пользователя. Оракулы, к которым не было запросов, находятся в одинаковом состоянии ожидания начала взаимодействия. Запросом к оракулу является строка, которую он интерпретирует как сообщение, пришедшее к нему из канала связи. Ответом оракула является строка, сформированная им в соответствии с протоколом и его текущим внутренним состоянием. Особым образом определяется первый запрос к оракулам, соответствующим клиентам сети, так как по протоколу клиент отправляет первое сообщение сам. В модели BPR первым запросом к клиенту должна быть строка-идентификатор того сервера, с которым этому клиенту следует начать взаимодействие. После последнего по протоколу запроса оракул переходит в состояние, в котором на любой запрос он возвращает один и тот же специальный символ (например, $\perp$), свидетельствующий о том, что сессия для него завершена.

Оракулы-участники имеют следующие специальные внутренние параметры, которые крайне важны для модели в целом:

— *sid*: идентификатор сессии. Перед началом взаимодействия этот параметр имеет специальное значение UNDEF, а после того, как оракул-участник успешно завершил взаимодействие по протоколу, становится равным конкатенации всех сообщений, переданных по каналу связи в рамках сессии;
— *pid*: идентификатор партнера. Перед началом взаимодействия этот параметр равен UNDEF. Он устанавливается равным строке-идентификатору того участника, с которым данный оракул-участник, по его мнению, осуществил успешное взаимодействие по протоколу и выработал общий ключ;
— *sk*: выработанный общий сеансовый ключ. Изначально также равен UNDEF и устанавливается равным ключу, выработанному в результате взаимодействия;
— *acc*: параметр, говорящий об успешности осуществленного оракулом взаимодействия. Изначально равен FALSE; устанавливается равным TRUE в тот момент, когда у оракула-участника становятся определены параметры *sid*, *pid* и *sk*;
— *term*: параметр, указывающий на то, что оракул завершил взаимодействие. Изначально равен FALSE.

Пример 1. Для иллюстрации рассмотрим в качестве AKE-протокола классический протокол Диффи — Хеллмана, в котором каждая из сторон снабжает посылаемое ей сообщение своим идентификатором. Рассмотрим последовательность запросов к оракулам-участникам и то, как меняются их внутренние состояния.

1) Запрос (B) к оракулу $\mathcal{O}_A^1$. Оракул выберет число x, вычислит $X = g^x$ и вернёт в качестве ответа на запрос пару (A, X). После этого он перейдёт в состояние ожидания ответа от сервера.
2) Запрос (A, X) к оракулу $\mathcal{O}_B^1$. Оракул $\mathcal{O}_B^1$ выберет y, вычислит $Y = g^y$ и общий ключ $sk = X^y$. Также он определит остальные внутренние параметры: $sid = (A, X)||(B, Y)$, $pid = A$, $acc = \text{TRUE}$ и $term = \text{TRUE}$. В качестве ответа на запрос оракул вернёт (B, Y).
3) Запрос (B, Y) к оракулу $\mathcal{O}_A^1$. Оракул вычислит общий ключ $sk = Y^x$, а также другие параметры: $sid = (A, X)||(B, Y)$, $pid = B$, $acc = \text{TRUE}$ и $term = \text{TRUE}$. В качестве ответа оракул вернёт пустую строку, свидетельствующую об успешном завершении взаимодействия.

Если запросом к оракулу $\mathcal{O}_B^1$ была бы не строка (A, X), а только идентификатор A, то оракул счёл бы это сообщение не соответствующим протоколу, вернул символ $\perp$, свидетельствующий об ошибке, и перешел в состояние завершённого взаимодействия. О том, что взаимодействие не завершилось успешно, свидетельствовали бы неустановленные (UNDEF) значения sid, pid, sk и то, что $acc = \text{FALSE}$.

Нестойкость этого протокола в модели BPR показана ниже в примере 2.

Оракулы, моделирующие возможности противника. Неформально, модель BPR предполагает, что противник полностью контролирует каналы связи между участниками сети, т. е. читает все передаваемые по ним сообщения, может изменять их, отправлять по каналам сообщения, сформированные им самостоятельно, и т. п. Он может также навязывать любому участнику сети выполнение протокола с любым другим участником сети. Помимо этого, он может получать сеансовые ключи некоторых выбранных им сессий (причины наличия такой возможности обсуждаются при описании соответствующего оракула). Цель противника — получить какую-либо информацию о сеансовом ключе выбранной им, но не вскрытой ранее с помощью упоминавшегося специального запроса сессии.

В модели BPR противник не может осуществлять запросы непосредственно к оракулам-участникам. Вместо этого ему предоставляется доступ к четырём специальным оракулам $\mathcal{O}_{\text{send}}$, $\mathcal{O}_{\text{exec}}$, $\mathcal{O}_{\text{rev}}$ и $\mathcal{O}_{\text{test}}$. Первые два моделируют возможности противника по влиянию на каналы связи. Третий оракул даёт возможность противнику получать выработанные участниками сеансовые ключи. Эта возможность не является естественной с точки зрения практики, поэтому мы подробно рассмотрим её далее. Четвёртый оракул $\mathcal{O}_{\text{test}}$ формализует угрозу в модели BPR, т. е. цель действий противника. В данной работе не анализируются криптографические свойства протокола при вскрытии противником долговременного ключа, поэтому мы не включаем в описание оракул $\mathcal{O}_{\text{cor}}$.

Прежде чем переходить к подробному описанию каждого из перечисленных оракулов, необходимо отметить, что модель BPR может быть параллельной (concurrent) и непараллельной (non-concurrent). В первом случае для одного участника допускается существование двух и более параллельных сессий, во втором — нет.

Четыре оракула, к которым может обращаться противник, имеют возможность получать значения любых внутренних параметров acc, sid, pid, sk и $term$ оракулов-участников:

1) $\mathcal{O}_{\text{send}}$: этот оракул моделирует возможность противника активно вмешиваться в обмен сообщениями в канале связи. Запросом к нему является тройка (U, i, M). В ответ на запрос оракул вернёт противнику пятёрку $(resp, sid, pid, acc, term)$,

где $resp$ — ответ оракула-участника $\mathcal{O}_U^i$ на запрос M; $sid, pid, acc, term$ — значения параметров оракула $\mathcal{O}_U^i$ после ответа на запрос M.

2) $\mathcal{O}_{\text{exec}}$: этот оракул моделирует возможность противника прочитать сообщения, пересланные участниками друг другу в ходе сессии. Запросом к нему является четвёрка (A, i, B, j). Оракул формирует ответ на запрос, используя оракул $\mathcal{O}_{\text{send}}$ для реализации штатного протокольного взаимодействия участников A и B. Ответом на запрос будет последовательность ответов оракула $\mathcal{O}_{\text{send}}$ на запросы, сделанные к нему оракулом $\mathcal{O}_{\text{exec}}$.

   Для этого оракула есть некоторые исключительные ситуации. Во-первых, если хотя бы один из оракулов-участников ($\mathcal{O}_A^i$ или $\mathcal{O}_B^j$) не находится в исходном состоянии (т. е. к нему уже был осуществлён хотя бы один запрос), то оракул $\mathcal{O}_{\text{exec}}$ вернёт специальное значение, сигнализирующее о невозможности выполнить запрос. Во-вторых, такое значение может быть возвращено ещё и в случае, если используется BPR-модель непараллельного типа, а для A или B есть оракул-участник с начатой, но не завершённой сессией.

   Легко видеть, что работу оракула $\mathcal{O}_{\text{exec}}$ противник может реализовать и самостоятельно, ведь для этого нужен только оракул $\mathcal{O}_{\text{send}}$. Выделение отдельного оракула для пассивного прослушивания противником канала связи является основной особенностью модели BPR по сравнению с общей моделью для оценки стойкости AKE-протоколов. За счёт этого удаётся отделить действия противника, связанные с изменением передаваемых сообщений, от пассивного прослушивания. Это особенно важно для PAKE-протоколов, так как их основная задача в том, чтобы самой эффективной атакой был online-перебор, который реализуется именно активным вмешательством в канал связи.

3) $\mathcal{O}_{\text{rev}}$: этот оракул позволяет противнику получить сессионный ключ участника сети в выбранной сессии. Запросом к оракулу является пара (U, i), где U — идентификатор участника; i — номер сессии. Ответом на запрос будет параметр sk оракула $\mathcal{O}_U^i$ (даже если он равен UNDEF).

   Наличие у противника доступа к этому оракулу выглядит менее естественно, чем возможность делать запросы к $\mathcal{O}_{\text{send}}$ и $\mathcal{O}_{\text{exec}}$. Следуя мотивации из оригинальной работы [8], отметим, что этот оракул призван отразить одно из свойств безопасности AKE-протоколов, а именно: потеря сеансовых ключей, полученных в результате выполнения AKE-протокола, не должна привести к возможности вскрытия остальных (ещё не потерянных) сеансовых ключей. Потеря сеансовых ключей участников сети возможна по целому ряду причин: использование нестойких криптографических механизмов в последующих протоколах (которые работают уже на основе выработанного сеансового ключа) или утечка ключа по причине ослабления защиты после завершения сессии (например, ключ может быть извлечён с помощью атаки, описанной в [19]).

   Дополнительно отметим, что наделение противника возможностью узнавать сеансовые ключи позволяет в определённой мере достичь модульности анализа, т. е. изолировать анализ AKE-протокола от условий его использования: даже при неаккуратном дальнейшем использовании результатов его работы протокол сам по себе должен быть стойким.

4) $\mathcal{O}_{\text{test}}$: этот оракул не отражает какой-либо возможности противника, а нужен только для формализации угрозы, т. е. задачи противника по нарушению целевых свойств безопасности AKE-протоколов. Как и для $\mathcal{O}_{\text{rev}}$, запросом к этому оракулу является пара (U, i). Если для оракула-участника $\mathcal{O}_U^i$ параметр acc ра-

вен FALSE, то оракул $\mathcal{O}_{\text{test}}$ вернёт символ ошибки $\perp$. Если $acc = \text{TRUE}$, то для формирования ответа оракул сначала выбирает бит b в соответствии с равномерным распределением на множестве $\{0, 1\}$. Если $b = 1$, то оракул возвращает ключ sk оракула-участника $\mathcal{O}_U^i$; если $b = 0$, то — ключ, выбранный в соответствии с равномерным распределением на множестве всех сеансовых ключей.
Формулировка угрозы в виде задачи отличения некоторого параметра от случайной строки является очень распространённой в математической криптографии [20] и отражает идею о том, что противник не должен получить об этом параметре никакой информации.

Особо отметим, что в BPR-модели противник может делать к оракулу $\mathcal{O}_{\text{test}}$ лишь один запрос, поэтому эту модель в литературе обычно называют моделью с угрозой FtG-типа («Find-then-Guess»). Развитие этой модели с расширением угрозы до так называемого RoR-типа («Real-or-Random») описано в работе [21]. Основное отличие моделей в том, что RoR-тип допускает больше одного запроса к оракулу $\mathcal{O}_{\text{test}}$ и не содержит оракула $\mathcal{O}_{\text{rev}}$.

Модель BPR формализует задачу оценки стойкости для PAKE-протоколов в общем. Для конкретных протоколов модель может быть уточнена путём добавления оракулов, необходимых для анализа конкретного протокола. Например, это может быть случайный оракул, которым заменяется используемая в протоколе хэш-функция, т. е. оракулы-участники тоже обращаются к этому оракулу вместо вычисления хэш-функции. Особенности использования модели со случайным оракулом («random-oracle model» — ROM) подробно рассматриваются в [22].

Преимущество противника. Прежде чем определить преимущество противника в модели BPR, необходимо ввести понятия *партнёрства* (*Partnering*) и *доступности* (*Freshness*) для оракулов-участников. Комментарии по поводу смысла этих понятий даны после определения преимущества противника.

Оракулы-участники $\mathcal{O}_A^i$ и $\mathcal{O}_B^j$ называются *партнёрами*, если выполнены следующие условия:

1) $acc_A^i = acc_B^j = \text{TRUE}$;
2) $sid_A^i = sid_B^j$;
3) $pid_A^i = B$, $pid_B^j = A$;
4) $sk_A^i = sk_B^j$;
5) не существует другой пары оракулов-участников с таким же значением sid, как у $\mathcal{O}_A^i$ и $\mathcal{O}_B^j$, для каждого из которых выполнено равенство $acc = \text{TRUE}$.

Оракул-участник $\mathcal{O}_A^i$ называется *доступным*, если выполнены следующие условия:

1) противник не делал запрос (A, i) к оракулу $\mathcal{O}_{\text{rev}}$;
2) противник не делал запрос (B, j) к оракулу $\mathcal{O}_{\text{rev}}$, где $\mathcal{O}_B^j$ — партнёр оракула $\mathcal{O}_A^i$.

Определим преимущество противника в BPR-модели для некоторого протокола $\mathcal{P}$. Противник в этой модели имеет доступ к оракулам $\mathcal{O}_{\text{send}}$, $\mathcal{O}_{\text{exec}}$, $\mathcal{O}_{\text{rev}}$, $\mathcal{O}_{\text{test}}$ и, возможно, дополнительным оракулам, необходимым для анализа конкретного протокола. Задача противника состоит в распознавании бита b, выбираемого оракулом $\mathcal{O}_{\text{test}}$, поэтому в результате работы противник возвращает свою «догадку» о нём — бит b'. Событие $SUCC$ состоит в том, что $b' = b$ и запрос к оракулу $\mathcal{O}_{\text{test}}$ был сделан для доступного оракула-участника (причём доступность должна сохраняться на протяжении всей работы противника, а не только до запроса к $\mathcal{O}_{\text{test}}$). Преимущество противника $\mathcal{A}$ по отношению к протоколу $\mathcal{P}$ в BPR-модели (определяемое стандартно в соответствии с п. 1.1), следуя оригинальной работе, будем обозначать через $\mathbf{Adv}_{\mathcal{P}}^{AKE}(\mathcal{A})$.

Неформально, модель противника можно сравнить с правилами игры между противником и криптосистемой. Хорошая игра не должна допускать выигрышную для одной из сторон стратегию, согласно которой другая сторона вообще не принимает участие в игре. В данном случае понятие доступного оракула необходимо, чтобы исключить следующий сценарий действий противника, который позволяет ему выигрывать, не зная ничего о строении анализируемого протокола. В нём противник делает запрос к оракулу $\mathcal{O}_{\text{exec}}$, после которого два оракула-участника становятся партнёрами, разделяя общий сеансовый ключ. После этого противник делает запросы к оракулам $\mathcal{O}_{\text{test}}$ и $\mathcal{O}_{\text{rev}}$: первый запрос — для одного из этих оракулов-участников, а второй — для другого. Чтобы угадать бит b оракула $\mathcal{O}_{\text{test}}$, противнику достаточно просто сравнить полученные ключи.

Пример 2. Опишем противника $\mathcal{A}$, осуществляющего угрозу в модели BPR для протокола, описанного в примере 1. Он выполняет следующую последовательность действий:

1) случайно выбирает z и вычисляет $Z = g^z$;
2) делает запрос $(B, 1, (A, Z))$ к оракулу $\mathcal{O}_{\text{send}}$. В качестве ответа $\mathcal{A}$ получает сообщение (B, Y) и значения внутренних переменных $sid = (A, Z)||(B, Y)$, $pid = A$, $acc = term = \text{TRUE}$;
3) вычисляет $K = Y^z$;
4) делает запрос $(B, 1)$ к оракулу $\mathcal{O}_{\text{test}}$. Оракул возвращает ключ K';
5) если $K = K'$, то $\mathcal{A}$ возвращает в качестве результата 1, иначе 0.

Заметим, что $\mathbf{Adv}^{AKE}(\mathcal{A})$ для данного протокола отличается от 1 на величину вероятности случайно выбрать ключ K' равным g^{zy}. Также стоит отметить, что оракул $\mathcal{O}_B^1$ является доступным, так как запросов к оракулу $\mathcal{O}_{\text{rev}}$ противник не делал.

Данный пример нужен исключительно для того, чтобы проиллюстрировать возможности и цели противника в модели BPR. Очевидно, что рассмотренный протокол не является в этой модели стойким, так как не использует каких-либо механизмов аутентификации сторон.

1.3. Стойкость PAKE-протоколов

Рассмотрим пример того, как в BPR-модели реализуется онлайн-перебор паролей на примере упрощённого PAKE-протокола $\mathcal{P}$ без подтверждения ключа.

Пример 3. Протокол $\mathcal{P}$ устроен следующим образом: клиент и сервер хранят в секрете общий пароль pw, который является элементом несекретного подмножества PW мощности N группы $G = \langle g \rangle$ простого порядка. Клиент C направляет серверу сообщение $(C, X = pw \cdot g^x)$, а сервер S отвечает сообщением $(S, Y = pw \cdot g^y)$ (x, y выбираются из множества $\{1, \ldots, q-1\}$ случайно и равновероятно). Далее стороны вычисляют общий ключ $K = (X/pw)^y = (Y/pw)^x$.

Опишем противника $\mathcal{A}$, осуществляющего угрозу в модели BPR для описанного протокола. Противник выполняет следующие действия:

1) полагает $T = PW$, $i = 0$;
2) выбирает pw' из T и полагает $T = T \setminus \{pw'\}$;
3) делает запрос $(S, i, (C, pw' \cdot g))$ к оракулу $\mathcal{O}_{\text{send}}$, извлекая из его ответа параметр $resp$, который является элементом группы G;
4) делает запрос (S, i) к оракулу $\mathcal{O}_{\text{rev}}$, получая в ответ ключ $K \in G$;
5) если $K \neq resp/pw'$, то полагает $i = i + 1$ и возвращается на шаг 2;
6) делает запрос $(S, i+1, (C, pw' \cdot g))$ к оракулу $\mathcal{O}_{\text{send}}$, извлекая из его ответа $resp \in G$;

7) делает запрос $(S, i+1)$ к оракулу $\mathcal{O}_{\text{test}}$, получая в ответ ключ K;
8) заканчивает работу с результатом 1, если $K = resp/pw'$, и 0 в противном случае.

Переход противника $\mathcal{A}$ на шаг 6 происходит в том случае, если $pw' = pw$, где pw — пароль, разделяемый C и S. Таким образом, противник просто осуществляет полный перебор паролей. Такой перебор называется онлайн-перебором, так как он предполагает активное взаимодействие противника с участниками сети.

Если противник может делать $N+1$ запрос к $\mathcal{O}_{\text{send}}$, то $\mathbf{Adv}^{AKE}(\mathcal{A}) = 2(1-1/|G|)-1 = 1 - 2/|G|$. Отсюда следует, что

$$\mathbf{Insec}_{\mathcal{P}}^{AKE}(t, q_{\text{send}} = N+1, q_{\text{rev}} = N, q_{\text{exec}} = 0) \geqslant 1 - 2/|G|,$$

где вычислительные ресурсы t равны $c \cdot N$; c — некоторая небольшая константа. Здесь через $q_{\text{send}}, q_{\text{rev}}, q_{\text{exec}}$ обозначено количество запросов к соответствующим оракулам.

Понятно, что описанный противник может быть изменён так, чтобы работать в условиях, когда ему доступно лишь $n \leqslant N+1$ запросов к оракулу $\mathcal{O}_{\text{send}}$, и иметь вероятность успеха порядка n/N.

Для любого PAKE-протокола противник может осуществить угрозу отличения ключа так, как показано в примере 3. Стойкий PAKE-протокол — это такой протокол, для которого сценарий с онлайн-перебором является наиболее эффективной атакой, причём ни относительно существенное увеличение вычислительных ресурсов, ни количества запросов к любым оракулам, кроме $\mathcal{O}_{\text{send}}$, не приводят к появлению более эффективных методов реализации угрозы.

Рассмотрим некоторые существующие PAKE-протоколы с точки зрения особенностей их строения, применимости модели BPR для их анализа и оценок их стойкости.

По формату этапа аутентификации протоколы выработки общего ключа на основе пароля можно подразделить на два типа:

1) Выработанный сессионный ключ непосредственно используется в качестве секретного аргумента функции вычисления аутентификационной информации.
2) Из выработанной обеими сторонами величины с помощью одного алгоритма создаётся сессионный ключ, а с помощью другого алгоритма генерируется аутентификационная информация.

Для протоколов первого типа BPR-модель не является корректной, так как существует противник, реализующий угрозу независимо от анализируемого протокола: ключ, выданный оракулом $\mathcal{O}_{\text{test}}$, проверяется на подлинность путём подстановки в известную противнику функцию получения аутентификационной информации и сравнения полученного результата с тем, что передавался по каналу связи. Поэтому BPR-модель не подходит для оценки стойкости протоколов первого типа. Авторам удалось найти доказательство безопасности протоколов такого типа только для этапа генерации ключа.

Для протоколов второго типа определённая выше угроза корректна.

Рассмотрим оценки стойкости, полученные для двух связанных друг с другом протоколов второго типа: One-Encryption Key-Exchange (OEKE) и AuthA. Протокол AuthA предложен M. Bellare и P. Rogaway в 2000 г. в [2] и стандартизирован IEEE (P1363.2: Standard Specifications for Password-Based Public-Key Cryptographic Techniques). В 2003 г. в работе [3] E. Bresson, O. Chevassut и D. Pointcheval привели оценку стойкости упрощённой версии этого протокола, которую назвали OEKE, в BPR-модели параллельного типа (concurrent) со случайным оракулом и оракулом

идеального шифра. В этой работе получена также оценка стойкости протокола OAuth в модели с угрозой ложной аутентификации, не основанной на задаче различения.

В терминологии настоящей работы оценка стойкости протокола OEKE в модели BPR, полученная в [3], выглядит следующим образом:

$$\mathbf{Insec}^{AKE}_{OEKE}(t, q_s, q_p, q_h, q_e) \leqslant 3\,\frac{q_s}{N} + 8q_h \cdot \mathbf{Insec}^{CDH}_{G}(t') + \frac{(2q_e + 3q_s + 3q_p)^2}{q-1} + \frac{q_h^2 + 4q_s}{2^{l_1}},$$

где q_s, q_p, q_h, q_e — максимально допустимые количества запросов к оракулам $\mathcal{O}_{\text{send}}$, $\mathcal{O}_{\text{exec}}$, случайному оракулу и оракулу идеального шифра соответственно; q — мощность используемой для вычислений группы G; $t' \leqslant t + (q_s + q_p + q_e + 1)\tau_G$; τ_G — трудоёмкость возведения в степень в группе G; l_1 — размер выхода хэш-функции, используемой для вычисления аутентификационной информации; $\mathbf{Insec}^{CDH}_{G}(t')$ — вероятность успеха наиболее эффективного алгоритма решения задачи CDH (её определение приведено в п. 4) в группе G.

Рассмотрим оценку подробнее. Величины 2^{l_1} и q на практике обычно достаточно большие (не меньше 2^{256}). Величину числителей двух правых слагаемых определяют значения q_h и q_e. Действительно, на практике для вычисления хэш-функции и зашифрования противник просто вычисляет эти функции и не обращается ни к каким оракулам, т. е. эти параметры ограничены исключительно его вычислительными ресурсами. Поэтому количество q_p пассивно прослушанных сеансов взаимодействия участников протокола существенно меньше q_h и q_e. Тем более меньше этих параметров оказывается величина q_s, которая на практике ограничивается организационными мерами (например, использованием счётчиков неуспешных сессий). Таким образом, если вычислительные ресурсы противника существенно меньше $\sqrt{q}$ (для $q \geqslant 2^{256}$ это предположение на сегодняшний день представляется вполне правдоподобным), то два правых слагаемых пренебрежимо малы.

Величина $\mathbf{Insec}^{CDH}_{G}(t')$ для хорошо подобранных групп (например, групп точек эллиптических кривых, выбранных с учётом критериев из работы [23]) оценивается с учётом общих методов дискретного логарифмирования, т. е. $\mathbf{Insec}^{CDH}_{G}(t') \approx (t')^2/q$. Мощность q используемой подгруппы точек эллиптической кривой должна быть выбрана таким образом, чтобы слагаемое $8q_h \cdot \mathbf{Insec}^{CDH}_{G}(t')$ было пренебрежимо малым с учётом предполагаемых вычислительных ресурсов противника. Здесь важно то, что это слагаемое можно сделать пренебрежимо малым с помощью изменения численного параметра протокола, а не принципа его работы.

Таким образом, величину $\mathbf{Insec}^{AKE}_{OEKE}(t, q_s, q_p, q_e, q_h)$ определяет первое слагаемое, которое соответствует атаке онлайн-перебора из примера 3. Оценка показывает, что эта атака является единственно эффективной для данного протокола в указанных предположениях о вычислительных возможностях и параметрах протокола. Именно о PAKE-протоколах с оценками стойкости такого типа говорят, что они являются стойкими в BPR-модели.

В работе [3] в условиях BPR-модели определяется также угроза ложной аутентификации нераспознавательного типа. Через $\mathbf{Adv}^{c-auth}_{oeke}(\mathcal{A})$ обозначена вероятность того, что противнику удалось выдать себя за клиента, которым он не является, т. е. оракул-сервер пришёл в состояние $acc = \text{TRUE}$, но не существует оракула-клиента, для которого этот оракул-сервер является партнёром. Недостатком данной модели является то, что в ней оценивается возможность противника выдавать себя лишь за клиента, но не за сервер.

2. Описание протокола SESPAKE

Через V_n будем обозначать множество наборов длины n с элементами из поля GF(2). Далее все операции с точками эллиптической кривой проводятся в подгруппе $E = \langle P \rangle$ простого порядка q группы точек некоторой эллиптической кривой порядка m. Через 0_E обозначим нейтральный элемент группы E; для конечного множества A через $a \xleftarrow{\mathcal{U}} A$ будем обозначать операцию выбора элемента a из A в соответствии с равномерным распределением.

Для конкретной реализации протокола необходимо зафиксировать следующие параметры:

— l — произвольное натуральное число;
— доказуемо псевдослучайные точки $P, Q_1, \ldots, Q_l \in E$. Доказуемая псевдослучайность гарантирует, что кратность любой точки относительно любой другой неизвестна (подробнее об этом см. в [13]);
— строковые константы T_A и T_B, которые используются клиентами и серверами соответственно.

Через F далее будем обозначать функцию PBKDF2, определённую в [24], а через H_{256} — хэш-функцию ГОСТ Р 34.11-2012 [25] с длиной выхода равной 256 битам. Через HMAC обозначим алгоритм [26] с длиной выхода 256 бит.

Протокол предполагает взаимодействие двух участников: клиента и сервера. Участники сети обладают фиксированными идентификаторами, которые являются байтовыми строками некоторой фиксированной длины N (длина одна для всех участников протокола), множество всех идентификаторов обозначим ID ($ID = V_8^N$). Если A — участник протокола, то его идентификатор из ID будем обозначать так же A.

Клиент в схеме работы протокола обозначается через A и хранит пароль $PW \in V_8^k$.

Сервер в схеме работы протокола обозначается через B и хранит следующие параметры для каждого клиента, с которым может взаимодействовать:

— число $ind \in \{1, \ldots, l\}$;
— строку $salt \in V_{64}$;
— точку $Q_{PW} = F(PW, salt, 2000)Q_{ind}$.

В секрете должны храниться только параметры PW и Q_{PW}.

Схема работы протокола приведена в таблице. Для удобства дальнейших рассуждений будем обозначать SESPAKE_{KA} и называть *протоколом выработки общего ключа* ту часть протокола, которая предшествует первому вычислению участником A значения функции HMAC. Оставшуюся часть протокола будем называть *протоколом подтверждения ключа с аутентификацией* и обозначать SESPAKE_{KC}. Это деление наглядно продемонстрировано в таблице. Под протоколом SESPAKE будем понимать аутентифицированную выработку общего ключа по протоколу SESPAKE_{KA} с последующим его подтверждением в соответствии с протоколом SESPAKE_{KC}.

Описание протокола SESPAKE

Фиксированные открытые параметры: $l, P, Q_1, \ldots, Q_l, m, q$			
$A\ [PW]$		$B\ [Q_{PW}, ind, salt]$	
	$\xrightarrow{A}$		SESPAKE$_{KA}$
$Q_{PW}^A = F(PW, salt, 2000)Q_{ind}$	$\xleftarrow{ind,\ salt}$		
$z_A = 0$, $\alpha \xleftarrow{\mathcal{U}} \{1, \ldots, q-1\}$			
$u_1 = \alpha \cdot P - Q_{PW}^A$	$\xrightarrow{u_1}$	if $u_1 \notin E \Rightarrow$ FINISH	
		$Q_B = u_1 + Q_{PW}$	
		$z_B = 0$, $\beta \xleftarrow{\mathcal{U}} \{1, \ldots, q-1\}$	
		if $\frac{m}{q}Q_B = 0_E \Rightarrow Q_B = P$, $z_B = 1$	
		$src = \left(\frac{m}{q} \cdot \beta\right) Q_B$	
		$K_B = H_{256}(src)$	
if $u_2 \notin E \Rightarrow$ FINISH	$\xleftarrow{u_2}$	$u_2 = \beta \cdot P + Q_{PW}$	
$Q_A = u_2 - Q_{PW}^A$			
if $\frac{m}{q}Q_A = 0_E \Rightarrow Q_A = P$, $z_A = 1$			
$src = \left(\frac{m}{q} \cdot \alpha\right) Q_A$			
$K_A = H_{256}(src)$			
$tag_A = T_A\|\|A\|\|ind\|\|salt\|\|u_1\|\|u_2$ $M_A = \text{HMAC}_{K_A}(tag_A)$	$\xrightarrow{M_A}$	$tag = T_A\|\|A\|\|ind\|\|salt\|\|u_1\|\|u_2$ $M = \text{HMAC}_{K_B}(tag)$	SESPAKE$_{KC}$
		If $M \neq M_A$ or $z_B \neq 0 \Rightarrow$ FINISH	
$tag = T_B\|\|B\|\|ind\|\|salt\|\|u_1\|\|u_2$ $M = \text{HMAC}_{K_A}(tag)$	$\xleftarrow{M_B}$	$tag_B = T_B\|\|B\|\|ind\|\|salt\|\|u_1\|\|u_2$ $M_B = \text{HMAC}_{K_B}(tag_B)$	
If $M \neq M_B$ or $z_A \neq 0 \Rightarrow$ FINISH			

3. Модели противника для оценки стойкости SESPAKE

В данной работе оценка стойкости протокола SESPAKE проводится в два этапа. Сначала оценивается стойкость протокола SESPAKE$_{KA}$ в непараллельной модели BPR. Далее оценивается стойкость протокола SESPAKE в модели с угрозой ложной аутентификации распознавательного типа, описываемой ниже. Обоснование стойкости протокола SESPAKE существенно основывается на стойкости протокола SESPAKE$_{KA}$.

Уточним, что стойкость протокола SESPAKE$_{KA}$ оценивается в BPR-модели непараллельного типа с дополнительным случайным оракулом $\mathcal{O}_H$, которым заменяется хэш-функция H_{256}, используемая для вычисления ключей K_A и K_B.

Для протокола SESPAKE рассматривается модель противника с некоторым вариантом угрозы ложной аутентификации. Более точно, рассматривается угроза отличения строки M, полученной с помощью имеющегося у участника ключа, от строки M', полученной с помощью случайно выбранного ключа.

Опишем возможности противника $\mathcal{A}$ для протокола SESPAKE. Помимо возможностей, которые есть у противника, рассматривавшегося для протокола SESPAKE$_{KA}$, противник $\mathcal{A}$ может делать запросы к оракулам $\mathcal{O}_{\text{HMAC}}$, $\mathcal{O}_{\text{check}}$ и $\mathcal{O}_{\text{auth}}$. Через $IDS_{A,i}$ обозначим конкатенацию тех данных, которые были пересланы по каналу связи с точки зрения оракула $\mathcal{O}_A^i$ до этапа подтверждения ключа. Эти оракулы работают по следующим правилам:

— $\mathcal{O}_{\text{HMAC}}$: реализует семейство случайных отображений $\{G_K : V^* \to V_{256}\}_{K \in V_{256}}$, которые выбираются случайно, независимо и равновероятно перед началом работы; на запрос (K, T) оракул $\mathcal{O}_{\text{HMAC}}$ возвращает значение $G_K(T)$;
— $\mathcal{O}_{\text{check}}$: в качестве ответа на запрос (A, i) этот оракул возвращает значение $G_{K_{A,i}}(T_A||IDS_{A,i})$, где $K_{A,i}$ — сессионный ключ, выработанный оракулом $\mathcal{O}_A^i$;
— $\mathcal{O}_{\text{auth}}$: с помощью этого оракула определяется угроза. Оракул $\mathcal{O}_{\text{auth}}$ принимает на вход пару (A, i), случайно и равновероятно выбирает бит $b \in \{0, 1\}$ и в зависимости от b возвращает либо пару (M_A, M') (если $b = 0$), либо пару (M_A, M) (если $b = 1$). При этом $M' = G_{K'}(T_B||IDS_{A,i})$, где $K' \xleftarrow{\mathcal{U}} V_{256}$; $M_A = G_{K_{A,i}}(T_A||IDS_{A,i})$; $M = G_{K_{A,i}}(T_B||IDS_{A,i})$.

Как и в модели для протокола SESPAKE_{KA}, противник может делать запрос к оракулу $\mathcal{O}_{\text{auth}}$ только для тех оракулов, которые допускают тестирование. Оракулом, допускающим тестирование, называется такой оракул $\mathcal{O}_A^i$, по отношению к которому не было запросов к $\mathcal{O}_{\text{check}}$ и для которого противник не получил значение M тривиальным образом, то есть не сделал запрос к $\mathcal{O}_{\text{check}}$ для оракула, разделяющего с $\mathcal{O}_A^i$ общий ключ. Говоря неформально, в такой модели противник перед тем, как попытаться осуществить угрозу, предупреждает, что он будет осуществлять угрозу по отношению именно к этому участнику и сеансу. Подчеркнем, что противник выбирает «цель» не в начале, а в процессе работы.

В качестве результата противник возвращает значение $a \in \{0, 1\}$. Через $\text{SUCC}_{\text{auth}}$ обозначим событие, состоящее в том, что $a = b$, где b — значение, которое выбрал оракул $\mathcal{O}_{\text{auth}}$.

4. Сопутствующие задачи и соотношения между ними

Рассмотрим задачи, которые используются при обосновании стойкости протокола SESPAKE. Все задачи формулируются для подгруппы E простого порядка q группы точек эллиптической кривой порядка m с образующим элементом P. Через τ будем обозначать трудоёмкость вычисления кратной точки в этой подгруппе.

Преимущества противников во всех задачах определяются как вероятность того, что результат соответствующего задаче эксперимента будет равен 1 (обозначается $\mathbf{Exp}(\mathcal{A}) = 1$). Трудоёмкость решения задач определяется величиной **Insec**, задаваемой так, как описано в п. 1.1.

4.1. Определения

Задача CDH (вычислительная задача Диффи — Хеллмана) определяется для противника $\mathcal{A}$ экспериментом $\mathbf{Exp}^{\text{CDH}}$:

— противнику $\mathcal{A}$ передаются точки $X = xP$, $Y = yP$, где $x, y \xleftarrow{\mathcal{U}} \mathbb{Z}_q$;
— противник $\mathcal{A}$ возвращает точку Z;
— если $Z = xyP$, то в качестве результата $\mathbf{Exp}^{\text{CDH}}(\mathcal{A})$ данного эксперимента выдаётся 1, иначе 0.

Для данных $X = xP$ и $Y = yP$ через $\text{CDH}_P(X, Y)$ будем обозначать точку xyP. Там, где из контекста ясно, о каком P идёт речь, будем писать просто $\text{CDH}(X, Y)$. Заметим, что справедливы следующие соотношения:

— $\text{CDH}_P(P, Y) = Y$;
— $\text{CDH}(aX, Y) = a \cdot \text{CDH}(X, Y)$;
— $\text{CDH}(X + Y, Z) = \text{CDH}(X, Z) + \text{CDH}(Y, Z)$.

Задача SCDH [27] отличается от CDH тем, что противнику передаётся только точка X, а его задача состоит в том, чтобы вычислить $\mathrm{CDH}(X, X)$. Через $\mathrm{SCDH}(X)$, где $X = xP$, будем обозначать точку x^2P.

Задача QCCDH определяется для противника $\mathcal{A}$ экспериментом $\mathbf{Exp}^{\mathrm{QCCDH}}$:

— противнику передаются точки $X, Q \xleftarrow{\mathcal{U}} E$;
— противник возвращает точки Y, U и V;
— если $U = \mathrm{CDH}(X, Y)$ и $V = \mathrm{CDH}(X - Q, Y - Q)$, то в качестве результата возвращается 1, иначе 0.

Задача $\mathbf{QPCCDH}_{\mathcal{D}}$.
Пусть $\mathcal{D} \subset \mathbb{Z}_q$ — множество паролей. Эксперимент $\mathbf{Exp}^{\mathrm{QPCCDH}}(\mathcal{A})$, определяющий задачу $\mathrm{QPCCDH}_{\mathcal{D}}$, состоит из двух этапов, а противник $\mathcal{A}$, участвующий в нём, использует при работе на разных этапах две вероятностные ленты, заполнение первой из которых возвращается вместе со значением Y. Заполнение случайной ленты для первого этапа может быть задано перед началом эксперимента произвольным образом. Если оно не задано, то выбирается случайно и равновероятно.

Эксперимент $\mathbf{Exp}^{\mathrm{QPCCDH}}(\mathcal{A})$:

— противнику $\mathcal{A}$ передаются точки $Q, X \xleftarrow{\mathcal{U}} E$;
— противник возвращает $Y \in E$ и u — заполнение своей случайной ленты, которая использовалась при вычислениях на первом этапе;
— противнику передаётся пароль $r \xleftarrow{\mathcal{U}} \mathcal{D}$;
— противник возвращает $K \in E$;
— если $K = \mathrm{CDH}(X - rQ, Y - rQ)$, то в качестве результата возвращается 1, иначе 0.

Замечание 1. Противник, угадавший перед возвращением Y пароль r, гарантированно решает задачу, возвращая $Y = P + rQ$ и $K = X - rQ$.

Замечание 2. Возврат заполнения случайной ленты обеспечивает возможность «запускать» противника $\mathcal{A}$ так, чтобы результат его работы на первом этапе (точка Y) для одних и тех же входных данных не менялся. Фактически это означает, что, получив от $\mathcal{A}$ точку Y, можно многократно «запускать» $\mathcal{A}$ для работы только на втором этапе.

Задача $\mathbf{S\text{-}QPCCDH}_{\mathcal{D},s}$ отличается от задачи $\mathrm{QPCCDH}_{\mathcal{D}}$ тем, что вместо одного ключа K противник $\mathcal{A}$ может выдать в качестве ответа множество ключей $\mathcal{K}$ мощности $s \in \mathbb{N}$. В эксперименте $\mathbf{Exp}^{\text{S-QPCCDH}}(\mathcal{A})$ в качестве результата возвращается 1, если точка $\mathrm{CDH}_P(X - rQ, Y - rQ)$ есть в множестве $\mathcal{K}$.

Везде далее, если из контекста ясно, для каких параметров s и $\mathcal{D}$ рассматриваются описанные задачи, их названия приводятся без индексов.

4.2. Соотношения между задачами

Докажем утверждения, итоговая цель которых — оценить сверху нестойкость задачи S-QPCCDH через нестойкость задачи SCDH. Согласно [27], наиболее эффективным методом её решения является дискретное логарифмирование.

Теорема 1. Справедливо неравенство

$$\mathbf{Insec}^{\mathrm{QCCDH}}(t) \leqslant \mathbf{Insec}^{\mathrm{SCDH}}(t + 4\tau).$$

Доказательство. Пусть $\mathcal{A} \in \mathcal{A}(t)$ — противник, решающий задачу QCCDH с вероятностью успеха ε. Построим противника $\mathcal{A}'$, решающего задачу SCDH с помощью $\mathcal{A}$.

Точку, полученную противником $\mathcal{A}'$ в рамках эксперимента $\mathbf{Exp}^{\mathrm{SCDH}}$, обозначим через Q. Противник $\mathcal{A}'$ осуществляет следующие действия: $b \xleftarrow{U} \mathbb{Z}_q$; подаёт $\mathcal{A}$ на вход пару $(X = 2Q + bP, Q)$, получая в ответ тройку Y, U, V. Заметим, что пара (X, Q), сформированная таким образом, принимает каждое значение из $E \times E$ с одинаковой вероятностью.

Если $\mathcal{A}$ успешно решил задачу QCCDH, то

$$U = \mathrm{CDH}(2Q + bP, Y) = 2 \cdot \mathrm{CDH}(Q, Y) + bY,$$
$$V = \mathrm{CDH}(Q + bP, Y - Q) = \mathrm{CDH}(Q, Y) - \mathrm{SCDH}(Q) + bY - bQ.$$

Таким образом, точка

$$R = 1/2 \cdot U - V + b/2 \cdot Y - bQ$$

является решением задачи SCDH для точки Q с вероятностью ε. Поэтому в силу произвольности $\mathcal{A}$ получаем следующую цепочку неравенств:

$$\mathbf{Insec}^{\mathrm{QCCDH}}(t) = \mathbf{Adv}^{\mathrm{QCCDH}}(\mathcal{A}) = \mathbf{Adv}^{\mathrm{SCDH}}(\mathcal{A}') \leqslant \mathbf{Insec}^{\mathrm{SCDH}}(t + 4\tau).$$

Теорема 1 доказана. ■

Заметим, что в работе [9] доказательство аналогичного утверждения требует корректировки. Неточность заключается в том, что итоговая оценка не учитывает, что построенный в [9] противник не может решить задачу CDH при $b = 0$ или $b = 1$, а также то, что входные аргументы для $\mathcal{A}$ выбираются не в соответствии с равномерным распределением.

Приведём формулировку леммы из работы [9], которая понадобится для доказательства теоремы 2.

Лемма 1 [9]. Для конечных множеств X и Y пусть $A \subset X \times Y$ таково, что $\Pr[(x, y) \in A] \geqslant \varepsilon$. Пусть также для произвольного $\alpha < \varepsilon$

$$B = \{x \in X : \Pr_{y' \in Y}[(x, y') \in A] \geqslant \varepsilon - \alpha\}.$$

Тогда $\Pr[B] \geqslant \alpha$ и $\Pr[B|A] \geqslant \alpha/\varepsilon$.

Теорема 2. Пусть $|\mathcal{D}| = n$. Для t, такого, что

$$\frac{2}{n} \geqslant \mathbf{Insec}^{\mathrm{QPCCDH}_{\mathcal{D}}}(t) \geqslant \frac{1}{n} + \varepsilon$$

для некоторого $\varepsilon \leqslant 1/n$, справедливо неравенство

$$\mathbf{Insec}^{\mathrm{QCCDH}}(2t + 2\tau) \geqslant \frac{n\varepsilon^3}{64}.$$

Доказательство. Пусть $\mathcal{A} \in \mathcal{A}(t)$ — противник, решающий задачу $\mathrm{QPCCDH}_{\mathcal{D}}$ с вероятностью успеха $\dfrac{2}{n} \geqslant \mathbf{Adv}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) \geqslant \dfrac{1}{n} + \varepsilon$.

Пусть $Q, X \in E$ — случайно и независимо выбранные точки, для которых требуется решить задачу QCCDH. Будем использовать противника $\mathcal{A}$ для того, чтобы решить эту задачу.

Выберем случайно и равновероятно заполнение u случайной ленты для первого шага работы противника $\mathcal{A}$ (заметим, что достаточно рассматривать ленту конечной длины t). Выберем случайно $r \neq r' \in \mathcal{D}$ и вычислим $\gamma = r' - r$.

Вычислим $Q' = \frac{1}{\gamma}Q$ и $X' = X + rQ'$. Далее дважды используем $\mathcal{A}$, чтобы решить задачу QPCCDH для одинаковых входных параметров первого этапа и разных параметров второго этапа.

Подадим на вход противнику $\mathcal{A}$ пару Q', X', записав на его вероятностную ленту первого этапа значение u. Пусть Y' — то, что противник выдал в качестве результата первого этапа (заполнение случайной ленты первого этапа, которое возвратит $\mathcal{A}$, очевидно, равно u). В качестве входного параметра второго этапа передадим $\mathcal{A}$ число r'. Пусть K' — значение, которое $\mathcal{A}$ возвратит в качестве результата.

Снова используем $\mathcal{A}$, подав на вход первого этапа ту же пару Q', X' и записав на ленту первого этапа u. Противник $\mathcal{A}$ возвратит то же самое значение Y'. В качестве входного параметра второго этапа теперь передадим число r. Пусть K — то, что выдал $\mathcal{A}$ в качестве результата вычислений на втором этапе.

Если противник $\mathcal{A}$ правильно решил обе задачи, то $K' = \mathrm{CDH}(X' - r'Q', Y' - r'Q')$, а $K = \mathrm{CDH}(X' - rQ', Y' - rQ')$. Пусть $Y = Y' - rQ'$, тогда тройка Y, K, K' будет решением задачи QCCDH для входных параметров X, Q. Это объясняется тем, что

$$K = \mathrm{CDH}(\underbrace{X'}_{X+rQ'} - rQ', \underbrace{Y' - rQ'}_{Y}) = \mathrm{CDH}(X, Y),$$

$$K' = \mathrm{CDH}(X' - r'Q', Y' - r'Q') = \mathrm{CDH}(X + rQ' - r'Q', Y + rQ' - r'Q) =$$
$$= \mathrm{CDH}(X - (r' - r)Q', Y - (r' - r)Q') = \mathrm{CDH}(X - Q, Y - Q),$$

так как $Q' = \dfrac{1}{r' - r}Q$.

Вероятность решить задачу QCCDH для случайной пары X, Q у противника, действующего описанным образом, не меньше, чем вероятность того, что противник $\mathcal{A}$ решит одновременно задачи для входов X', Q', r и X', Q', r', где X', Q', r, r' случайны и независимы, при условии, что заполнение случайной ленты u первого этапа для обеих задач одинаково. Обозначим это событие через SUCC.

Множество $\Omega_0 = \{(X', Q', u, v, r) : X', Q' \in E,\ u, v \in \{0,1\}^t,\ r \in \mathcal{D}\}$ является пространством элементарных событий случайной величины, которой является результат эксперимента $\mathbf{Exp}^{\mathrm{QPCCDH}}$. По условию теоремы $\Pr_{\Omega_0}\{\mathbf{Exp}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) = 1\} \geqslant \dfrac{1}{n} + \varepsilon$.

Представим Ω_0 в виде декартова произведения $\Omega_1' \times \Omega_1$:

$$\Omega_1' = \{u : u \in \{0,1\}^t\},$$
$$\Omega_1 = \{(X', Q', v, r) : X', Q' \in E,\ v \in \{0,1\}^t,\ r \in \mathcal{D}\}.$$

Пусть $S_1 = \left\{u \in \Omega_1' : \Pr_{\Omega_1}[\mathbf{Exp}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) = 1] \geqslant \dfrac{1}{n} + \dfrac{\varepsilon}{2}\right\}$. Тогда из леммы 1 следует, что

$$\Pr_{\Omega_0}[u \in S_1 : \mathbf{Exp}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) = 1] \geqslant \frac{\varepsilon/2}{1/n + \varepsilon} = \frac{n\varepsilon}{2 + 2n\varepsilon}.$$

Теперь представим Ω_1 в виде декартова произведения $\Omega_2' \times \Omega_2$:

$$\Omega_2' = \{(X', Q') : X', Q' \in E\}, \quad \Omega_2 = \{(v, r) : v \in \{0,1\}^t,\ r \in \mathcal{D}\}.$$

Для любого $u \in S_1$ пусть

$$S_2(u) = \left\{(X', Q') \in E \times E : \Pr_{\Omega_2}[\mathbf{Exp}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) = 1] \geqslant \frac{1}{n} + \frac{\varepsilon}{4}\right\},$$

тогда по лемме 1 справедливо неравенство

$$\Pr_{\Omega_1}[(X',Q') \in S_2(u) : \mathbf{Exp}^{\mathrm{QPCCDH}_{\mathcal{D}}}(\mathcal{A}) = 1] \geqslant \frac{\varepsilon/4}{1/n + \varepsilon/2} = \frac{n\varepsilon}{4 + 2n\varepsilon}.$$

Вероятность успеха противника $\mathcal{A}$ при решении задачи для случайного входа X', Q', r не меньше чем $\dfrac{1}{n} + \varepsilon$, причём вероятность того, что выполнены условия $u \in S_1$, $(X', Q') \in S_2(u)$, не меньше чем

$$\frac{n\varepsilon}{2 + 2n\varepsilon} \frac{n\varepsilon}{4 + 2n\varepsilon} \geqslant \frac{1}{8} \left(\frac{n\varepsilon}{1 + n\varepsilon} \right)^2.$$

Противник $\mathcal{A}$ решает задачу для входа X', Q', r' при фиксированных $u \in S_1$, $(X', Q') \in$ $\in S_2(u)$ с вероятностью не меньше чем $\dfrac{1}{n} + \dfrac{\varepsilon}{4} \geqslant \dfrac{\varepsilon}{4}$. Следовательно, выполняется следующее неравенство:

$$\Pr[\mathrm{SUCC}] \geqslant \left(\frac{1}{n} + \varepsilon \right) \frac{1}{8} \left(\frac{n\varepsilon}{1 + n\varepsilon} \right)^2 \frac{\varepsilon}{4}.$$

Из условия $\dfrac{2}{n} \geqslant \dfrac{1}{n} + \varepsilon$ получаем требуемое неравенство:

$$\mathbf{Insec}^{\mathrm{QCCDH}}(2t + 2\tau) \geqslant \Pr[\mathrm{SUCC}] \geqslant \frac{n\varepsilon^3}{64}.$$

Теорема 2 доказана. ■

Теорема 3. Справедливо неравенство

$$\mathbf{Insec}^{\text{S-QPCCDH}_{s,\mathcal{D}}}(t) \leqslant \frac{1}{|\mathcal{D}|} + \sqrt[6]{\frac{\mathbf{Insec}^{\mathrm{SCDH}}(4t + \Theta + O(s\tau))}{|\mathcal{D}|^2} + \frac{2^{13}s^4}{|\mathcal{D}|^2 q}},$$

где Θ — трудоёмкость поиска пары одинаковых элементов в двух множествах размера s^2.

Доказательство. Пусть $\mathcal{A} \in \mathcal{A}(t)$ — противник, решающий задачу $\text{S-QPCCDH}_{s,\mathcal{D}}$ с вероятностью успеха $\dfrac{1}{n} + \varepsilon$, где $n = |\mathcal{D}|$; $R \in E$ — точка, для которой требуется решить задачу SCDH. Будем использовать для решения противника $\mathcal{A}$.

Выберем случайно и независимо друг от друга числа $x_1, x_2, q_1, q_2 \in \mathbb{F}_p$. Вычислим точки $X_i = 2R + x_i P$ и $Q_i = R + q_i P$. Выберем случайно и независимо друг от друга два заполнения u_1 и u_2 случайной ленты первого этапа противника $\mathcal{A}$. Далее используем противника $\mathcal{A}$ в качестве чёрного ящика таким же образом, как описано в доказательстве теоремы 2, сначала для пары X_1, Q_1 и заполнения u_1, а потом для пары X_2, Q_2 и заполнения u_2. С учётом порядка построения противника в теореме 2, в последующих выкладках используем следующие дополнительные обозначения: r_i, r_i', $Q_i' = \dfrac{1}{r_i' - r_i} Q_i$, $X_i' = X_i + r_i Q_i'$ и Y_i. Два этих эксперимента независимы в силу независимости выбора параметров x_i, q_i и u_i. В результате такой процедуры получим четыре множества по s точек в каждом.

Вероятность того, что в обоих множествах, полученных в результате проведения одной процедуры, найдутся решения поставленной перед противником $\mathcal{A}$ задачи S-QPCCDH$_{s,\mathcal{D}}$, оценивается снизу величиной $n\varepsilon^3/64$ (рассуждения совпадают с теми, что приведены в доказательстве теоремы 2). Поскольку при проведении процедур параметры выбираются независимо, вероятность того, что в каждом из четырёх полученных множеств найдётся правильное решение, не меньше $(n\varepsilon^3/64)^2$. Полученные множества обозначим через S_1, S_1', S_2 и S_2'.

Если противник $\mathcal{A}$ успешно решил обе задачи при выполнении одной из процедур, то в множествах S_i и S_i' присутствуют точки K_i и K_i', для которых выполнены следующие соотношения:

$$K_i = \mathrm{CDH}(X_i' - r_iQ_i', \underbrace{Y_i - r_iQ_i'}_{Y_i'}) = \mathrm{CDH}(X_i, Y_i') = 2\mathrm{CDH}(R, Y_i') + x_iY_i',$$

$$\begin{gathered} K_i' = \mathrm{CDH}(X_i' - r_i'Q_i', Y_i - r_i'Q_i') = \mathrm{CDH}(X_i - (r_i' - r_i)Q_i', Y_i' - (r_i' - r_i)Q_i') = \\ = \mathrm{CDH}(X_i - Q_i, Y_i' - Q_i) = \mathrm{CDH}(R + x_iP - q_iP, Y_i' - R - q_iP) = \\ = \mathrm{CDH}(R, Y_i') - \mathrm{SCDH}(R) - q_iR + x_iY_i' - x_iR - x_iq_iP - q_iY_i' + qR + q^2P. \end{gathered}$$

Отсюда заключаем, что

$$\mathrm{SCDH}(R) = 1/2 \cdot K_i - K_i' + \underbrace{q_iR - x_i/2 \cdot Y_i' + x_iR + x_iq_iP + q_iY_i' - qR - q^2P}_{C}. \qquad (1)$$

Точка C вычисляется один раз для каждой из процедур. Для каждого из четырёх множеств вычисляется s точек $1/2 \cdot K_i$ или K_i' (в зависимости от рассматриваемого множества). После этого с использованием соотношения 1 формируются два множества по s^2 элементов в каждом. Первый элемент из первого такого множества, для которого нашёлся совпадающий с ним элемент во втором множестве, выдаётся в качестве предполагаемого значения $\mathrm{SCDH}(Q)$. Вероятность случайного совпадения оценим сверху величиной $2s^4/q$.

Таким образом, алгоритм решает задачу SCDH с вероятностью успеха не меньше чем $\dfrac{n^2\varepsilon^6}{2^{12}} - \dfrac{2s^4}{q}$. Поэтому

$$\mathbf{Insec}^{\mathrm{SCDH}}(4t + \Theta + O(s\tau)) \geqslant \frac{n^2\varepsilon^6}{2^{12}} - \frac{2s^4}{q}.$$

Отсюда получаем

$$\mathbf{Insec}^{\text{S-QPCCDH}_{s,\mathcal{D}}}(t) \leqslant \frac{1}{|\mathcal{D}|} + \sqrt[6]{\frac{\mathbf{Insec}^{\mathrm{SCDH}}(4t + \Theta + O(s\tau))}{|\mathcal{D}|^2} + \frac{2^{13}s^4}{|\mathcal{D}|^2q}}.$$

Теорема 3 доказана. ■

Отметим, что в оригинальной работе [9] доказательство схожей теоремы также требует корректировки.

5. Оценка стойкости SESPAKE$_{KA}$

Докажем нижнюю оценку преобладания в задаче отличения ключа от случайной строки для протокола SESPAKE$_{KA}$.

Теорема 4. При $q_{\text{send}} \geqslant 2$ и некотором t, достаточном для осуществления одним клиентом одного взаимодействия в соответствии с протоколом, справедливо неравенство

$$\mathbf{Insec}^{AKE}_{\text{SESPAKE}_{KA}}(t, q_{\text{send}}) \geqslant \frac{1}{|\mathcal{D}|} - \frac{1}{q}.$$

Доказательство. Приведём пример противника, решающего задачу отличения ключа от случайной строки для протокола SESPAKE_{KA}. Противник с помощью $\mathcal{O}_{\text{send}}$ посылает некоторый A_{ID} участнику B, получая в ответ $(ind, salt)$. Противник выбирает пароль PW' из словаря $\mathcal{D}$ в соответствии с равномерным распределением и вычисляет $Q_{PW'}$, а также элемент α из множества $\{1, \ldots, q-1\}$ в соответствии с равномерным распределением. После этого он вычисляет u_1 в соответствии со спецификацией протокола (как если бы он был пользователем A) и с помощью запроса к $\mathcal{O}_{\text{send}}$ посылает u_1 участнику B. Получив его ответ, противник вычисляет ключ сеанса k' в соответствии со спецификацией протокола, используя PW' в качестве пароля.

После этого противник делает запрос к оракулу $\mathcal{O}_{\text{test}}$ на отличение того ключа, который был только что установлен участником B. Получая некоторую строку k, противник сравнивает её с k', полагает $b' = 1$, если они равны, и $b' = 0$ в противном случае и возвращает b' в качестве результата.

Если k является ключом, установленным участником B в ходе описанного взаимодействия с противником, то $\Pr[k' = k] \geqslant \dfrac{1}{|\mathcal{D}|}$ (не учитываем вероятность коллизии хэш-функции при выработке ключа и функции F при выработке $Q_{PW'}$). Пусть b — случайный бит, который выбрал оракул $\mathcal{O}_{\text{test}}$. Справедливо соотношение

$$\Pr[b' = b] = \Pr[b' = 0 | b = 0] \Pr[b = 0] + \Pr[b' = 1 | b = 1] \Pr[b = 1] =$$
$$= \frac{1}{2}\left((1 - 1/q) + \Pr[b = 1 | b' = 1]\right) \geqslant \frac{1}{2}\left((1 - 1/q) + \frac{1}{|\mathcal{D}|}\right) = \frac{1}{2} + \frac{1}{2|\mathcal{D}|} - \frac{1}{2q}.$$

Таким образом, $\mathbf{Insec}^{AKE}_{\text{SESPAKE}_{KA}}(t, q_{\text{send}}) \geqslant \dfrac{1}{|\mathcal{D}|} - \dfrac{1}{q}$. ■

Учитывая, что $|\mathcal{D}| \ll q$, преобладание описанного противника по порядку соответствует $\dfrac{1}{|\mathcal{D}|}$.

Доказательство стойкости (с некоторыми параметрами) протокола SESPAKE_{KA} проводится в модели со случайным оракулом $\mathcal{O}_H$ (используемую в протоколе хэш-функцию H считаем случайной функцией). Стойкость в конечном итоге базируется на сложности вычислительной задачи Диффи — Хеллмана (CDH) в соответствующей группе.

Замечание 3. Приведём комментарии по поводу случайного оракула, которые призваны облегчить понимание некоторых шагов доказательств последующих утверждений. Термин «случайный оракул» широко используется в работах по математической криптографии. В нашем случае моделирование некоторой функции $H : A \to B$ с помощью случайного оракула означает, что в рассуждениях функцию H реализует некоторый вычислитель $\mathcal{O}_H$, который принимает на вход элементы множества A, возвращая значения из множества B (этот вычислитель принято называть оракулом). «Случайность» оракула $\mathcal{O}_H$ заключается в том, каким образом он выбирает значение из B, которое возвратит в ответ на значение-запрос из A. При первом запросе оракул $\mathcal{O}_H$ случайно и равновероятно выбирает функцию из множества всех функций,

определённых на A и принимающих значения из B, после чего возвращает значение выбранной функции, вычисленное для поданного ему на вход аргумента. Для всех последующих запросов оракул $\mathcal{O}_H$ просто возвращает соответствующие значения выбранной функции.

Приведённое определение поведения оракула $\mathcal{O}_H$ удобнее интерпретировать следующим эквивалентным образом. Оракул хранит таблицу T размера $|A|$, индексируемую элементами множества A, все значения в которой перед первым запросом не определены. При поступлении на вход значения $a \in A$ оракул $\mathcal{O}_H$ либо возвращает значение $T(a)$, если оно определено, либо выбирает случайно и равновероятно значение $b \in B$, устанавливает $T(a) = b$ и возвращает b в качестве ответа. Таким образом, оракул фактически определяет реализуемую им функцию не единовременно, а в процессе ответов на запросы.

Рассмотрим задачу распознавания двух сценариев в следующем эксперименте. Пусть $\mathcal{O}_1$ и $\mathcal{O}_2$ — случайные оракулы, реализующие отображения $A \to B$. Противник может делать запросы вида (i, a), $i \in \{1, 2\}$, $a \in A$, к оракулу $\mathcal{O}$, который возвращает элемент множества B, используя при этом оракулов $\mathcal{O}_1$ и $\mathcal{O}_2$. Перед началом работы оракул $\mathcal{O}$ случайно и равновероятно выбирает значение $\beta \in \{1, 2\}$. Оракул $\mathcal{O}$ хранит таблицу T, которая изначально пуста. При поступлении на вход запроса (i, a) он действует следующим образом:

— если $\beta = 1$, то $\mathcal{O}(i, a) = \mathcal{O}_1(a)$;
— если $\beta = 2$, то если $T(a)$ не определено, то $\mathcal{O}(i, a) = \mathcal{O}_i(a)$ и $T(a) = \mathcal{O}_i(a)$, иначе $\mathcal{O}(i, a) = T(a)$.

Перед противником стоит задача различения двух сценариев: $\beta = 1$ или $\beta = 2$.

Такое подробное описание приведено ради строгости. Неформально, противник должен определить, получает ли он ответы от двух случайных оракулов или от одного, причём он не имеет возможности делать запросы с одинаковым вторым аргументом a к двум разным оракулам (чтобы сравнить реализуемые оракулами функции покоординатно).

Ясно, что при каждом значении β противник фактически осуществляет запросы к одному случайному оракулу (во втором случае этот оракул просто определяется несколько нестандартно). Таким образом, он не может получить никакой информации о значении β. Преобладание противника при решении такой задачи равно нулю независимо от вычислительных ресурсов.

Далее q_{exec}, q_{send}, q_H — количество запросов к оракулам $\mathcal{O}_{\text{exec}}$, $\mathcal{O}_{\text{send}}$, $\mathcal{O}_H$ соответственно, которые совершает противник $\mathcal{A}_{\text{SESPAKE}_{KA}}$. В теореме 5 значения этих параметров могут быть произвольными. На практике ограничение их значений является важнейшим условием для обеспечения стойкости протокола. Подробные комментарии даны в п. 7.1.

Теорема 5. Справедливо неравенство

$$\mathbf{Insec}^{AKE}_{\text{SESPAKE}_{KA}}(t, q_H, q_{\text{send}}, q_{\text{exec}}, q_{\text{rev}}) \leqslant$$

$$\leqslant \frac{(2q_{\text{exec}} + q_{\text{send}})^2}{q} + 2q_H \mathbf{Adv}^{\text{CDH}}(t + 2\tau q_{\text{exec}}) + 2q_{\text{send}} \mathbf{Adv}^{\text{S-QPCCDH}_{\mathcal{D},2q_H}}(t + q_{S1}\tau + C),$$

где C — некоторая константа; τ — трудоёмкость вычисления кратной точки в группе E; q_{S1} — количество запросов к оракулу $\mathcal{O}_{\text{send}}$ вида $(ind, salt)$.

Структура доказательства. Пусть $\mathcal{A}_{\text{SESPAKE}_{KA}}$ — противник, для которого $\Pr[\text{SUCC}]$ — вероятность успеха в процедуре, описанной в конце п. 3. Будем исполь-

зовать противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в качестве чёрного ящика, перехватывая его запросы к оракулам $\mathcal{O}_{\text{exec}}$, $\mathcal{O}_{\text{send}}$, $\mathcal{O}_{\text{rev}}$ и $\mathcal{O}_{\text{test}}$, моделируя работу участников в соответствии с протоколом SESPAKE_{KA}. Далее постепенно будем «лишать» этого противника возможности получать истинные ответы оракулов. При этом будем показывать, насколько изменяются от шага к шагу вероятности успеха противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в модифицированных экспериментах. После некоторого количества модификаций получим эксперимент, вероятность успеха в котором равна $1/2$. Таким образом, отклонение вероятности успеха в «чистом» эксперименте от $1/2$ можно будет оценить сверху суммой оценок, полученных при каждой модификации эксперимента.

Опишем качественную структуру получаемой оценки. Модифицируя условия эксперимента, мы фактически лишаем противников определённых возможностей и тем самым исключаем из рассмотрения некоторый класс противников, которые не могут быть успешны в новых условиях (т. е. тех, которые существенно используют заблокированные возможности). Так, класс, который выводится из рассмотрения при рандомизации ответов оракула $\mathcal{O}_{\text{exec}}$, состоит из противников, пассивно прослушивающих канал связи. Основные рассуждения при этом направлены на оценку преобладания «исключаемых» противников. Далее полученная оценка включается в итоговую в качестве одного из слагаемых. Итого, если всё множество A противников подходящим образом разбито, например, на классы B_1, B_2 и B_3, то оценка имеет следующий вид:

$$\mathbf{Adv}(A) = \max\{\mathbf{Adv}(B_1), \mathbf{Adv}(B_2), \mathbf{Adv}(B_3)\} \leqslant C_1 + C_2 + C_3,$$

где C_i — верхняя оценка для $\mathbf{Adv}(B_i)$.

Доказательство. Пусть вероятность успеха противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ равна $\Pr[\text{SUCC}_0]$. Будем использовать противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в качестве чёрного ящика, перехватывая его запросы к оракулам, при этом нас интересует вероятность успеха противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в модифицированном эксперименте. Для обработки его запросов к оракулам будем моделировать подразумеваемое поведение всех возможных участников протокола: случайным образом выбирать для каждого пароль, генерировать случайные точки на этапе выработки ключа и т. п. В этом случае смоделированное множество участников с точки зрения противника не отличается от того, которое используется в «чистом» эксперименте. Поэтому вероятность $\Pr[\text{SUCC}_1]$ равна вероятности $\Pr[\text{SUCC}_0]$.

Далее используем противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в качестве чёрного ящика в эксперименте, отличающемся от предыдущего эксперимента с полноценно смоделированной сетью абонентов. Отличие заключается в том, что будем заканчивать всякую работу и выдавать на выход случайный бит в том случае, если среди точек $\{u_1\}$ и $\{u_2\}$, полученных в результате выполнения противником $\mathcal{A}_{\text{SESPAKE}_{KA}}$ запросов к оракулам $\mathcal{O}_{\text{send}}$ и $\mathcal{O}_{\text{exec}}$ (смоделированных нами), появились две одинаковые точки. Вероятность такого события не превышает $(2q_{\text{exec}} + q_{\text{send}})^2/(2q)$. Поэтому вероятность успеха $\Pr[\text{SUCC}_2]$ в этом эксперименте отличается от $\Pr[\text{SUCC}_1]$ не более чем на $(2q_{\text{exec}} + q_{\text{send}})^2/(2q)$. Данный эксперимент призван освободить дальнейшие рассуждения от противников, которые, например, посылают запросы с одинаковым u_1 участнику B, восстанавливая потом (с помощью $\mathcal{O}_{\text{rev}}$) установленный им ключ K_B, до тех пор, пока не появится ранее передававшийся u_2. Если такое событие случается, то ключ для K_A следует установить равным тому K_B, который соответствует ранее встреченному u_2.

В следующей модификации эксперимента (назовём её $\mathbf{Exp}_3$) будем при запросе, поступившем от $\mathcal{A}_{\text{SESPAKE}_{KA}}$ к оракулу $\mathcal{O}_{\text{exec}}$, в качестве сессионного ключа выдавать значение произвольной неизвестной и недоступной для вычисления противнику

$\mathcal{A}_{\text{SESPAKE}_{KA}}$ случайной функции $H'(u_1, u_2)$. Пусть вероятность успеха в этом эксперименте отличается от вероятности успеха в предыдущем эксперименте на $\triangle P$, то есть $\triangle P = |\Pr[\text{SUCC}_2] - \Pr[\text{SUCC}_1]|$. Поскольку мы считаем, что функция H, с помощью которой формируется реальный ключ, является случайной, то различия в поведении противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в этом и предыдущем экспериментах могут наблюдаться лишь в том случае, если он осуществил запрос $\dfrac{m}{q}\,\text{CDH}\,(u_1 + Q_{PW}, u_2 - Q_{PW})$ к оракулу $\mathcal{O}_H$. То есть с вероятностью ΔP среди q_H запросов к оракулу $\mathcal{O}_H$ фактически присутствует решение задачи CDH для пары $(u_1 + Q_{PW}, u_2 - Q_{PW})$. Этим можно воспользоваться, чтобы решить задачу CDH для произвольных точек Q_1 и Q_2. Для этого в ответ на запрос к оракулу $\mathcal{O}_{\text{exec}}$ будем в качестве u_1 и u_2 использовать точки $(Q_1 + \alpha P) - Q_{PW}$ и $(Q_2 + \beta P) + Q_{PW}$. Если в запросах к оракулу $\mathcal{O}_H$ (их всего q_H), которые осуществлял противник, есть значение $\text{CDH}(u_1 + Q_{PW}, u_2 - Q_{PW})$, то

$$\text{CDH}(Q_1, Q_2) = \text{CDH}(u_1 + Q_{PW}, u_2 - Q_{PW}) - \beta Q_1 - \alpha Q_2 - \alpha\beta P.$$

Из этого можно сделать вывод, что ΔP не превосходит значения $q_H \cdot \mathbf{Insec}^{\text{CDH}}(t + {} + 2\tau q_{\text{exec}} + 3\tau)$. Это означает, что для пассивно прослушивающего канал связи противника задача отличения сессионных ключей от случайных данных не легче задачи CDH.

На следующем шаге эксперимент $\mathbf{Exp}_3$ модифицируется путём рандомизации ответов оракула $\mathcal{O}_{\text{send}}$: на запросы, подразумевающие содержательные с точки зрения криптографии ответы, будем возвращать случайные точки и формировать сессионный ключ с помощью функции H' (новый эксперимент назовём $\mathbf{Exp}_4$). Таким образом, в новом эксперименте противник не получает никакой информации от запросов к оракулам. Поэтому вероятность успеха $\Pr[\text{SUCC}_4]$ в таком эксперименте равна $1/2$.

Лемма 2. Справедливо неравенство

$$\left|\Pr[\text{SUCC}_3] - \underbrace{\Pr[\text{SUCC}_4]}_{=1/2}\right| \leqslant q_{\text{send}} \cdot \mathbf{Insec}^{\text{S-QPCCDH}_{|\mathcal{D}|,2q_H}}(t + q_{S1}\tau + C).$$

Доказательство. Чтобы оценить разницу между вероятностями успехов в $\mathbf{Exp}_3$ и $\mathbf{Exp}_4$, построим специальную цепочку из $q_u = q_{u_1} + q_{u_2} + 1$ (q_{u_1}, q_{u_2} — количества запросов к $\mathcal{O}_{\text{send}}$, аргументами которых являются точки u_1 и u_2 соответственно) экспериментов $\mathbf{Hyb}_0, \ldots, \mathbf{Hyb}_{q_{u_1}+q_{u_2}}$, каждый из которых будет чуть больше отличаться от $\mathbf{Exp}_3$ и будет чуть больше похож на $\mathbf{Exp}_4$:

$$\mathbf{Exp}_3 = \mathbf{Hyb}_0 \rightsquigarrow \mathbf{Hyb}_1 \rightsquigarrow \cdots \rightsquigarrow \mathbf{Hyb}_{q_{u_1}+q_{u_2}} = \mathbf{Exp}_4. \tag{2}$$

Далее в экспериментах $\mathbf{Hyb}_j$ будем через i обозначать номер того запроса к оракулу $\mathcal{O}_{\text{send}}$, параметром которого является либо u_1, либо u_2 (то есть этот номер учитывает только такие запросы).

Описание эксперимента $\boldsymbol{Hyb}_j$*:* запросы ко всем оракулам, кроме $\mathcal{O}_{\text{send}}$, обрабатываются так же, как в $\mathbf{Exp}_3$. Запросы к оракулу $\mathcal{O}_{\text{send}}$ обрабатываются следующим образом:

— запрос с номером $1 \leqslant i \leqslant j$ к оракулу $\mathcal{O}_{\text{send}}$ обрабатываем, как в $\mathbf{Exp}_4$;
— запрос с номером $i > j$ к оракулу $\mathcal{O}_{\text{send}}$ обрабатываем, как в $\mathbf{Exp}_3$.

Если P_j — вероятность успеха противника $\mathcal{A}_{\text{SESPAKE}_{KA}}$ в эксперименте $\mathbf{Hyb}_j$, то

$$\left|\Pr[\text{SUCC}_3] - \Pr[\text{SUCC}_4]\right| \leqslant \sum_{j=1}^{q_u} |P_j - P_{j-1}|.$$

Опишем набор противников D_j и D'_j, $j = 1, \ldots, q_u$, с помощью которых оценим разность $|P_j - P_{j-1}|$. Будем использовать этих противников для решения задачи S-QPCCDH с некоторыми модификациями, а именно: при описании D_j будем считать, что для полученных на вход точек W и Q_s известны их кратности относительно P: $W = wP$ и $Q_s = zP$. Для D'_j такая модификация не рассматривается (в описании этого алгоритма w и z не участвуют).

Противник D_j

Через i будем обозначать номер следующего запроса к оракулу $\mathcal{O}_{\text{send}}$ с параметром u_1 (пересылка от A к B, в ответ на которую при правильном состоянии выдаётся точка u_2) или с параметром u_2 (пересылка от B к A, после которой при правильном состоянии A формирует ключ K_A). Эти запросы будем обозначать $S2$ и $S3$. Кроме того, D_j особым образом обрабатывает запросы с параметрами $(ind, salt)$, которые будем обозначать $S1$.

Противник D_j обрабатывает запросы ко всем оракулам, кроме $\mathcal{O}_{\text{send}}$, так же, как в $\mathbf{Exp}_3$. Запросы к оракулам $\mathcal{O}_{\text{send}}$ обрабатываются следующим образом:

— Запрос $S1$:
  1) Если $i \leqslant j$, то D_j выбирает случайный x^*, в качестве ответа возвращает $W + x^*P$.
  2) Если $i > j$, то D_j обрабатывает запрос так же, как в $\mathbf{Exp}_3$.

— Запрос $S2$ (параметр u_1):
  1) Если $i < j$, то запрос обрабатывается так же, как в $\mathbf{Exp}_4$.
  2) Если $i = j$, то в ответ на запрос возвращается $(B, u_2 = W)$. В качестве ответа на первом шаге задачи S-QPCCDH возвращает $(st, Y = u_1)$, получая в ответ пароль (st, PW'). Устанавливает пароль, разделяемый участниками A и B, равным PW', и устанавливает ключ

$$K_B = H_{256}((m/q(w - zr') \bmod q)(u_1 - r'Q_s)),$$

     где $r' = F(PW', salt, 2000)$.
  3) Если $i > j$, то запрос обрабатывается так же, как в $\mathbf{Exp}_3$.

— Запрос $S3$ (параметр $u2$):
  1) Если $i < j$, то запрос обрабатывается так же, как в $\mathbf{Exp}_4$.
  2) Если $i = j$, то возвращает $(st, Y = u_2)$ в качестве ответа на первом шаге задачи S-QPCCDH. Получает в ответ (st, PW'). Устанавливает PW' в качестве пароля, разделяемого между A и B. Устанавливает ключ

$$K_A = H_{256}((m/q(w + x^* - zr') \bmod q)(u_2 - r'Q_s)),$$

     где $r' = F(PW', salt, 2000)$.
  3) Если $i > j$, то запрос обрабатывается так же, как в $\mathbf{Exp}_3$.

Из запросов к оракулу $\mathcal{O}_H$ противник D_j извлекает ключи K (то есть точки, которые умножаются на m/q). Для каждой точки K он формирует две точки $K' = K$ и $K'' = K - x^*(Y - rQ_s)$. В качестве ответа D_j выдаёт список из $2q_H$ точек $\{K'_1, K''_1, K'_2, K''_2, \ldots, K'_{q_H}, K''_{q_H}\}$.

Противник D'_j

Этот противник отличается от D_j тем, что при обработке запросов $S2$ и $S3$ для установки ключа он использует функцию H', к которой у $\mathcal{A}_{\text{SESPAKE}_{KA}}$ нет доступа.

Отметим два важных обстоятельства, которые позволяют с помощью описанных противников оценить разность $|P_j - P_{j-1}|$.

С точки зрения $\mathcal{A}_{\mathrm{SESPAKE}_{KA}}$ поведение противника D'_j, то есть то, как он обрабатывает запросы к оракулам $\mathcal{O}_{\mathrm{send}}$, $\mathcal{O}_{\mathrm{exec}}$ и т. д., полностью совпадает с экспериментом $\mathbf{Hyb}_j$, поскольку «честная» обработка запросов к $\mathcal{O}_{\mathrm{send}}$ начинается лишь с $j+1$-го запроса. Для противника D_j «честная обработка» запросов к $\mathcal{O}_{\mathrm{send}}$ начинается уже с j-го запроса. Например, чему бы ни был равен параметр u_1 в запросе $S2$, множитель, на который умножается точка $u_1 - r'Q_s$, равен кратности точки, которая была возвращена в качестве ответа, то есть W, после снятия «маски» $r'Q_s$. Таким образом, A, получив данный ответ, получит тот же ключ, что и B, если A изначально (то есть на запрос $S1$) моделировался в соответствии с протоколом.

Поведение противника в экспериментах D_j и D'_j будет отличаться лишь в том случае, если он сделает запрос к оракулу $\mathcal{O}_H$ с параметрами, среди которых будет точка $\mathrm{CDH}(u_1 - r'Q_s, u_2 - r'Q_s)$, где u_1 или u_2 присутствовали в j-м запросе $S2$ или $S3$. Если такой параметр в запросах к $\mathcal{O}_H$ был, то найти $\mathrm{CDH}(W - r'Q_s, Y - r'Q_s)$ можно, так как

$$K_A = \mathrm{CDH}(u_2 - r'Q_s, W + x^*P - r'Q_s) = \mathrm{CDH}(W - r'Q_s, Y - r'Q_s) + x^*(Y - r'Q_s),$$
$$K_B = \mathrm{CDH}(u_1 - r'Q_s, W - r'Q_s) = \mathrm{CDH}(W - r'Q_s, Y - r'Q_s).$$

Учитывая способ формирования выходного списка из $2q_H$ точек противника D'_j, можно заключить, что искомое решение в нём найдётся. Таким образом, вероятность того, что среди параметров запросов к $\mathcal{O}_H$ найдётся $\mathrm{CDH}(u_1 - r'Q_s, u_2 - r'Q_s)$ с u_1, u_2, фигурировавшими в j-м запросе $S2$ или $S3$, можно оценить сверху наибольшей вероятностью успеха при решения задачи S-QPCCDH противником, работающим за время $t + q_{S1}\tau + C$, где q_{S1} — количество запросов $S1$ к участнику A; C — некоторая фиксированная константа. Таким образом, выполнено следующее неравенство:

$$\left|\Pr[\mathrm{SUCC}_3] - \underbrace{\Pr[\mathrm{SUCC}_4]}_{=1/2}\right| \leqslant \sum_{j=1}^{q_u} |P_j - P_{j-1}| \leqslant q_{\mathrm{send}} \cdot \mathbf{Insec}^{\text{S-QPCCDH}_{|\mathcal{D}|,2q_H}}(t + q_{S1}\tau + C).$$

Лемма 2 доказана. ■

Учитывая все полученные оценки, можно заключить, что выполнено неравенство

$$\left|\Pr[\mathrm{SUCC}_0] - \frac{1}{2}\right| \leqslant \frac{(2q_{\mathrm{exec}} + q_{\mathrm{send}})^2}{2q} + q_H \mathbf{Insec}^{\mathrm{CDH}}(t + 2\tau q_{\mathrm{exec}}) +$$
$$+ q_{\mathrm{send}} \cdot \mathbf{Insec}^{\text{S-QPCCDH}_{|\mathcal{D}|,2q_H}}(t + q_{S1}\tau + C).$$

Из этого соотношения непосредственно следует доказываемое неравенство для $\mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t, q_H, q_{\mathrm{send}}, q_{\mathrm{exec}}, q_{\mathrm{rev}})$. Теорема 5 доказана. ■

6. Оценка стойкости SESPAKE

6.1. Используемые подзадачи

Для обоснования стойкости протокола SESPAKE помимо модели, описанной в п. 3, введём несколько дополнительных задач, которые выступят в роли промежуточных. Отличия между всеми используемыми моделями заключаются в том, как работает оракул $\mathcal{O}_{\mathrm{auth}}$.

Отметим особенность промежуточных задач. Оракулы, с помощью которых формулируется угроза для $\mathrm{SESPAKE}_{KA}$ и SESPAKE, возвращают «честные» данные при $b = 1$ и случайные при $b = 0$. В задачах 1HMAC и 2HMAC мы поменяли смысловую

нагрузку значений, возвращаемых оракулом $\mathcal{O}_{\text{auth}}$: в этих задачах «более честные» данные возвращаются при $b = 0$, а не при $b = 1$. Это сделано для удобства оценки основных параметров.

6.2. Задача 1HMAC

В этой задаче противник имеет доступ ко всем оракулам, к которым имеет доступ противник в модели для протокола SESPAKE. Отличие состоит в том, что оракул $\mathcal{O}_{\text{auth}}$ возвращает не пару строк, а либо M'_A (при $b = 1$), либо M_A (при $b = 0$).

6.3. Задача 2HMAC

В задаче 2HMAC оракул $\mathcal{O}_{\text{auth}}$ в ответ на запрос возвращает противнику либо строки (M'_A, M') (при $b = 1$), либо строки (M_A, M'') (при $b = 0$). Строки определяются следующим образом:

1) $M_A = \text{HMAC}_{K_{A,i}}(T_A||IDS_{A,i})$;
2) $M'_A = \text{HMAC}_{K'}(T_A||IDS_{A,i})$ и $M' = \text{HMAC}_{K'}(T_B||IDS_{A,i})$, где $K' \xleftarrow{\mathcal{U}} V_{256}$;
3) $M'' = \text{HMAC}_{K''}(T_B||IDS_{A,i})$, где $K'' \xleftarrow{\mathcal{U}} V_{256}$.

6.4. Модель противника «с предупреждением»

Заметим, что преобладание $\mathbf{Adv}(\mathcal{A})$ некоторого противника, решающего задачу распознавания для бита b, можно преобразовать следующим образом:

$$\mathbf{Adv}(\mathcal{A}) = 2\Pr[\text{SUCC}] - 1 = 2\left[\frac{1}{2}\Pr[\mathcal{A} = 1|b = 1] + \frac{1}{2}(1 - \Pr[\mathcal{A} = 1|b = 0])\right] - 1 =$$
$$= \Pr[\mathcal{A} = 1|b = 1] - \Pr[\mathcal{A} = 1|b = 0].$$

Далее в основном будем оценивать именно таким образом преобразованную величину.

Теорема 6. Справедливо соотношение

$$\mathbf{Insec}^{\text{1HMAC}}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}, q_{\text{check}}) \leqslant$$
$$\leqslant \mathbf{Insec}^{AKE}_{\text{SESPAKE}_{KA}}(t + q_{\text{HMAC}} + q_{\text{check}}, q_{\text{exec}}, q_{\text{send}}, q_H) + O\left(\frac{1}{2^n}\right).$$

Доказательство. Пусть $\mathcal{A}$ — противник, решающий задачу 1HMAC. Опишем противника $\mathcal{A}'$, реализующего угрозу отличения ключа для протокола SESPAKE_{KA} и использующего $\mathcal{A}$ в качестве чёрного ящика. $\mathcal{A}'$ транслирует все запросы к имеющимся в его распоряжении оракулам ($\mathcal{O}_{\text{exec}}$, $\mathcal{O}_{\text{send}}$, $\mathcal{O}_{\text{rev}}$ и $\mathcal{O}_H$) и их ответы без изменений. При запросе к оракулу $\mathcal{O}_{\text{check}}$ противник $\mathcal{A}'$ делает запрос к $\mathcal{O}_{\text{rev}}$, получая ключ того оракула, для которого $\mathcal{A}$ хочет получить значение аутентификационных данных. После этого $\mathcal{A}'$ возвращает $\mathcal{A}$ «честное» значение запрашиваемых им данных, используя полученный ключ и оракул $\mathcal{O}_{\text{HMAC}}$. Если $\mathcal{A}$ делает запрос к оракулу $\mathcal{O}_{\text{auth}}$, то $\mathcal{A}'$ делает запрос к $\mathcal{O}_{\text{test}}$, получая либо случайный ключ K', либо истинный ключ K, который хранит тестируемый оракул. После этого $\mathcal{A}'$ передает $\mathcal{A}$ значение HMAC, вычисленное на полученном ключе. В качестве ответа $\mathcal{A}'$ возвращает значение, противоположное тому, которое вернул $\mathcal{A}$.

Из описания видно, что $\mathcal{A}'$ полностью моделирует эксперимент, в условиях которого работает противник $\mathcal{A}$. Преобладание $\mathbf{Adv}^{AKE}_{\text{SESPAKE}_{KA}}(\mathcal{A}')$ отличается от преобладания $\mathbf{Adv}^{\text{1HMAC}}(\mathcal{A})$ не более чем на величину порядка 2^{-n}, что объясняется тем, что значение HMAC на случайно выбранном ключе может совпасть со значением HMAC на сессионном ключе атакуемого оракула. Поэтому

$$\mathbf{Adv}^{AKE}_{\text{SESPAKE}_{KA}}(\mathcal{A}') \geqslant \mathbf{Adv}^{\text{1HMAC}}(\mathcal{A}) - O\left(\frac{1}{2^n}\right).$$

Поскольку $\mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}}, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) \geqslant \mathbf{Adv}^{AKE}_{\mathrm{SESPAKE}_{KA}}(\mathcal{A}')$, то

$$\mathbf{Adv}_{1\mathrm{HMAC}}(\mathcal{A}) = \mathbf{Insec}^{1\mathrm{HMAC}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}}) \leqslant$$
$$\leqslant \mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}}, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) + O\left(\frac{1}{2^n}\right).$$

Теорема 6 доказана. ■

Теорема 7. Справедливо соотношение

$$\mathbf{Insec}^{2\mathrm{HMAC}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}}) \leqslant$$
$$\leqslant \mathbf{Insec}^{1\mathrm{HMAC}}(t + 1, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}} + 1) + \frac{q_{\mathrm{HMAC}}}{2^{n-1}}.$$

Доказательство. Пусть $\mathcal{A} \in \mathcal{A}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}})$ — противник, решающий задачу 2HMAC с преобладанием $\mathbf{Insec}^{2\mathrm{HMAC}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}})$. Построим противника $\mathcal{A}'$, использующего $\mathcal{A}$ в качестве чёрного ящика и решающего задачу 1HMAC.

$\mathcal{A}'$ ничего не меняет во взаимодействии $\mathcal{A}$ со всеми доступными ему оракулами, кроме $\mathcal{O}_{\mathrm{auth}}$. Если $\mathcal{A}$ делает запрос к оракулу $\mathcal{O}_{\mathrm{auth}}$, то $\mathcal{A}'$ делает запрос к своему оракулу $\mathcal{O}_{\mathrm{auth}}$ и пересылает противнику $\mathcal{A}$ его ответ и значение M'' функции HMAC, вычисленное на случайно выбранном ключе K''. В итоге противник $\mathcal{A}$ фактически отличает следующие ситуации: он получил либо пару (M_A, M''), либо пару (M'_A, M''). Отличие от «честного» с точки зрения $\mathcal{A}$ эксперимента состоит в том, что пара (M'_A, M'') порождена с помощью двух независимо выбранных случайных ключей.

Справедливо соотношение

$$\mathbf{Adv}^{1\mathrm{HMAC}}(\mathcal{A}') = \Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M'_A, M'')] - \Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M_A, M'')] =$$
$$= \Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M'_A, M'')] - \Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M'_A, M')] +$$
$$+ \underbrace{\Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M'_A, M')] - \Pr[\mathcal{A} = 1 | \mathcal{A} \leftarrow (M_A, M'')]}_{=\mathbf{Adv}^{2\mathrm{HMAC}}(\mathcal{A})} \geqslant -\frac{2q_{\mathrm{HMAC}}}{2^n} + \mathbf{Adv}^{2\mathrm{HMAC}}(\mathcal{A}).$$

Действительно, первая из двух оцениваемых разностей не превосходит преобладания наилучшего противника, работающего с параметрами противника $\mathcal{A}$ и решающего задачу отличения пары выходов случайной функции, полученных на одном случайном ключе, от пары выходов случайной функции, полученных на двух независимых случайных ключах. Противник $\mathcal{A}$ сможет это сделать лишь в том случае, если среди его запросов к $\mathcal{O}_{\mathrm{HMAC}}$ присутствовал либо ключ K', либо ключ K'', один из которых $\mathcal{O}_{\mathrm{auth}}$ использовал в процессе работы. Поскольку оба ключа выбираются случайно, то вероятность такого события не превосходит $2q_{\mathrm{HMAC}}/2^n$.

Отметим также, что противник $\mathcal{A}'$ в процессе работы дополнительно порождает случайный ключ K'' и делает дополнительный запрос к оракулу $\mathcal{O}_{\mathrm{HMAC}}$. В итоге получаем

$$\mathbf{Adv}^{2\mathrm{HMAC}}(\mathcal{A}') \leqslant \mathbf{Adv}^{1\mathrm{HMAC}}(\mathcal{A}') + \frac{q_{\mathrm{HMAC}}}{2^{n-1}} =$$
$$= \mathbf{Insec}^{1\mathrm{HMAC}}(t + 1, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H, q_{\mathrm{HMAC}}, q_{\mathrm{check}} + 1) + \frac{q_{\mathrm{HMAC}}}{2^{n-1}}.$$

Теорема 7 доказана. ■

Теорема 8. Справедливо соотношение

$$\mathbf{Insec}_{\mathrm{SESPAKE}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_{\mathrm{check}}, q_H, q_{\mathrm{HMAC}}) \leqslant$$
$$\leqslant 2 \cdot \mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}} + 2, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) + \frac{q_{\mathrm{HMAC}}}{2^{n-1}} + O\left(\frac{1}{2^n}\right).$$

Доказательство. Пусть $\mathcal{A} \in \mathcal{A}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_{\mathrm{check}}, q_H, q_{\mathrm{HMAC}})$ — произвольный противник, реализующий угрозу для протокола SESPAKE с преобладанием $\mathbf{Adv}_{\mathrm{SESPAKE}}(\mathcal{A})$. Опишем противника $\mathcal{A}'$, который использует $\mathcal{A}$ для реализации угрозы для протокола $\mathrm{SESPAKE}_{KA}$. $\mathcal{A}'$ использует $\mathcal{A}$ так же, как это описано в теореме 6. Таким образом, если оракул $\mathcal{O}_{\mathrm{test}}$ вернул случайный ключ K', то противник $\mathcal{A}$ в ответ на запрос к $\mathcal{O}_{\mathrm{auth}}$ получит пару $(M'_A, M') = (\mathrm{HMAC}_{K'}(T_A||IDS_{A,i}), \mathrm{HMAC}_{K'}(T_B||IDS_{A,i}))$, иначе — пару $(M_A, M) = (\mathrm{HMAC}_{K_{A,i}}(T_A||IDS_{A,i}), \mathrm{HMAC}_{K_{A,i}}(T_B||IDS_{A,i}))$. Поскольку в качестве ответа $\mathcal{A}'$ возвращает то, что вернул $\mathcal{A}$, то для преобладания $\mathbf{Adv}(\mathcal{A}')$ выполнено соотношение

$$\mathbf{Adv}(\mathcal{A}') = \Pr[\mathcal{A} = 1|(M_A, M)] - \Pr[\mathcal{A} = 1|(M'_A, M')] =$$
$$= \underbrace{\Pr[\mathcal{A} = 1|(M_A, M)] - \Pr[\mathcal{A} = 1|(M_A, M'')]}_{S} +$$
$$+ \underbrace{\Pr[\mathcal{A} = 1|(M_A, M'')] - \Pr[\mathcal{A} = 1|(M'_A, M')]}_{T},$$

где $M'' = \mathrm{HMAC}_{K''}(T_B||IDS_{A,i})$ и $K'' \xleftarrow{\mathcal{U}} V_{256}$. Заметим, что S равно $\mathbf{Adv}_{\mathrm{SESPAKE}}(\mathcal{A})$.

Оценим величину T, то есть преобладание противника $\mathcal{A}$ в задаче отличения следующих ответов (M_A, M'') и (M'_A, M') оракула $\mathcal{O}_{\mathrm{auth}}$. Оценим сверху величину $-T$, получив тем самым нижнюю оценку для T. Для $-T$ справедливо соотношение $-T = \Pr[\mathcal{A} = 1|(M'_A, M')] - \Pr[\mathcal{A} = 1|(M_A, M'')] \leqslant \mathbf{Insec}^{\mathrm{2HMAC}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_{\mathrm{check}}, q_H, q_{\mathrm{HMAC}})$, так как $\mathcal{A}$ — конкретный противник из множества $\mathcal{A}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_{\mathrm{check}}, q_H, q_{\mathrm{HMAC}})$, а стоящее справа в неравенстве преобладание характеризует наиболее успешного при решении задачи 2HMAC противника из этого множества. Используя теоремы 6 и 7, получаем неравенство

$$T \geqslant -\mathbf{Insec}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}} + 2, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) - \frac{q_{\mathrm{HMAC}}}{2^{n-1}} - O\left(\frac{1}{2^n}\right).$$

В итоге имеем

$$\mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}}, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) \geqslant \mathbf{Adv}^{AKE}_{\mathrm{SESPAKE}_{KA}}(\mathcal{A}') \geqslant$$
$$\geqslant \mathbf{Adv}_{\mathrm{SESPAKE}}(\mathcal{A}) - \mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}} + 2, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) - \frac{q_{\mathrm{HMAC}}}{2^{n-1}} - O\left(\frac{1}{2^n}\right).$$

Следовательно,

$$\mathbf{Adv}_{\mathrm{SESPAKE}}(\mathcal{A}) = \mathbf{Insec}_{\mathrm{SESPAKE}}(t, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_{\mathrm{check}}, q_H, q_{\mathrm{HMAC}}) \leqslant$$
$$\leqslant 2 \cdot \mathbf{Insec}^{AKE}_{\mathrm{SESPAKE}_{KA}}(t + q_{\mathrm{HMAC}} + q_{\mathrm{check}} + 2, q_{\mathrm{exec}}, q_{\mathrm{send}}, q_H) + \frac{q_{\mathrm{HMAC}}}{2^{n-1}} + O\left(\frac{1}{2^n}\right).$$

Теорема 8 доказана. ■

Суть оценки, доказанной в теореме 8, состоит в том, что добавление к протоколу SESPAKE_{KA} этапа подтверждения ключа даёт противнику лишь критерий для проверки сеансового ключа, но не даёт никаких существенно новых возможностей для атаки на пароль. Это объясняется тем, что слагаемые $q_{\text{HMAC}}/2^{n-1}$ и $O(1/2^n)$ пренебрежимо малы по сравнению с первым слагаемым. Таким образом, наилучшим способом для атаки на пароль является его угадывание с активным вмешательством во взаимодействие по каналу, о чём говорит то, что первое слагаемое по порядку совпадает с $q_{\text{send}}/|\mathcal{D}|$.

6.5. Модель противника «без предупреждения»

Ранее рассмотрено преобладание $\mathbf{Adv}_{\text{SESPAKE}}$ противника $\mathcal{A}_{\text{SESPAKE}}$, который с помощью запросов к оракулу $\mathcal{O}_{\text{check}}$ предупреждает, по отношению к каким сессиям он не будет пытаться осуществить угрозу. Теорема 8 относится именно к этой модели противника. Переобозначим его преобладание через $\mathbf{Adv}_{\text{SESPAKE},w}$.

Теперь рассмотрим преобладание $\mathbf{Adv}_{\text{SESPAKE}}$ противника $\mathcal{A}_{\text{SESPAKE}}$, который может делать запросы к оракулу $\mathcal{O}_{\text{auth}}$ для любого участника протокола, независимо от того, сделан ли по отношению к нему запрос к оракулу $\mathcal{O}_{\text{check}}$ или нет. Условия, в которых работает противник, ничем не отличаются, кроме формата ответа на запрос к оракулу $\mathcal{O}_{\text{auth}}$:

— оракул $\mathcal{O}_{\text{auth}}$ принимает на вход пару (A, i), случайно и равновероятно выбирает значение $b \in \{0, 1\}$ и в зависимости от b возвращает либо M' (если $b = 0$), либо M_B (если $b = 1$). При этом $M' = \text{HMAC}_{K'}(T_B||IDS_{A,i})$, где $K' \xleftarrow{U} V_{256}$, $M_B = \text{HMAC}_{K_{A,i}}(T_B||IDS_{A,i})$.

Противник может делать запрос к оракулу $\mathcal{O}_{\text{auth}}$ только для тех оракулов, которые допускают тестирование. Оракулом, допускающим тестирование, называется такой оракул $\mathcal{O}_A^i$, для которого противник не получил значение M_B тривиальным образом, то есть не сделал запрос к $\mathcal{O}_{\text{check}}$ для оракула, разделяющего с $\mathcal{O}_A^i$ общий ключ. Оценим преобладание $\mathbf{Insec}_{\text{SESPAKE}}$.

Теорема 9. Справедливо соотношение

$$\begin{gathered}\mathbf{Insec}_{\text{SESPAKE}}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}) \leqslant \\ \leqslant q_{\text{send}} \cdot \mathbf{Insec}_{\text{SESPAKE},w}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}) + \\ +2q_H \cdot \mathbf{Insec}^{\text{CDH}}(t + \tau(2q_{\text{exec}} + q_{\text{send}})) + O\left(\frac{q_H + q_{\text{HMAC}}}{2^n}\right).\end{gathered}$$

Доказательство. Пусть $\mathcal{A}_{SESPAKE}$ — противник, действующий в описанной модели, с параметрами $t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}$ и преобладанием $\mathbf{Insec}_{\text{SESPAKE}}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}})$. Введём на его основе двух вспомогательных противников $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ и $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$.

Будем называть сессию Send-сессией, если она инициирована посредством запроса к оракулу $\mathcal{O}_{\text{send}}$, и Exec-сессией — к оракулу $\mathcal{O}_{\text{exec}}$. Через s_{auth} будем обозначать ту сессию, по отношению к участнику которой противник осуществил запрос к оракулу $\mathcal{O}_{\text{auth}}$. Тот факт, что сессия s_{auth} является Send-сессией (Exec-сессией), будем обозначать через $s_{\text{auth}} \in \text{Send}$ ($s_{\text{auth}} \in \text{Exec}$).

Противник $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ использует противника $\mathcal{A}_{\text{SESPAKE}}$ в качестве чёрного ящика и работает как транслятор между ним и оракулами, вплоть до обращения к оракулу $\mathcal{O}_{\text{auth}}$. Если зафиксировано обращение противника $\mathcal{A}_{\text{SESPAKE}}$ к оракулу $\mathcal{O}_{\text{auth}}$ для

Send-сессии ($s_{\text{auth}} \in \text{Send}$), то $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ выдаёт тот же бит, что и $\mathcal{A}_{\text{SESPAKE}}$; если сделано обращение к Exec-сессии ($s_{\text{auth}} \in \text{Exec}$), то $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ независимо от выхода оракула $\mathcal{O}_{\text{auth}}$ выдаёт случайный бит. Через $\text{SUCC}_{\text{SESPAKE}}$, $\text{SUCC}_{\text{SESPAKE}_{\text{send}}}$ и $\text{SUCC}_{\text{SESPAKE}_{\text{exec}}}$ будем обозначать событие, заключающееся в успешности решения задачи ложной аутентификации противником $\mathcal{A}_{\text{SESPAKE}}$, $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ и $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ соответственно.

Верно следующее равенство:

$$\begin{gathered}\Pr\left[\,\text{SUCC}_{\text{SESPAKE}_{\text{send}}}\,\right] = \\ = \Pr\left[\,\text{SUCC}_{\text{SESPAKE}} \mid s_{\text{auth}} \in \text{Send}\,\right] \cdot \Pr\left[\,s_{\text{auth}} \in \text{Send}\,\right] + \frac{1}{2} \cdot \Pr\left[\,s_{\text{auth}} \in \text{Exec}\,\right],\end{gathered} \tag{3}$$

где $\Pr\left[\,s_{\text{auth}} \in \text{Send}\,\right]$ ($\Pr\left[\,s_{\text{auth}} \in \text{Exec}\,\right]$) — вероятность того, что противник сделал запрос к оракулу $\mathcal{O}_{\text{auth}}$ по отношению к Send-сессии (Exec-сессии). Имеем в виду, что $\Pr\left[\,s_{\text{auth}} \in \text{Send}\,\right] + \Pr\left[\,s_{\text{auth}} \in \text{Exec}\,\right] = 1$.

Противник $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ работает аналогичным образом, только при запросе $\mathcal{A}_{\text{SESPAKE}}$ для Exec-сессии он выдаёт тот же бит, что и $\mathcal{A}_{\text{SESPAKE}}$, а при запросе к Send-сессии — случайный бит:

$$\begin{gathered}\Pr\left[\,\text{SUCC}_{\text{SESPAKE}_{\text{exec}}}\,\right] = \\ = \Pr\left[\,\text{SUCC}_{\text{SESPAKE}} \mid s_{\text{auth}} \in \text{Exec}\,\right] \cdot \Pr\left[\,s_{\text{auth}} \in \text{Exec}\,\right] + \frac{1}{2} \cdot \Pr\left[\,s_{\text{auth}} \in \text{Send}\,\right].\end{gathered} \tag{4}$$

Оценим преобладание противника $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$.

Лемма 3. Справедливо неравенство

$$\mathbf{Adv}\left(\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}\right) \leqslant 2q_H \cdot \mathbf{Insec}^{\text{CDH}}(t + \tau(2q_{\text{exec}} + q_{\text{send}}) + 3\tau) + O\left(\frac{q_H + q_{\text{HMAC}}}{2^n}\right).$$

Доказательство. Пусть $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ — противник с параметрами t, q_{exec}, q_{send}, q_H, q_{HMAC}, решающий задачу ложной аутентификации с преобладанием $\mathbf{Adv}\left(\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}\right)$. Построим на его основе противника $\mathcal{A}_{\text{CDH}}$, решающего вычислительную задачу CDH, и оценим снизу его вероятность успеха $\mathbf{Adv}\left(\mathcal{A}_{\text{CDH}}\right)$.

Пусть противнику $\mathcal{A}_{\text{CDH}}$ необходимо решить задачу CDH для произвольных точек Q_1 и Q_2. Будем использовать противника $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ в качестве чёрного ящика.

Противник $\mathcal{A}_{\text{CDH}}$ запускает противника $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ и симулирует выполнение протокола, где в ответ на запрос к оракулу $\mathcal{O}_{\text{exec}}$ в качестве u_1 и u_2 использует точки $(Q_1 + \alpha P) - Q_{PW}$ и $(Q_2 + \beta P) + Q_{PW}$, α и β — случайно выбранные значения. При этом в качестве сессионного ключа при запросах противника $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ к оракулам $\mathcal{O}_{\text{check}}$ и $\mathcal{O}_{\text{auth}}$ противник $\mathcal{A}_{\text{CDH}}$ использует значение произвольной неизвестной и недоступной для вычисления противнику $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$ случайной функции $H'(u_1, u_2)$. Затем он случайно выбирает один из q_H запросов к оракулу $\mathcal{O}_H$ (обозначим этот запрос через R), которые осуществлял противник $\mathcal{A}_{\text{SESPAKE}_{\text{exec}}}$, и вычисляет значение T следующим образом:

$$T = R - \beta Q_1 - \alpha Q_2 - \alpha\beta P,$$

где соответствующие α и β выбираются по значениям точек u_1 и u_2, исходя из запросов к оракулу $\mathcal{O}_{\text{HMAC}}$ и ответов на запросы к оракулу $\mathcal{O}_H$. Вероятность того, что для разных α и β мы получим одинаковые ключи, равна $O\left(2^{-n}\right)$.

Если противник $\mathcal{A}_{\mathrm{SESPAKE_{exec}}}$ правильно вычислил значение $R = \mathrm{CDH}(u_1 + Q_{PW}, u_2 - Q_{PW})$, то T равно искомому значению $\mathrm{CDH}(Q_1, Q_2)$. Тогда для построенного противника преобладание $\mathbf{Adv}\,(\mathcal{A}_{\mathrm{CDH}})$ равно $\Pr\,[T = \mathrm{CDH}(Q_1, Q_2)]$.

Обозначим через A событие, при котором среди запросов к оракулу $\mathcal{O}_H$ есть значение, из которого можно получить искомое $\mathrm{CDH}(Q_1, Q_2)$ по указанной формуле. Тогда верно следующее соотношение:

$$\mathbf{Insec}^{\mathrm{CDH}}(t + \tau(2q_{\mathrm{exec}} + q_{\mathrm{send}}) + 3\tau) \geqslant \mathbf{Adv}\,(\mathcal{A}_{\mathrm{CDH}}) = \frac{1}{q_H} \cdot \Pr\,[\ A\ ]\,.$$

Рассмотрим преобладание $\mathbf{Adv}\,(\mathcal{A}_{\mathrm{SESPAKE_{exec}}}) = 2\Pr\,[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}}\,] - 1$ противника $\mathcal{A}_{\mathrm{SESPAKE_{exec}}}$:

$$\begin{aligned}\Pr\,[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}}\,] = \underbrace{\Pr\,[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}} \mid A\,]}_{\leqslant 1} \cdot \Pr\,[\,A\,] + \\ + \Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}} \mid \overline{A}\,\right] \underbrace{(1 - \Pr\,[\,A\,])}_{\leqslant 1} \leqslant \Pr\,[\,A\,] + \Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}} \mid \overline{A}\,\right].\end{aligned} \tag{5}$$

Рассмотрим случай, когда противнику необходимо угадать значение бита b при условии, что среди запросов к оракулу $\mathcal{O}_H$ нет значения $\mathrm{CDH}(u_1 + Q_{PW}, u_2 - Q_{PW})$, и оценим вероятность его успеха $\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}} \mid \overline{A}\,\right]$.

В силу случайности функции H, данные, полученные в ответ на запрос к оракулу $\mathcal{O}_H$ со значениями, не равными $\mathrm{CDH}(u_1 + Q_{PW}, u_2 - Q_{PW})$, случайны и поэтому не несут никакой информации о настоящих сессионных ключах. Следовательно, в этом случае перед противником встает следующая задача: исходя только из ответов на запросы к оракулам $\mathcal{O}_{\mathrm{check}}$ и $\mathcal{O}_{\mathrm{auth}}$, определить, на каком ключе вычислено значение HMAC. Так как в такой ситуации для противника и при $b = 0$, и при $b = 1$ ключи выглядят одинаково случайными, то угадать значение бита с вероятностью, равной единице с точностью до $O\,(2^{-n})$, он может только с помощью следующей тактики: случайно выбрать ключ, вычислить на нём значения HMAC и сравнить полученные значения с ответами на запросы к оракулам $\mathcal{O}_{\mathrm{check}}$ и $\mathcal{O}_{\mathrm{auth}}$. Если вычисленные на выбранном ключе значения HMAC совпали с ответом на запрос и к оракулу $\mathcal{O}_{\mathrm{check}}$, и к оракулу $\mathcal{O}_{\mathrm{auth}}$, то противник выдаёт $b' = 1$, и наоборот, если вычисленные на выбранном ключе значения HMAC совпали с ответом на запрос либо к оракулу $\mathcal{O}_{\mathrm{check}}$, либо к оракулу $\mathcal{O}_{\mathrm{auth}}$, то выдаёт $b' = 0$.

Приближённое равенство вероятности успеха единице (с точностью до $O\,(2^{-n})$) объясняется вероятностью ошибиться за счёт случайности функции H, либо когда значения HMAC, вычисленные на случайно выбранном ключе, совпали и с ответом на запрос к оракулу $\mathcal{O}_{\mathrm{check}}$ (M_A), и с ответом на запрос к оракулу $\mathcal{O}_{\mathrm{auth}}$ при $b = 0$ (M') (противник ошибочно выдаст $b' = 1$), либо когда при $b = 1$ значения HMAC, вычисленные на случайно выбранном ключе, совпали только с одним из значений M_A или M_B (противник ошибочно выдаст $b' = 0$). Тогда

$$\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE_{exec}}} \mid \overline{A}\,\right] = \frac{1}{2} + O\Big(\frac{q_H + q_{\mathrm{HMAC}}}{2^n}\Big).$$

Отсюда с учётом (5) следует, что

$$2\Pr\,[\,A\,] \geqslant \mathbf{Adv}\,(\mathcal{A}_{\mathrm{SESPAKE_{exec}}}) - O\Big(\frac{q_H + q_{\mathrm{HMAC}}}{2^n}\Big).$$

Следовательно,

$$\mathbf{Insec}^{\mathrm{CDH}}(t+\tau(2q_{\mathrm{exec}}+q_{\mathrm{send}})+3\tau)\geqslant\frac{1}{2q_H}\left(\mathbf{Adv}\left(\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{exec}}}\right)-O\Big(\frac{q_H+q_{\mathrm{HMAC}}}{2^n}\Big)\right).$$

Лемма 3 доказана. ■

По формуле полной вероятности верно следующее равенство:

$$\begin{aligned}\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\,\right]=\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\mid s_{\mathrm{auth}}\in\mathrm{Send}\,\right]\cdot\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Send}\,\right]+\\+\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\mid s_{\mathrm{auth}}\in\mathrm{Exec}\,\right]\cdot\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Exec}\,\right].\end{aligned}\tag{6}$$

Преобразуем выражения (3) и (4):

$$\begin{aligned}\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\mid s_{\mathrm{auth}}\in\mathrm{Send}\,\right]\cdot\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Send}\,\right]=\\=\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}_{\mathrm{send}}}\,\right]-\frac{1}{2}\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Exec}\,\right];\end{aligned}\tag{7}$$

$$\begin{aligned}\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\mid s_{\mathrm{auth}}\in\mathrm{Exec}\,\right]\cdot\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Exec}\,\right]=\\=\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}_{\mathrm{exec}}}\,\right]-\frac{1}{2}\Pr\left[\,s_{\mathrm{auth}}\in\mathrm{Send}\,\right].\end{aligned}\tag{8}$$

Подставив (7) и (8) в правую часть равенства (6), окончательно получим

$$\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}_{\mathrm{send}}}\,\right]=\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}}\,\right]-\Pr\left[\,\mathrm{SUCC}_{\mathrm{SESPAKE}_{\mathrm{exec}}}\,\right]+\frac{1}{2},$$

или

$$\begin{aligned}\mathbf{Insec}_{\mathrm{SESPAKE}_{\mathrm{send}}}(t,q_{\mathrm{exec}},q_{\mathrm{send}},q_H,q_{\mathrm{HMAC}})=\\=\mathbf{Insec}_{\mathrm{SESPAKE}}(t,q_{\mathrm{exec}},q_{\mathrm{send}},q_H,q_{\mathrm{HMAC}})-\mathbf{Insec}_{\mathrm{SESPAKE}_{\mathrm{exec}}}(t,q_{\mathrm{exec}},q_{\mathrm{send}},q_H,q_{\mathrm{HMAC}}).\end{aligned}$$

Воспользовавшись леммой 3, приходим к следующей оценке:

$$\begin{aligned}\mathbf{Insec}_{\mathrm{SESPAKE}_{\mathrm{send}}}(t,q_{\mathrm{exec}},q_{\mathrm{send}},q_H,q_{\mathrm{HMAC}})\geqslant\mathbf{Insec}_{\mathrm{SESPAKE}}(t,q_{\mathrm{exec}},q_{\mathrm{send}},q_H,q_{\mathrm{HMAC}})-\\-2q_H\cdot\mathbf{Insec}^{\mathrm{CDH}}(t+\tau(2q_{\mathrm{exec}}+q_{\mathrm{send}}))-O\left(\frac{q_H+q_{\mathrm{HMAC}}}{2^n}\right).\end{aligned}$$

Построим противника «с предупреждением» $\mathcal{A}_{\mathrm{SESPAKE},w}$ на основе противника «без предупреждения» $\mathcal{A}_{\mathrm{SESPAKE}}$, используя его в качестве чёрного ящика. Для осмысленного противника «с предупреждением» логичнее делать запрос к оракулу $\mathcal{O}_{\mathrm{auth}}$ для участника Send-сессии, так как преобладание противника $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{exec}}}$ сравнимо с вероятностью успеха решения задачи CDH. Поэтому вместо противника $\mathcal{A}_{\mathrm{SESPAKE}}$ можно рассматривать противника $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$. В процессе атаки на протокол противник «с предупреждением», работая как транслятор между оракулами и противником $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$, случайно и равновероятно выбирает одну из Send-сессий, инициированных противником $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$, и делает запрос к оракулу $\mathcal{O}_{\mathrm{auth}}$ для участника выбранной сессии.

Если противник $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$ сделал запрос к оракулу $\mathcal{O}_{\mathrm{check}}$ для участника выбранной противником $\mathcal{A}_{\mathrm{SESPAKE},w}$ сессии, $\mathcal{A}_{\mathrm{SESPAKE},w}$ возвращает противнику $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$ первую часть ответа на сделанный им запрос к оракулу $\mathcal{O}_{\mathrm{auth}}$. Если противник $\mathcal{A}_{\mathrm{SESPAKE}_{\mathrm{send}}}$ сделал запрос к оракулу $\mathcal{O}_{\mathrm{check}}$ для участника другой сессии, $\mathcal{A}_{\mathrm{SESPAKE},w}$ просто транслирует этот запрос.

Если противник $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ сделал запрос к оракулу $\mathcal{O}_{\text{auth}}$ для участника выбранной противником $\mathcal{A}_{\text{SESPAKE},w}$ сессии, $\mathcal{A}_{\text{SESPAKE},w}$ возвращает противнику $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ вторую часть ответа на сделанный запрос к оракулу $\mathcal{O}_{\text{auth}}$, а в качестве результата выдаёт тот же бит, что и противник $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$. Если $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ сделал запрос к оракулу $\mathcal{O}_{\text{auth}}$ для участника другой сессии, $\mathcal{A}_{\text{SESPAKE},w}$ случайно выбирает ключ, вычисляет на нём значение M' и передаёт его противнику $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$, а в качестве результата выдаёт случайное значение бита.

Для противника $\mathcal{A}_{\text{SESPAKE},w}$ вероятность того, что выбранная им сессия совпадёт с сессией, для участника которой $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ сделает запрос к оракулу $\mathcal{O}_{\text{auth}}$, равна $1/q_{\text{send}}$. Следовательно, преобладание $\mathcal{A}_{\text{SESPAKE},w}$ меньше преобладания $\mathcal{A}_{\text{SESPAKE}_{\text{send}}}$ не более чем в q_{send} раз:

$$\mathbf{Insec}_{\text{SESPAKE},w}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}) \geqslant \frac{1}{q_{\text{send}}} \mathbf{Insec}_{\text{SESPAKE}_{\text{send}}}(t, q_{\text{exec}}, q_{\text{send}}, q_H, q_{\text{HMAC}}).$$

Отсюда получаем искомую оценку. Теорема 9 доказана. ■

7. Комментарии и пояснения

Обсудим некоторые конструктивные особенности протокола SESPAKE и их влияние на его свойства.

7.1. Ограничения количества сеансов для одного пароля

Говоря неформально, протокол SESPAKE считается стойким, если лучшим способом определения сессионного ключа (или какой-либо информации о нём) является опробование пароля путём взаимодействия с участником сети (будем называть такой способ online-угадыванием пароля). Недопустимой для стойкого протокола альтернативой online-угадыванию является практически осуществимый перебор, который не требует такого взаимодействия (offline-перебор).

При online-угадывании пароля противником каждая неудачная попытка приводит к разрыву соединения, вызванному тем, что проверки, предусмотренные протоколом SESPAKE_{KC}, не проходят. Отсутствие ограничений на количество возможных неудачных попыток соединения приводит к существенному увеличению вероятности online-угадывания. В связи с этим в протоколе SESPAKE должно быть предусмотрено ограничение как количества неудачных попыток соединения подряд, так и количества неудачных соединений для данного установленного пароля. Необходимо также вводить ограничение на общее количество соединений — как удачных, так и неудачных. Эти ограничения являются важной частью протокола.

Отметим влияние указанных ограничений на оценку теоремы 4. Так, например, в слагаемом $(2q_{\text{exec}} + q_{\text{send}})^2/q$ величина q_{exec} органичена количеством допустимых сеансов работы протокола, а q_{send} — ещё и количеством допустимых неудачных соединений для данного пароля. Обычно допустимое количество неудачных соединений полагается равным нескольким десяткам, а допустимое количество сеансов ограничивается величиной порядка 10^7–10^9. В таком случае, если $q \approx 2^{256}$, то

$$\frac{(2q_{\text{exec}} + q_{\text{send}})^2}{q} \leqslant (3 \cdot 10^9)^2/2^{256} \leqslant 2^{64}/2^{256} = 2^{-192}.$$

7.2. Обработка случая точки малого порядка

Рассмотрим причины, по которым при выработке общего ключа случай, когда $\frac{m}{q} Q_B = 0_E$ $\left(\text{или } \frac{m}{q} Q_A = 0_E\right)$, обрабатывается специальным образом. Под «специ-

альной обработкой» понимается использование флага z_b и точки P в качестве Q_B, если $\frac{m}{q} Q_B = 0_E$.

Протокол, в котором указанный случай не обрабатывается специальным образом, может быть уязвим к online-перебору паролей при нестрогом использовании счётчиков количества неудачных попыток аутентификации. Поясним этот тезис.

Пусть счётчик неудачных попыток аутентификации уменьшается в начале протокола SESPAKE$_{KC}$, а случай $\frac{m}{q} Q_B = 0_E$ не обрабатывается специальным образом. Тогда противник, имитируя сторону A, посылает стороне B точки $u_1 = X - Q_{PW'}$, где $\frac{m}{q} X = 0_E$, для разных PW' и измеряет время, через которое B отвечает на запрос; после ответа стороны B противник разрывает соединение. Таким образом, сторона B не уменьшает счетчик неудачных попыток соединения, а измеренное время даёт противнику критерий для определения правильного пароля — для него время будет существенно меньше, так как операция вычисления кратной точки (для получения src) окажется вырожденной.

Разница во времени проверки условия $\frac{m}{q} Q_B = 0_E$ для случаев $Q_B = 0_E$ и $Q_B \neq 0_E$ несущественна, так как кофактор на практике достаточно мал. Это объясняется тем, что характеристика p базового поля в целях оптимизации выбирается минимально возможной, чтобы достичь установленных в [28] значений параметра q ($2^{254} < q < 2^{256}$ или $2^{508} < q < 2^{512}$). Поэтому значения q и p оказываются близкими, а значит, малым оказывается и значение кофактора m/q.

Описанный метод анализа становится неприменим, если стороны уменьшают счётчик неправильных попыток аутентификации в самом начале работы.

Заключение

Протоколы типа PAKE являются одним из наиболее мощных механизмов для достижения стойкости аутентификации с использованием параметров, которые пользователь способен запомнить. Разработанный и стандартизированный в России протокол SESPAKE является самодостаточным протоколом такого типа, который может применяться в системах, в которых после аутентификации и выработки сеансовых ключей работает протокол защиты данных наподобие подпротокола Record протокола TLS. В настоящей работе приведены оценки стойкости данного протокола, которые позволяют выбирать значения его параметров для достижения требуемых порогов безопасности конечными реализациями.

ЛИТЕРАТУРА

1. *Bellovin S. and Merritt M.* Encrypted key exchange: password-based protocols secure against dictionary attacks // Proc. IEEE Symp. Research in Security and Privacy, Oakland, CA, USA, May 1992. P. 72–84.
2. *Bellare M. and Rogaway P.* The AuthA protocol for password-based authenticated key exchange // Contributions to IEEE P1363. March 2000. `https://web.cs.ucdavis.edu/~rogaway/papers/autha.pdf`
3. *Bresson E., Chevassut O., and Pointcheval D.* Security proofs for an efficient password-based key exchange // Proc. 10th ACM Conf. CCS'03. 2003. P. 241–250.
4. *Boyko V., MacKenzie P., and Patel S.* Provably secure password authenticated and key exchange using Diffie — Hellman // LNCS. 2000. V. 1807. P. 156–171.

5. *Canetti R., Halevi S., Katz J., et al.* Universally composable password-based key exchange // LNCS. 2005. V. 3494. P. 404–421.
6. *Bellare M., Canetti R., and Krawczyk H.* A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract) // Proc. 30th Ann. ACM Symp. Theory Comput. ACM Press, 1998. P. 419–428.
7. *Bellare M. and Rogaway P.* Entity authentication and key distribution // LNCS. 1994. V. 773. P. 232–249.
8. *Bellare M., Pointcheval D., and Rogaway P.* Authenticated key exchange secure against dictionary attacks // LNCS. 2000. V. 1807. P. 139–155.
9. *Abdalla M. and Pointcheval D.* Simple password-based encrypted key exchange protocols // LNCS. 2005. V. 3376. P. 191–208.
10. *Bender J., Fischlin M., and Kugler D.* Security analysis of the PACE key-agreement protocol // LNCS. 2009. V. 5735. P. 33–48.
11. *Jager T., Kohlar F., Schage S., and Schwenk J.* On the Security of TLS-DHE in the Standard Model. Cryptology ePrint Archive: Report 2011/219.
12. Рекомендации по стандартизации Р 50.1.115–2016 «Информационная технология. Криптографическая защита информации. Протокол выработки общего ключа с аутентификацией на основе пароля». М.: Стандартинформ, 2016.
13. *Алексеев Е. К., Ахметзянова Л. Р., Ошкин И. Б., Смышляев С. В.* Обзор уязвимостей некоторых протоколов выработки общего ключа с аутентификацией на основе пароля и принципы построения протокола SESPAKE // Математические вопросы криптографии. 2016. Т. 7. № 4. С. 7–28.
14. *Abdalla M.* Reducing The Need For Trusted Parties In Cryptography. `https://tel.archives-ouvertes.fr/tel-00917187`. 2011.
15. *Bellare M.* Practice-oriented provable-security // LNCS. 1998. V. 1396. P. 221–231.
16. *Ahmetzyanova L. R., Alekseev E. K., Karpunin G. A., and Smyshlyaev S. V.* On cryptographic properties of the CVV and PVV parameters generation procedures in payment systems // Математические вопросы криптографии. 2018. Т. 9. № 2. С. 23–46.
17. *Goldreich O.* Foundations of Cryptography. V. 1. N.Y.: Cambridge University Press, 2006.
18. *Алексеев Е. К., Ахметзянова Л. Р., Зубков А. М. и др.* Об одном подходе к формализации задач криптографического анализа // Математические вопросы криптографии. 2020 (в печати).
19. *Halderman J. A., Schoen S. D., Heninger N., et al.* Lest we remember: Cold-boot attacks on encryption keys // Commun. ACM. 2009. V. 52. No. 5. `http://www.cs.umd.edu/class/spring2016/cmsc414/papers/coldboot.pdf`
20. *Bellare M., Desai A., Jokipii E., and Rogaway P.* A concrete security treatment of symmetric encryption // Proc. 38th Symp. Foundations Comput. Sci. IEEE, 2000. P. 394–403.
21. *Abdalla M., Fouque P.A., and Pointcheval D.* Password-based authenticated key exchange in the three-party setting // LNCS. 2005. V. 3386. P. 65–84.
22. *Bellare M. and Rogaway P.* Random oracles are practical: a paradigm for designing efficient protocols // Proc. 1st ACM Conf. CCS'93. 1993. P. 62–73. `https://doi.org/10.1145/168588.168596`
23. *Alekseev E., Nikolaev V., and Smyshlyaev S.* On the security properties of Russian standardized elliptic curves // Математические вопросы криптографии. 2018. Т. 9. № 3. С. 5–32.
24. Рекомендации по стандартизации Р 50.1.111–2016 «Информационная технология. Криптографическая защита информации. Парольная защита ключевой информации». М.: Стандартинформ, 2016.

25. ГОСТ Р 34.11-2012 «Информационная технология. Криптографическая защита информации. Функция хэширования». М.: Стандартинформ, 2012.
26. Рекомендации по стандартизации Р 50.1.113–2016 «Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования». М.: Стандартинформ, 2016.
27. *Vercauteren F.* Final Report on Main Computational Assumptions in Cryptography. ICT-2007-216676. ECRYPT II, European Network of Excellence in Cryptology II. D.MAYA.6. 2013.
28. ГОСТ Р 34.10-2012 «Информационная технология. Процессы формирования и проверки электронной цифровой подписи». М.: Стандартинформ, 2012.

REFERENCES

1. *Bellovin S. and Merritt M.* Encrypted key exchange: password-based protocols secure against dictionary attacks. Proc. IEEE Symp. Research in Security and Privacy, Oakland, CA, USA, May 1992, pp. 72–84.
2. *Bellare M. and Rogaway P.* The AuthA protocol for password-based authenticated key exchange. Contributions to IEEE P1363, March 2000. `https://web.cs.ucdavis.edu/~rogaway/papers/autha.pdf`
3. *Bresson E., Chevassut O., and Pointcheval D.* Security proofs for an efficient password-based key exchange. Proc. 10th ACM Conf. CCS'03, 2003, pp. 241–250.
4. *Boyko V., MacKenzie P., and Patel S.* Provably secure password authenticated and key exchange using Diffie — Hellman. LNCS, 2000, vol. 1807, pp. 156–171.
5. *Canetti R., Halevi S., Katz J., et al.* Universally composable password-based key exchange. LNCS, 2005, vol. 3494, pp. 404–421.
6. *Bellare M., Canetti R., and Krawczyk H.* A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). Proc. 30th Ann. ACM Symp. Theory Comput., ACM Press, 1998, pp. 419–428.
7. *Bellare M. and Rogaway P.* Entity authentication and key distribution. LNCS, 1994, vol. 773, pp. 232–249.
8. *Bellare M., Pointcheval D., and Rogaway P.* Authenticated key exchange secure against dictionary attacks. LNCS, 2000, vol. 1807, pp. 139–155.
9. *Abdalla M. and Pointcheval D.* Simple password-based encrypted key exchange protocols. LNCS, 2005, vol. 3376, pp. 191–208.
10. *Bender J., Fischlin M., and Kugler D.* Security analysis of the PACE key-agreement protocol. LNCS, 2009, vol. 5735, pp. 33–48.
11. *Jager T., Kohlar F., Schage S., and Schwenk J.* On the Security of TLS-DHE in the Standard Model. Cryptology ePrint Archive: Report 2011/219.
12. Rekomendatsii po standartizatsii R 50.1.115–2016 «Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Protokol vyrabotki obshchego klyucha s autentifikatsiey na osnove parolya» ["Information Technology. Cryptographic Data Security. Password Authenticated Key Establishment Protocol". Recommendations for Standardization R 50.1.115–2016]. Moscow, Standartinform Publ., 2016. (in Russian)
13. *Alekseev E. K., Akhmetzyanova L. R., Oshkin I. B., and Smyshlyaev S. V.* Obzor uyazvimostey nekotorykh protokolov vyrabotki obshchego klyucha s autentifikatsiey na osnove parolya i printsipy postroeniya protokola SESPAKE [A review of the password authenticated key exchange protocols vulnerabilities and principles of the SESPAKE protocol construction]. Matem. Vopr. Kriptogr., 2016, vol. 7, no. 4, pp. 7–28. (in Russian)

14. *Abdalla M.* Reducing The Need For Trusted Parties In Cryptography. https://tel.archives-ouvertes.fr/tel-00917187. 2011.
15. *Bellare M.* Practice-oriented provable-security. LNCS, 1998, vol. 1396, pp. 221–231.
16. *Ahmetzyanova L. R., Alekseev E. K., Karpunin G. A., and Smyshlyaev S. V.* On cryptographic properties of the CVV and PVV parameters generation procedures in payment systems. Matem. Vopr. Kriptogr., 2018, vol. 9, no. 2, pp. 23–46.
17. *Goldreich O.* Foundations of Cryptography. Vol. 1. New York, Cambridge University Press, 2006.
18. *Alekseev E. K., Axmetzyanova L. R., Zubkov A. M., et al.* Ob odnom podkhode k formalizatsii zadach kriptograficheskogo analiza [On one approach to formalization of cryptographic analysis tasks]. Matem. Vopr. Kriptogr., 2020, to be published. (in Russian)
19. *Halderman J. A., Schoen S. D., Heninger N., et al.* Lest we remember: Cold-boot attacks on encryption keys. Commun. ACM, 2009, vol. 52, no. 5. http://www.cs.umd.edu/class/spring2016/cmsc414/papers/coldboot.pdf
20. *Bellare M., Desai A., Jokipii E., and Rogaway P.* A concrete security treatment of symmetric encryption. Proc. 38th Symp. Foundations Comput. Sci., IEEE, 2000, pp. 394–403.
21. *Abdalla M., Fouque P.A., and Pointcheval D.* Password-based authenticated key exchange in the three-party setting. LNCS, 2005, vol. 3386, pp. 65–84.
22. *Bellare M. and Rogaway P.* Random oracles are practical: a paradigm for designing efficient protocols. Proc. 1st ACM Conf. CCS'93, 1993, pp. 62–73. https://doi.org/10.1145/168588.168596
23. *Alekseev E., Nikolaev V., and Smyshlyaev S.* On the security properties of Russian standardized elliptic curves. Matem. Vopr. Kriptogr., 2018, vol. 9, no. 3, pp. 5–32.
24. Rekomendatsii po standartizatsii R 50.1.111–2016 «Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Parol'naya zashchita klyuchevoy informatsii» ["Information Technology. Cryptographic Data Security. Password-Based Key Protection". Recommendations for Standardization R 50.1.111–2016]. Moscow, Standartinform Publ., 2016. (in Russian)
25. GOST R 34.11-2012 «Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Funktsiya kheshirovaniya» [GOST R 34.11-2012 "Information Technology. Cryptographic Data Security. Hash Function"]. Moscow, Standartinform Publ., 2012. (in Russian)
26. Rekomendatsii po standartizatsii R 50.1.113–2016 «Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Kriptograficheskie algoritmy, soputstvuyushchie primeneniyu algoritmov elektronnoy tsifrovoy podpisi i funktsii kheshirovaniya» ["Information Technology. Cryptographic Data Security. Additional Cryptographic Algorithms for Digital Signature Algorithms and Hash Function". Recommendations for Standardization R 50.1.113–2016]. Moscow, Standartinform Publ., 2016. (in Russian)
27. *Vercauteren F.* Final Report on Main Computational Assumptions in Cryptography. ICT-2007-216676, ECRYPT II, European Network of Excellence in Cryptology II. D.MAYA.6. 2013.
28. GOST R 34.10-2012 «Informatsionnaya tekhnologiya. Protsessy formirovaniya i proverki elektronnoy tsifrovoy podpisi» [GOST R 34.10-2012 "Information Technology. Cryptographic Data Security. Processes of Digital Signature Creation and Verification"]. Moscow, Standartinform Publ., 2012. (in Russian)