А. И. Голенков, О. В. Ниссенбаум
Тюменский государственный университет, г. Тюмень
УДК 004.056.5

РАЗРАБОТКА СЕРВИСА МНОГОСЕРВЕРНОЙ АУТЕНТИФИКАЦИИ С ПОМОЩЬЮ МНОГОСТОРОННИХ БЕЗОПАСНЫХ ВЫЧИСЛЕНИЙ

Аннотация. В данной работе нами построен четырехсерверный протокол аутентификации пользователя с помощью многосторонних безопасных вычислений, сохраняющий работоспособность и обеспечивающий конфиденциальность ключа подписания JWT-токена и хэшей паролей пользователей даже при компрометации любого из серверов активным противником. Выполняемый протокол аутентификации состоит из шагов проверки хэша пароля и выдачи JWT-токена, подписанного цифровой подписью.

Ключевые слова: аутентификация, токен, цифровая подпись, MPC, безопасные многосторонние вычисления, разделение секрета, криптографический протокол.

Введение. Крупные приложения имеют огромные базы данных с учетными данными пользователей, и их кража может нанести значительный ущерб как оператору системы, так и ее пользователям, к приложению падает доверие, а сама система получает долговременные проблемы из-за недостоверных пользователей. Стандартная модель аутентификации предполагает, что один сервер аутентификации способен самостоятельно аутентифицировать пользователя, что также требует от него доступа к базе учетных данных. Подобная модель имеет очевидную критическую точку — сервер аутентификации. Если сервер будет скомпрометирован, то нарушитель сможет нарушить работу сервера, получить базу учетных данных (логины и хэши паролей пользователей), или даже авторизоваться под именем любого пользователя. В то же время, количество кибератак на информационные системы возрастает с каждым годом [1], одной из их основных целей являются учетные данные. В таких условиях, рискованно полагаться на сервер, обладающий полным контролем над учетными данными и процессом аутентификации. Мы предлагаем

избавиться от этой уязвимой точки, распределив задачу аутентификации на несколько серверов так, чтобы компрометация одного сервера не могла стать причиной утечки учетных данных или неправильной работы сервиса аутентификации. Для этого мы будем использовать концепцию **многосторонних безопасных вычислений** (Secure Multiparty Computation — MPC).

MPC — это криптографические протоколы, позволяющие сторонам совместно вычислять функцию над своими входными данными, не раскрывая при этом ни свои входные данные, ни промежуточные результаты вычислений другим участникам. Математическая база MPC была создана еще в 80-е годы [2, 3], но первое применение произошло в 2008 г. на аукционе для обеспечения приватности ставок [4]. В последующее десятилетие протоколы MPC нашли свое применение в различных задачах: аналитика, аукционы, электронные голосования, управление паролями и др. [5-7] В том числе, такие протоколы находят применение в задачах обеспечения информационной безопасности.

Протоколы MPC могут обеспечивать различные уровни безопасности в зависимости от количества скомпрометированных участников [8]. Если скомпрометированный участник не может отклоняться от выполнения протокола, то противник считается **пассивным**, если же участник может отклониться от протокола, выдав неверные данные, прервав протокол или как-то иначе, то противник считается **активным**. Активный нарушитель может прервать протокол до его завершения, чтобы устроить отказ в обслуживании или получить свои выходные данные и не дать получить выходные данные другим участникам.

В модели активного противника протокол может предоставлять различные гарантии безопасности. Это безопасность **с прерыванием** (атаки приводят к прерыванию протокола), **с идентифицируемым прерыванием** (то же самое, но скомпрометированный участник выявляется), **честные** протоколы (либо все честные участники получают одинаковый верный результат, либо никакой участник не получает результата). Наиболее сильные гарантии предоставляют протоколы **с гарантированной доставкой вывода**, или **с полной**

безопасностью, в которых честные участники получают верный вывод даже при наличии атакующего активного противника. Доказано, что построить такой протокол можно, только если количество нарушителей не превышает некоторое пороговое значение (половина или треть участников).

Цель работы — разработка протокола и сервиса многосерверной аутентификации с распределенным сервером. Мы поставили задачу разработать сервис аутентификации с использованием 4 серверов с полной безопасностью при компрометации одного любого сервера активным противником. Ни один из серверов при этом не должен обладать ни полным доступом к идентификационным данным пользователей, ни полным контролем над процессом аутентификации.

Мы проанализировали теоретические подходы к организации сервиса аутентификации. В статье Andrea Bissoli и Fabrizio d'Amore [9] рассматривается аутентификация с помощью пороговой схемы Шамира, данная схема позволяет реализовать различное количество участников, но она может обеспечить лишь безопасность с идентифицируемым прерыванием. Далее мы рассмотрели возможность повысить безопасность с прерыванием до полной безопасности. Две статьи Elette Boyle [10, 11] описывают подходы к повышению уровня безопасности от безопасности с идентифицируемым прерыванием до полной и описывают подобные протоколы, но наиболее полезной для нас оказалась статья Anders Dalskov [12], которая описывает протокол с полной безопасностью при менее трети скомпрометированных участников активным противником, а также фреймворк с полной безопасностью для 4 участников при 1 активном противнике.

4 участника будут оптимальным выбором для протокола с полной безопасностью, если активный противник может скомпрометировать лишь одного участника. Для реализации протокола с полной безопасностью при отсутствии широковещательного канала и наличии только каналов типа точка-точка между всеми участниками необходимо, чтобы было скомпрометировано менее трети участни-

ков. Широковещательный канал может быть эмулирован, но это потребует дополнительных вычислений и коммуникаций между участниками. Увеличение участников увеличит вычислительную нагрузку, поэтому мы останавливаем свой выбор на 4 серверах аутентификации.

В схеме аутентификации необходимо обратить внимание на возможность создания нарушителем учетной записи или компрометации клиента, который является более уязвимым, чем сервер, что в совокупности с компрометацией одного сервера не должно нарушить безопасность протокола.

1. Предыдущие решения. Вопрос о использовании MPC в аутентификации поднимал еще в 2016 году Найджел Смарт на конференции Microsoft Research [13]. Technical University of Darmstadt разработал протокол аутентификации PrivateDrop [14] для аутентификации устройств с операционной системой IOS или macOS между собой. Technical University of Cluj-Napoca опубликовал статью, в которой описал использование трехсерверной [15] системы аутентификации с возможностью отказа одного из серверов и одним пассивным противником. Была найдена публикация в блоге производителя о разделении им пароля и использовании MPC для последующей аутентификации [7]. Сравнительные характеристики перечисленных решений приведены в табл. 1.

Таблица 1

Сравнение решений аутентификации с MPC

Авторы	*Количество участников*	*Нарушители*	*Открытый доступ*	*Гарантированная доставка выхода*
Nigel P. Smart [13]	2	пассивные	нет	нет
университет Дармштадт [14]	2	пассивные	есть	нет
университет Cluj-Napoca [15]	3	активные	есть	только при отказе в обслуживании одним участником
Keyless [7]	неизвестно	неизвестно	нет	неизвестно
Наш	4	активные	есть	есть

2. Материалы и методы. Разрабатываемый нами сервис должен осуществлять распределенный безопасный двухшаговый процесс: сличение хэша пароля из базы с присланным хэшем пароля пользователя; в случае успешного прохождения проверки, выдачу jwt токена, подписанного цифровой подписью центра аутентификации. При этом, в соответствии с концепцией MPC, ни один из четырех серверов не будет иметь доступа ни к хранимым хэшам паролей, ни к секретному ключу подписания. Нашей задачей будет обеспечение конфиденциальности учетных данных пользователя, защита от подделки и гарантированная выдача jwt токена, даже если будет скомпрометирован один сервер.

2.1. Фреймворк DEN22. Для решения нашей задачи мы нашли фреймворк Anders Dalskov, Daniel Escudero и Ariel Nof [12], далее называемый DEN22, который реализует протокол MPC общего назначения для 4 участников, и предлагающий полную безопасность от одного активного противника. Фреймворк DEN22 позволяет работать в любом конечном кольце, в том числе и некоммутативном.

Этот фреймворк основан на концепции **разделения секрета**, в которой конфиденциальные данные (в нашем случае — ключи и хранимые хэши) — разделены между участниками протокола так, что никакой участник самостоятельно не способен восстановить никакой информации об этих данных. Для достижения возможности гарантированной доставки выхода используется **реплицированное разделение секрета**: пусть x — это некоторое конфиденциальное значение. Его можно представить в данном кольце в виде суммы четырех чисел: $x=x_1+x_2+x_3+x_4$. Каждому участнику (серверу) P_i выдается три из них: x_j, $i \neq j$. Так, у участника P_1 будут доли x_2, x_3, x_4 у участника P_2: x_1, x_3, x_4, P_3: x_1, x_2, x_4, P_4: x_1, x_2, x_3. При приведенном выше разделении секрета он может быть восстановлен с помощью любых двух участников. Разделенный таким образом секрет x мы будем обозначать как $[[x]]$.

В протоколе описываются **базовые операции**: сложение разделенных значений $[[x]]+[[y]]$, умножение разделенного значения на константу $a[[x]]$, сложение разделенного значения с константой $a+[[x]]$ и умножение двух разделенных значений $[[x]][[y]]$. При этом

все операции, кроме последней, реализуются без коммуникации между участниками, поэтому процедура аутентификации и выдачи токена должна состоять из таких операций над учетными данными, чтобы содержать как можно меньше операций умножения разделенных значений. Кроме того, описан протокол получения участниками (серверами) долей случайного числа $[[k]]$.

В случае ошибки фреймворк позволяет исключить пару участников, один из которых нарушил вычисление и был активным противником, при этом у нас остаются два участника, которые могут произвести повторное умножение вдвоем, так как противник остался среди исключенной пары.

Мы выделили два вида секретной информации, обрабатываемой на сервере аутентификации: хэш пароля пользователя h и секретный ключ для подписи jwt токена a. Эта информация будет храниться разделенной между серверами (обозначаемые $[[h]]$ и $[[a]]$ соответственно), что делает невозможным получение значений h и a ни одним из серверов самостоятельно. В многостороннем процессе аутентификации мы выделим две основные операции: проверка правильности введенного пользователем пароля и подписание токена секретным ключом.

2.2. Цифровая подпись. Важнейшим процессом нашего сервиса аутентификации является выдача токена аутентифицированному пользователю. Содержимое такого токена является открытым, и его формирование почти полностью можно доверить любому, даже скомпрометированному серверу, остальные сервера должны лишь следить за правильностью сформированного токена. Единственная часть токена, вычисление которой необходимо выполнить в безопасном многостороннем режиме — это цифровая подпись под ним. Существует множество стойких алгоритмов цифровой подписи, из них для нашего сервиса нужно выбрать тот или те, что допускают эффективное вычисление в нашем фреймворке, и создать протокол подписания. Рассмотрим 4 возможных алгоритма: Эль-Гамаля [16], Шнорра [17], DSA [18] и ГОСТ Р 34.10-94 [19]. Все они используют большое простое число p и представляют подпись под сообщением m в виде пары чисел (r, s). В таблице 2 приведены параметры, ключи

и процедуры этих подписей. Обозначения унифицированы в целях сравнения, под m понимается хэш от содержимого сертификата.

Таблица 2

Сравнение цифровых подписей

Алгоритм	*Эль-Гамаля*	*Шнорра*	*DSA*	*ГОСТ Р 34.10-94*
Параметры	p — б.п.ч., g — порождающий Z_p^*	p, q — б.п.ч.: $q \backslash (p-1)$, $g \in Z_p^*$ элемент порядка q.		
Секретный ключ	$a \in Z_p^*$	$a \in Z_q^*$		
Открытый ключ	$b = g^a \bmod p$			
Одноразовый ключ	$k \in Z_{p-1}^*$	$k \in Z_q^*$		
Подпись	$r = g^k \bmod p$ $s = k^{-1}(m - ar) \bmod (p-1)$	$r = H(m \mid g^k \bmod p)$ $s = k + ar \bmod q$	$r = (g^k \bmod p) \bmod q$ $s = k^{-1}(m + ar) \bmod q$	$r = (g^k \bmod p) \bmod q$ $s = km + ar \bmod q$

В схеме подписи секретным (а значит, в виде разделенного секрета) должны быть ключ подписания a и одноразовый ключ k, что мы в соответствии с нашей нотацией будем обозначать как $[[a]]$ и $[[k]]$. Напомним, что при используемом нами подходе, мы можем выполнять лишь операции сложения и умножения в кольце, причем коммуникаций многостороннего протокола требуют лишь умножения секретных значений, поэтому желательно использовать как можно меньше таких умножений.

Одноразовый секретный ключ $[[k]]$ в схеме Эль-Гамаля требует проверки на взаимную простоту с $(p-1)$, то есть слишком массивных вычислений в протоколе, т. к. k является разделенным секретом. В схеме DSA, нужно вычисление обратного к $[[k]]$, для чего потребуется $\lceil \log q \rceil$ умножений и возведений в квадрат (не менее 160 согласно стандарту). Схема Шнорра требует дополнительного вызова хэш-функции, к тому же она не является стандартной, и сама по себе

себе обладает некоторыми уязвимостями. Таким образом, наиболее подходящей для наших целей является ГОСТ Р 34.10-94, где оба ключа k и a разделены в кольце Z_q^*. Далее кратко опишем наш протокол **MultiSign** вычисления такой подписи.

Протокол MultiSign([[*a*]], *p*, *q*, *g*, *m*)

Вход: каждый сервер обладает долей секретного ключа [[a]], параметрами схемы p, q, g и хэшем от сертификата m.

1. Сервера выполняют протокол генерации случайных долей [[k]].
2. Каждый сервер, обладающий реплицированной долей k_i, вычисляет и публикует $r_i = g^{ki} \bmod p$ для $i = 1, 2, 3, 4$.
3. Каждый сервер принимает преобладающее значение r_i для i=1, 2, 3, 4 и вычисляет $r = (r_1 \cdot r_2 \cdot r_3 \cdot r_4 \bmod p) \bmod q$.
4. Каждый сервер локально вычисляет доли [[s]] = [[k]] $\cdot$ m + + [[a]]$\cdot r \bmod q$.

Выход: каждый сервер обладает r и долей [[s]].

Обратим внимание, что шаги 1–3 не зависят от m и поэтому могут быть вычислены серверами заранее, на этапе препроцессинга. Непосредственно между запросом сертификата и ответом в **MultiSign** выполняется только шаг 4, не требующий коммуникаций.

Безопасность протокола **MultiSign** обеспечивается фреймворком DEN22, за исключением раскрытия значений r_i на шаге 2, где она обеспечена предположением о сложности проблемы Диффи-Хэллмана, на которой основана также и стойкость подписи ГОСТ.

За рамками данного раздела остались подписи на эллиптических кривых. Алгоритмы подписания в них аналогичны рассмотренным, за исключением группы, в которой производится вычисление r. Все выводы и построенный нами протокол аналогичны описанным, т. е. наиболее предпочтительной подписью является ГОСТ Р 34.10-2018 [20], протокол многостороннего подписания в которой отличается от **MultiSign** лишь на шагах 2-3.

2.3. Наш 4-серверный протокол аутентификации на основе DEN22. В нашей системе клиент должен будет вычислить хэш пароля h и отправить серверам его доли [[h]]. При этом, в базе учетных данных для этого логина, сервера имеют [[H]]. Результатом

проверки будет разделенное значение [[*check*]], вычисленное серверами локально как [[*check*]]=[[H]]−[[*h*]]. Если *check* = 0, то хэши совпадают. Заметим, что доли *check* в любом случае выглядят для каждого отдельного сервера как набор трех случайных чисел.

Так как хэш непредсказуемо меняет свое значение при изменении пароля, то при неверном пароле *check* будет непредсказуема и ее можно считать случайной. Единица является нейтральным числом при умножении, поэтому мы будем умножать подпись на (*check*+1), что даст валидную подпись для *check* = 0, и случайное значение, из которого без знания секрета *check* (т. е. H) невозможно вычислить верную подпись. А именно, протокол аутентификации выглядит следующим образом:

Протокол MultiAuth([[H]], [[*a*]], *p, q, g*)

Вход: каждый сервер обладает долей секретного ключа [[*a*]], параметрами схемы подписания *p, q, g* и долей хранимого хэша пользователя [[H]].

1. Каждый сервер получает долю хэша пользователя [[*h*]] в Z_q.

2. Каждый сервер локально вычисляет долю контрольного значения [[*check*]] = [[H]] - [[*h*]], сертификат M и неподписанный хэш сертификата *m*.

3. Сервера совместно выполняют протокол (*r*,[[*s*]]) = **MultiSign**([[*a*]], *p, q, g, m*).

4. Сервера выполняют вычисление [[*s**]] = ([[*check*]] + 1)[[*s*]] в DEN22.

5. Сервера отправляют клиенту (M, *r*, [[*s**]])

Выход: клиент вычисляет *m* = h(M), восстанавливает *s** из [[*s**]] и проверяет подпись (*r, s**) под сертификатом M. В случае успешной верификации принимает (M, *r, s**) как действительный сертификат.

Заметим, что, если при *check* ≠ 0 подпись будет непредсказуемо изменена и не будет действительна. После получения результата серверы отправляют клиенту свои части подписи и остальную часть jwt токена. В случае выявления активного нарушителя оба сервера, из исключенной пары должны отправить сообщение, в котором указан второй из пары как нарушитель. Нарушитель может не отправить подобное сообщение, но отправка честного сервера защищает

клиента от получения неправильных данных. После получения ответа от серверов клиент получает 4 части подписи и может их объединить и проверить правильность подписи.

3. Результаты. Нами был построен четырехсерверный протокол аутентификации с использованием многосторонних безопасных вычислений с конфиденциальностью и гарантией выхода при не более чем одном активном противнике.

4. Заключение. В данной статье описаны базовые принципы построенного нами протокола многосерверной аутентификации с гарантированной доставкой выхода при одном активном противнике. В ходе дальнейшей работы над проектом разрабатывается программная реализация сервиса, с возможностью регистрации, преждевременного прекращения действия токена, реализовывать схему access и refresh токенов.

СПИСОК ЛИТЕРАТУРЫ

1. Актуальные киберугрозы: IV квартал 2022 года // Positive Technologies — 2023. — URL: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2022-q4/ (дата обращения 20.03.2023). — Текст: электронный.
2. Chaum D. Multiparty unconditionally secure protocols / Chaum D., Crépeau C., Damgard I. — Direct text // Proceedings of the twentieth annual ACM symposium on Theory of computing. — 1988. — P. 11-19.
3. Ben-Or M. Completeness theorems for non-cryptographic fault-tolerant distributed computation / Ben-Or M., Goldwasser S., Wigderson A. — Direct text // Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali. — 2019. — P. 351-371.
4. Bogetoft P. Multiparty computation goes live / Bogetoft P., Christensen D., Damgard I. et al. — Direct text // Cryptology ePrint Archive. — 2008.
5. Bestavros A. User-centric distributed solutions for privacy-preserving analytics / Bestavros A., Lapets A., Varia M. — Direct text // Communications of the ACM. — 2017. — Т. 60. — №. 2. — P. 37-39.
6. Nair D. G. An improved e-voting scheme using secret sharing based secure multi-party computation / Nair D. G., Binu V. P., Kumar G. S — Direct text // arXiv preprint arXiv:1502.07469. — 2015.
7. A beginner’s guide to Secure Multiparty Computation. // Keyless. — 2022. — URL: https://keyless.io/blog/post/a-beginners-guide-to-secure-

multiparty-computation (date of the application 10.05.2023). — Text: electronic.
8. Goldreich O. Foundations of cryptography: volume 2, basic applications. — Cambridge university press, 2009. — P. 599-765. — Direct text.
9. Bissoli A. Authentication as a service: Shamir Secret Sharing with byzantine components / Bissoli A., d'Amore F. — Direct text // arXiv preprint arXiv:1806.07291. — 2018.
10. Boyle E. Practical fully secure three-party computation via sublinear distributed zero-knowledge proofs / Boyle E., Gilboa N., Ishai Y., Nof A. — Direct text // Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. — 2019. — C. 869-886.
11. Boyle E. Efficient fully secure computation via distributed zero-knowledge proofs / E. Boyle, N. Gilboa, Y. Ishai, A. Nof. — Direct text // Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part III 26. — Springer International Publishing, 2020. — C. 244-276.
12. Dalskov A. Fast Fully Secure Multi-Party Computation over Any Ring with Two-Thirds Honest Majority / A. Dalskov, D. Escudero, A. Nof. — Direct text // Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. — 2022. — P. 653-666.
13. Nigel P. Smart Multi-Party Computation: From Theory to Practice // MicrosoftResearch. — 2016. — URL: https://www.youtube.com/watch?v=pNNLAEygPQI&ab_channel=MicrosoftResearch (date of the application 20.03.2023). Image (moving, three-dimensional): video.
14. Heinrich A. PrivateDrop: Practical Privacy-Preserving Authentication for Apple AirDrop / A. Heinrich, M. Hollick, T. Schneider [et al.]. — Direct text // USENIX Security Symposium. — 2021. — P. 3577-3594.
15. Fălămaş D. E. Assessment of Two Privacy Preserving Authentication Methods Using Secure Multiparty Computation Based on Secret Sharing / Fălămaş D. E., Marton K., Suciu A. — Direct text // Symmetry. — 2021. — T. 13, № 5. — P. 894.
16. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms / ElGamal T. — Direct text // IEEE transactions on information theory. — 1985. — T. 31, № 4. — P. 469-472.
17. Schnorr C. P. Efficient signature generation by smart cards / C. P. Schnorr. — Direct text // Journal of cryptology. — 1991. — T. 4. — P. 161-174.
18. Kerry C. F. Digital signature standard (DSS) / C. F. Kerry, P. D. Gallagher. — Direct text // FIPS PUB. — 2013. — P. 186-4.

19. ГОСТ Р 34.10–94. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма: национальный стандарт Российской Федерации : издание официальное: принят и введ. в действие постановлением Госстандарта России от 23.05.94 N 154 : введ. впервые : дата введ. 1995-01-01 / разработан Главным управлением безопасности связи Федерального агентства правительственной связи и информации и Всероссийским научно-исследовательским институтом стандартизации. — Москва : Издательство стандартов, 1994. 18 с. — Текст : непосредственный.
20. ГОСТ Р 34.10–2018. Информационная технология. Криптографическая защита информации. Процедуры формирования и проверки электронной цифровой подписи : национальный стандарт Российской Федерации: издание официальное : введ. в действие приказом Федерального агентства по техническому регулированию и метрологии от 4 декабря 2018 г. N 1059-ст: введ. впервые : дата введ. 2019-06-01 / разработан Центром защиты информации и специальной связи ФСБ России с участием ОАО «ИнфоТеКС». — Москва : Стандартинформ, 2018. — 21 с. — Текст : непосредственный.