МРНТИ 81.96.00

10.51889/2959-5894.2025.89.1.020

Р.М. Оспанов[1], Е.Н. Сейткулов[1*], Б.Б. Ергалиева[1], К.А. Утебаев[2], С.К. Атанов[1]

[1]Евразийский национальный университет им. Л.Н.Гумилева, г.Астана, Казахстан
[2]Алматинский филиал Национального исследовательского ядерного университета «МИФИ»
[*]e-mail: yerzhan.seitkulov@gmail.com

TANBA-SPHINCS$^+$ - ПОСТКВАНТОВЫЙ КРИПТОГРАФИЧЕСКИЙ АЛГОРИТМ ЦИФРОВОЙ ПОДПИСИ, ОСНОВАННОЙ НА ХЕШИРОВАНИИ

Аннотация

Целью работы является исследование и разработка нового постквантового алгоритма цифровой подписи, основанной на хешировании. В настоящее время одним из направлений постквантовой криптографии является исследование и разработка криптографических алгоритмов, основанных на хешировании. Безопасность таких алгоритмов основывается на использовании свойств безопасности криптографических хеш-функций. Известной постквантовой схемой цифровых подписей, основанной на хешировании, является схема SPHINCS$^+$, ставшая основой одного из современных стандартов постквантовой криптографии. В данной работе рассматривается новый алгоритм цифровой подписи TANBA-SPHINCS$^+$, являющийся модификацией SPHINCS$^+$. Алгоритм использует криптографическую хеш-функцию TANBA, разработанную на базе модифицированной схемы Sponge, где вместо одной внутренней функции применяется множество внутренних функций. Алгоритм TANBA-SPHINCS$^+$ сохраняет ключевые свойства SPHINCS$^+$, включая устойчивость к квантовым атакам и независимость от сложных криптографических предположений. Это делает алгоритм перспективным решением для задач постквантовой криптографии, позволяя адаптировать его под конкретные требования безопасности и производительности.

Ключевые слова: криптографический алгоритм, цифровая подпись, криптографическая хеш-функция, постквантовая криптография, асимметричная криптография.

Р.М. Оспанов[1], Е.Н. Сейткулов[1], Б.Б. Ерғалиева[1], К.А. Утебаев[2], С.К. Атанов[1]
[1]Л.Н. Гумилев атындағы Еуразия ұлттық университеті, Астана қ., Қазақстан
[2]Мәскеу инженерлік-физикалық институтының Ұлттық зерттеу ядролық университетінің Алматы филиалы

TANBA-SPHINCS$^+$ - ХЕШТЕУ НЕГІЗДЕЛГЕН ЦИФРЛЫҚ ҚОЛТАҢБАҒА АРНАЛҒАН ПОСТКВАНТТЫҚ КРИПТОГРАФИЯЛЫҚ АЛГОРИТМ

Аңдатпа

Бұл жұмыстың мақсаты хэшингке негізделген жаңа постквантты цифрлық қолтаңба алгоритмін зерттеу және әзірлеу болып табылады. Қазіргі уақытта постквантты криптографияның бір саласы хэштеу негізінде криптографиялық алгоритмдерді зерттеу және әзірлеу болып табылады. Мұндай алгоритмдердің қауіпсіздігі криптографиялық хэш функцияларының қауіпсіздік қасиеттерін пайдалануға негізделген. Хэшингке негізделген белгілі постквантты цифрлық қолтаңба схемасы постквантты криптографияның заманауи стандарттарының біріне негіз болған SPHINCS+ схемасы болып табылады. Бұл мақалада SPHINCS+ модификациясы болып табылатын TANBA-SPHINCS+ жаңа цифрлық қолтаңба алгоритмі талқыланады. Алгоритм TANBA криптографиялық хэш функциясын пайдаланады, ол өзгертілген Sponge схемасына негізделген, мұнда бір ішкі функцияның орнына бірнеше ішкі функциялар пайдаланылады. TANBA-SPHINCS+ алгоритмі SPHINCS+ негізгі қасиеттерін, соның ішінде кванттық шабуылдарға төзімділікті және күрделі криптографиялық жорамалдардан тәуелсіздікті сақтайды. Бұл алгоритмді арнайы қауіпсіздік пен өнімділік талаптарына бейімдеуге мүмкіндік беретін пост кванттық криптографиялық тапсырмалар үшін перспективалы шешімге айналдырады.

Түйін сөздер: криптографиялық алгоритм, цифрлық қолтаңба, криптографиялық хеш функциясы, постквантты криптография, асимметриялық криптография.

R.M.Ospanov[1], Ye.N.Seitkulov[1], B.B. Yergaliyeva[1], K.A. Utebayev[2]. S.K. Atanov[1]
[1]L.N. Gumilyov Eurasian National University, Astana, Kazakhstan
[2]Almaty branch of the National Research Nuclear University of Moscow Engineering Physics Institute

TANBA-SPHINCS$^+$ - POST-QUANTUM CRYPTOGRAPHIC HASH-BASED DIGITAL SIGNATURE ALGORITHM

Abstract

The goal of this work is to study and develop a new post-quantum algorithm for a hash-based digital signature. Currently, one of the areas of post-quantum cryptography is the study and development of hash-based cryptographic algorithms. The security of such algorithms is based on the use of security properties of cryptographic hash functions. A well-known post-quantum scheme of hash-based digital signatures is the SPHINCS+ scheme, which became the basis for one of the modern standards of post-quantum cryptography. This paper considers a new digital signature algorithm TANBA-SPHINCS+, which is a modification of SPHINCS+. The algorithm uses the TANBA cryptographic hash function, developed on the basis of a modified Sponge scheme, where multiple internal functions are used instead of one internal function. The TANBA-SPHINCS+ algorithm preserves key properties of SPHINCS+, including resistance to quantum attacks and independence from complex cryptographic assumptions. This makes the algorithm a promising solution for post-quantum cryptography tasks, allowing it to be adapted to specific security and performance requirements.

Keywords: cryptographic algorithm, digital signature, cryptographic hash function, post-quantum cryptography, asymmetric cryptography.

Основные положения

В данной работе рассматривается постквантовый криптографический алгоритм цифровой подписи, основанный на хешировании, TANBA-SPHINCS$^+$. Алгоритм представляет собой вариант реализации постквантовой схемы цифровой подписи SPHINCS$^+$. В этом варианте используется криптографическая хеш-функция TANBA, разработанная на базе модифицированной схемы Sponge, в которой применяются множество внутренних функций вместо одной. Такой подход обеспечивает высокую устойчивость к криптоаналитическим атакам и адаптируемость под различные требования безопасности. TANBA-SPHINCS$^+$ сохраняет ключевые преимущества SPHINCS$^+$, включая долгосрочную стойкость к атакам на квантовых компьютерах, и расширяет возможности схемы за счет инновационного подхода к конструкции хеш-функций. Это делает алгоритм перспективным решением для задач постквантовой криптографии.

Введение

В августе 2024 года Национальный институт стандартов и технологий США (NIST) официально объявил о первых трех завершенных стандартах проекта по стандартизации алгоритмов постквантовой криптографии и их готовности к немедленному использованию. Криптографические алгоритмы, представленные в новых стандартах, предназначены для решения двух основных криптографических задач: общее шифрование, используемое для защиты информации, передаваемой через общедоступную сеть, и цифровые подписи, используемые для аутентификации личности [1]. И главное их общее свойство – это криптографическая стойкость к атакам на квантовых компьютерах.

Стандарт FIPS 203 [2], предназначенный в качестве основного стандарта для общего шифрования, основан на алгоритме CRYSTALS-Kyber. В стандарте это алгоритм переименован в ML-KEM, сокращение от Module-Lattice-Based Key-Encapsulation Mechanism.

Стандарт FIPS 204 [3], предназначенный в качестве основного стандарта цифровых подписей, основан на алгоритме CRYSTALS-Dilithium. В стандарте это алгоритм переименован в ML-DSA, сокращение от Module-Lattice-Based Digital Signature Algorithm.

Стандарт FIPS 205 [4], предназначенный в качестве резервного стандарта цифровых подписей, основан на алгоритме SPHINCS$^+$ [5],[6], который был переименован в SLH-DSA, сокращение от Stateless Hash-Based Digital Signature Algorithm, т.е. алгоритм цифровой

подписи, основанной на хешировании, без сохранения состояния. Стандарт основан на другом математическом подходе, чем ML-DSA, и он предназначен в качестве резервного метода на случай, если ML-DSA окажется уязвимым.

Подход, применяемый в SPHINCS$^+$, состоит в использовании свойств безопасности криптографических хеш-функций для обеспечения безопасности цифровых подписей. Также такой метод позволяет проектировать алгоритмы цифровой подписи с любой безопасной криптографической хеш-функцией. Таким образом, если однажды в выбранной хеш-функции в будущем будет обнаружена уязвимость, то надо просто поменять её на новую безопасную хеш-функцию, тем самым сохраняя безопасность алгоритма цифровой подписи в целом. SPHINCS$^+$ представляет собой схему для построения семейства алгоритмов цифровой подписи. Так, изначально, в SPHINCS$^+$ рассматриваются варианты реализации с использованием алгоритмов SHA2, SHAKE и Haraka, а в версии схемы, изложенной в стандарте FIPS 205, т.е. в SLH-DSA, оставили варианты с SHA2 и SHAKE. В работе [7] рассматривается реализация SPHINCS$^+$ с использованием криптографической хеш-функции «Стрибог», описываемой в российском стандарте ГОСТ Р 34.11-2018 [8]. В работе [9] предлагается схема цифровой подписи, основанной на хешировании, K-SPHINCS$^+$, в которой хеш-функции из SPHINCS$^+$ заменены на корейские криптографические алгоритмы LSH, CHAM и LEA.

В данной работе предлагается новый вариант реализации схемы SPHINCS$^+$ с использованием криптографической хеш-функции TANBA, построенной на основе модифицированной схемы Sponge [10].

Методология исследования

Предлагаемый в данной работе криптографический алгоритм цифровой подписи основан на двух схемах, схеме цифровой подписи SPHINCS$^+$ и модифицированной схеме криптографической хеш-функции TANBA. И, таким образом, называется TANBA-SPHINCS$^+$.

Формально схему цифровой подписи можно определить следующим образом.

Пусть Σ – некоторый алфавит.

Σ^n, $n \in \mathbb{N}$ – множество всех слов длины n над алфавитом Σ.

Σ^∞ - множество всех слов над алфавитом Σ.

Схема цифровой подписи представляет собой некоторую совокупность алгоритмов, как минимум включающая в себя алгоритм формирования ключевой пары, алгоритм формирования цифровой подписи, алгоритм проверки цифровой подписи.

Алгоритмом формирования ключевой пары называется преобразование $G_A: 1^k \to \Sigma^{f_1(k)} \times \Sigma^{f_2(k)}$. Входные данные алгоритма составляет глобальная информация I, которая может содержать, например, параметр безопасности, некоторые математические параметры, описание используемой криптографической хеш-функции. В результате работы алгоритма получают пару открытого и закрытого ключей.

Алгоритмом формирования цифровой подписи называется преобразование $S_A: \Sigma^\infty \to \Sigma^n$. Аргументы этого преобразования называются сообщениями. Значения преобразования называются цифровыми подписями (соответствующих сообщений). Входными данными алгоритма являются сообщение M, глобальная информация I, закрытый ключ. В результате работы алгоритма получают сообщение M с подписью.

Алгоритмом проверки цифровой подписи называется преобразование $V_A: \Sigma^\infty \times \Sigma^n \to \{\text{true}, \text{false}\}$. Входными данными алгоритма являются открытый ключ, сообщение M и подпись. В результате работы алгоритма подпись принимается, если она действительна, или отклоняется в противном случае.

Схема цифровой подписи SPHINCS$^+$ представляет собой совокупность следующих алгоритмов: алгоритм формирования ключевой пары, алгоритм формирования цифровой подписи и алгоритм проверки цифровой подписи [5], [6].

Пусть $M = m_1 m_2 \dots m_k$, $m_i \in \{0,1\}$ – k-битное сообщение.

Алгоритм формирования ключевой пары (генерация секретного и открытого ключей) определяется следующим образом.

Пусть G – криптографически стойкий генератор псевдослучайных чисел. С помощью генератора G получают n-битовые sk_{seed}, pk_{seed} и sk_{prf}. Тогда секретный ключ будет состоять из sk_{seed}, sk_{prf}, pk_{seed} и pk_{root} ($sk = sk_{seed} \parallel sk_{prf} \parallel pk_{seed} \parallel pk_{root}$), а открытый ключ будет состоять из pk_{seed} и pk_{root} ($pk = pk_{seed} \parallel pk_{root}$). Здесь pk_{root} – открытый ключ гипердерева, т. е. открытый ключ (корневой узел) единственного дерева XMSS на верхнем слое.

Алгоритм формирования цифровой подписи определяется следующим образом.

Входными данными являются сообщение M и секретный ключ $sk = sk_{seed} \parallel sk_{prf} \parallel pk_{seed} \parallel pk_{root}$.

Во-первых, генерируется n-битовая строка рандомизации $R = PRF_{msg}(sk_{prf}, opt, M)$, где $PRF_{msg}: \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^n$ – псевдослучайная функция, opt – некоторое дополнительное случайное n-битовое значение. Затем, используя R, вычисляется $dg = H_{msg}(R, pk_{seed}, pk_{root}, M)$, где $H_{msg}: \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^m$ – хеш-функция с ключом. Далее, используя вычисленное значение dg, определяются md_{temp} – первые $\lfloor (k \log_2 t + 7)/8 \rfloor$ байтов dg, ti_{temp} – следующие $\lfloor (\mathrm{h} - \mathrm{h/d} + 7)/8 \rfloor$ байтов dg, li_{temp} – следующие $\lfloor (\mathrm{h/d} + 7)/8 \rfloor$ байтов dg. Далее определяются md – первые $k \log_2 t$ битов md_{temp}, ti – первые $\mathrm{h} - \mathrm{h/d}$ битов ti_{temp}, li – первые $\mathrm{h/d}$ битов li_{temp}. Используя полученные значения, последовательно определяются σ_{FORS} – подпись FORS и σ_{HT} – подпись гипердерева. Тогда подпись будет состоять из полученных R, σ_{FORS} и σ_{HT} ($\sigma = R \parallel \sigma_{FORS} \parallel \sigma_{HT}$).

Алгоритм проверки цифровой подписи определяется следующим образом.

Входными данными являются сообщение M, подпись $\sigma = R \parallel \sigma_{FORS} \parallel \sigma_{HT}$ и открытый ключ $pk = pk_{seed} \parallel pk_{root}$.

Во-первых, вычисляется $dg = H_{msg}(R, pk_{seed}, pk_{root}, M)$, определяются md_{temp} – первые $\lfloor (k \log_2 t + 7)/8 \rfloor$ байтов dg, ti_{temp} – следующие $\lfloor (\mathrm{h} - \mathrm{h/d} + 7)/8 \rfloor$ байтов dg, li_{temp} – следующие $\lfloor (\mathrm{h/d} + 7)/8 \rfloor$ байтов dg, затем определяются md – первые $k \log_2 t$ битов md_{temp}, ti – первые $\mathrm{h} - \mathrm{h/d}$ битов ti_{temp}, li – первые $\mathrm{h/d}$ битов li_{temp}. Далее выполняется вычисление кандидата на открытый ключ FORS и проверка подписи гипердерева на этом открытом ключе.

Криптографическая хеш-функция TANBA основана на модифицированной схеме Sponge, в которой используется множество внутренних функций [10].

Схема Sponge для вычисления хеш-значения заданного сообщения представляет собой алгоритм, состоящий из следующих преобразований: дополнение (padding), в результате которого входное сообщение дополняется некоторым количеством битов так, чтобы длина дополненного сообщения была кратна заданной длине блока сообщения; инициализация состояния, при котором устанавливается начальное (корневое) значение так называемого состояния, битовой последовательности некоторой определенной длины; «впитывание» (absorbing), при котором итеративно обрабатываются фрагменты сообщения определенной фиксированной длины, путем побитового сложения их с соответствующим фрагментом состояния, и затем применения к измененному состоянию заданной внутренней функции; «выжимание» (squeezing), при котором последовательно выводятся биты конечного хеш-значения путем извлечения битов из состояния и последующего применения к нему внутренней функции [11], [12].

Центральным компонентом схемы внутренняя функция, которая является либо перестановкой, либо преобразованием некоторой фиксированной длины. В предложенной в [10] модифицированной схеме вместо использования одной фиксированной внутренней функции, в процессе работы применяется набор внутренних функций. Выбор одной из них в

итерациях «впитывания» и «выжимания» осуществляется с помощью некоторой функции выбора, например, псевдослучайной функции.

В алгоритме TANBA начальное значение состояния состоит из определенного количества битов «0», за которыми следует 80-битное UNICODE-представление слова TANBA (в казахской транскрипции) (0...0 0000 0100 0010 0010 0000 0100 0001 0000 0000 0100 1010 0010 0000 0100 0001 0001 0000 0100 0001 0000).

Функция выбора, построенная с помощью демультиплексеров, корректора фон Неймана и XOR корректора, позволяет выбирать внутреннюю функцию из множества трех функций, описанных в [10]. Однако можно построить функцию выбора, работающую с любым количеством внутренних функций.

Выбор схемы SPHINCS$^+$ в качестве основы для проектирования криптографического алгоритма цифровой подписи обосновывается следующими важными факторами.

Схема SPHINCS$^+$ представляет собой гибкий фреймворк для построения квантово-устойчивых цифровых подписей, основанный на использовании свойств безопасности криптографических хеш-функций. Схема позволяет проектировать алгоритмы цифровой подписи с любой безопасной криптографической хеш-функцией. Таким образом, достаточно только выбирать новую безопасную криптографическую хеш-функцию каждый раз, как будут обнаружены проблемы с уже используемой в схеме хеш-функцией. Это в целом позволяет управлять безопасностью алгоритма цифровой подписи в целом. В работах [7], [9] предлагаются такие реализации схемы с заменой используемых криптографических хеш-функций.

Кроме того, схема SPHINCS$^+$ прошла все этапы конкурса NIST по выбору постквантовых криптографических алгоритмов, и в итоге на ее основе был принят стандарт FIPS 205 [4], предназначенный в качестве резервного стандарта цифровых подписей.

Алгоритм активно исследовался на протяжении прошедших лет, и никаких уязвимостей обнаружено не было [13], [14], [15].

Выбор схемы TANBA в качестве основы для проектирования криптографической хеш-функции, используемой в схеме SPHINCS$^+$, обосновывается следующими причинами.

Схема TANBA разработана на базе модифицированной схемы Sponge, в которой применяются множество внутренних функций вместо одной. Схема Sponge является известным фреймворком для построения криптографических хеш-функций. На основе ее была разработана известная криптографическая хеш-функция Keccak [16], вошедшая в стандарт FIPS 202 “SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions” [17].

Предполагается, что использование множества внутренних функций повышает устойчивость к атакам, направленным на компрометацию одной внутренней функции, и обеспечивает дополнительную гибкость и адаптивность алгоритма.

Результаты исследования

Предлагаемый в данной работе алгоритм TANBA-SPHINCS$^+$ является реализацией схемы SPHINCS$^+$, в которой хеш-функции из SPHINCS$^+$ заменены на хеш-функцию TANBA. В целом алгоритм представляет собой совокупность трех алгоритмов: алгоритм генерации открытого и секретного ключей, алгоритм генерации подписи и алгоритм проверки подписи. Алгоритм TANBA-SPHINCS$^+$ использует те же параметры и структуры, что и SPHINCS$^+$.

Параметры определяются следующим образом.

Параметр безопасности в байтах n задает длину выходных данных почти всех используемых внутри алгоритма семейств криптографических функций, а также размер практически любого узла в структуре алгоритма и, таким образом, размер всех элементов в подписи, т. е. размер подписи кратен n. n = 32

Параметр Винтерница w задает количество и длину цепочек хешей WOTS+. Параметр связан с размером и скоростью подписи. Чем больше значение параметра, тем короче подписи,

но медленнее скорость подписания. В то же время параметр не имеет значения для безопасности алгоритма в целом. Значение параметра принадлежат множеству {4, 16, 256}.

Параметр h, высота гипердерева, задает количество экземпляров FORS. Параметр имеет значение для безопасности алгоритма. Чем больше значение параметра, тем больше безопасность, но и в то же время больше размер подписи.

Параметр d задает количество слоев деревьев XMSS в гипердереве. Параметр имеет значение для производительности алгоритма, но не влияет на безопасность.

Понятно, что d должен делить без остатка h, получаем еще один параметр h' = h / d, который задает размер поддерева HT.

Параметры k, количество деревьев в FORS, и t, количество листьев в FORS, определяют производительность и безопасность FORS. Значение параметра t должно быть степенью двойки, а значение параметра k может быть выбрано свободно. Эти два параметра должны быть сбалансированы. Поскольку, с одной стороны, чем меньше значение t, тем меньше и быстрее подписи, но, с другой стороны, при заданном уровне безопасности при меньшем значении t требуется большее значение параметра k, что приводит к увеличению и меньшей скорости подписи.

Соответственно, параметр a = lg(t) определяет высоту деревьев FORS. = 9

Настройка набора значений этих параметров является темой отдельного исследования. В данной работе используются значения, определенные в [4], [5], [6].

Алгоритм, также как в SPHINCS+, использует различные семейства настраиваемых криптографических хеш-функций.

Настраиваемые хеш-функции (tweakable hash functions) – это специальный вид ключевых хеш-функций (хеш-функций с ключом) $H: \mathcal{P} \times \mathcal{T} \times \mathcal{M} \to \Sigma^n$ ($\mathcal{P} \subseteq \Sigma^\infty$ - множество открытых параметров, $\mathcal{T} \subseteq \Sigma^\infty$ - множество настроек, $\mathcal{M} \subseteq \Sigma^\infty$ - множество сообщений). Аргументами настраиваемых хеш-функций являются сообщения, открытые параметры и контекстная информация (настройки).

Каждый экземпляр SPHINCS+ ДОЛЖЕН описывать, как реализовать каждую из следующих функций. Для основных экземпляров, приведенных в этом документе, это будет сделано с использованием одной (хэш) функции, т. е. SHA2-256 или SHAKE-128. Конкретные экземпляры приведены в Разделе 7.

В частности, алгоритм использует следующие функции.

$PRF: \{0,1\}^n \times \{0,1\}^{256} \to \{0,1\}^n$ – псевдослучайная функция для генерации псевдослучайных ключей, определяется, как

$PRF(seed, adrs) = TANBA(seed||adrs, 8n)$.

$PRF_{msg}: \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^n$ – псевдослучайная функция для генерации случайных чисел при сжатии сообщений, определяется, как

$PRF_{msg}(sk_{prf}, opt, M) = TANBA(sk_{prf}||opt||M, 8n)$.

$H_{msg}: \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^m$ - дополнительная хеш-функция с ключом, которая может обрабатывать сообщения произвольной длины, определяется, как

$H_{msg}(R, pk_{seed}, pk_{root}, M) = TANBA(R||pk_{seed}||pk_{root}||M, 8m)$.

$T_l: \{0,1\}^n \times \{0,1\}^{256} \times \{0,1\}^{ln} \to \{0,1\}^n$, $F = T_1: \{0,1\}^n \times \{0,1\}^{256} \times \{0,1\}^n \to \{0,1\}^n$, $H = T_2: \{0,1\}^n \times \{0,1\}^{256} \times \{0,1\}^{2n} \to \{0,1\}^n$ – настраиваемые хеш-функции, определяются, как

$F(pk_{seed}, adrs, M_1) = TANBA(pk_{seed}||adrs||M_1, 8n)$,

$H(pk_{seed}, adrs, M_1||M_2) = TANBA(pk_{seed}||adrs||M_1||M_2, 8n)$,

$T_l(pk_{seed}, adrs, M) = TANBA(pk_{seed}||adrs||M, 8n)$, соответственно.

Здесь $adrs$ – адрес одного из следующих семи видов.

Адрес WOTS+ хеша – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_0 = lr_0 \parallel tr_0 \parallel tp_0 \parallel kp_0 \parallel ch_0 \parallel h_0$, где lr_0 – 32-битовое слово, значение которого определяет адрес слоя, tr_0 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_0 - 32-битовое слово, значение которого определяет тип адреса

(равно 0), kp_0 - 32-битовое слово, значение которого определяет адрес ключевой пары, ch_0 - 32-битовое слово, значение которого определяет адрес цепи, h_0 - 32-битовое слово, значение которого определяет адрес хеша.

Адрес сжатия WOTS+ открытого ключа – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_1 = lr_1 \| tr_1 \| tp_1 \| kp_1 \| pd_1$, где lr_1 – 32-битовое слово, значение которого определяет адрес слоя, tr_1 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_1 - 32-битовое слово, значение которого определяет тип адреса (равно 1), kp_1 - 32-битовое слово, значение которого определяет адрес ключевой пары, pd_1 - 64-битовое слово, значение которого определяет паддинг (равно 0).

Адрес хеш дерева – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_2 = lr_2 \| tr_2 \| tp_2 \| pd_2 \| th_2 \| ti_2$, где lr_2 – 32-битовое слово, значение которого определяет адрес слоя, tr_2 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_2 - 32-битовое слово, значение которого определяет тип адреса (равно 2), pd_2 - 32-битовое слово, значение которого определяет паддинг (равно 0), th_2 - 32-битовое слово, значение которого высоту дерева, ti_2 - 32-битовое слово, значение которого определяет индекс дерева.

Адрес FORS дерева – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_3 = lr_3 \| tr_3 \| tp_3 \| kp_3 \| th_3 \| ti_3$, где lr_3 – 32-битовое слово, значение которого определяет адрес слоя, tr_3 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_3 - 32-битовое слово, значение которого определяет тип адреса (равно 3), th_3 - 32-битовое слово, значение которого определяет высоту дерева, ti_3 - 32-битовое слово, значение которого определяет индекс дерева.

Адрес сжатия корней FORS дерева – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_4 = lr_4 \| tr_4 \| tp_4 \| kp_4 \| pd_4$, где lr_4 – 32-битовое слово, значение которого определяет адрес слоя, tr_4 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_4 - 32-битовое слово, значение которого определяет тип адреса (равно 4), kp_4 - 32-битовое слово, значение которого определяет адрес ключевой пары, pd_4 - 64-битовое слово, значение которого определяет паддинг (равно 0).

Адрес генерации ключа WOTS+ – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_5 = lr_5 \| tr_5 \| tp_5 \| ch_5 \| h_5$, где lr_5 – 32-битовое слово, значение которого определяет адрес слоя, tr_5 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_5 - 32-битовое слово, значение которого определяет тип адреса (равно 5), ch_5 - 32-битовое слово, значение которого определяет адрес цепи, h_5 - 32-битовое слово, значение которого определяет адрес хеша (равно 0).

Адрес генерации ключа FORS – это 256-битовое (32-байтовое) слово со следующей структурой $adrs_6 = lr_6 \| tr_6 \| tp_6 \| th_6 \| ti_6$, где lr_6 – 32-битовое слово, значение которого определяет адрес слоя, tr_6 – 96-битовое слово (3 32-битовых слова), значение которого определяет адрес дерева, tp_6 - 32-битовое слово, значение которого определяет тип адреса (равно 6), th_6 - 32-битовое слово, значение которого определяет высоту дерева, ti_6 - 32-битовое слово, значение которого определяет индекс дерева.

Идея структуры $adrs$ заключается в том, что каждое вычисление хеша в SPHINCS$^+$ получает отдельную структуру адреса. Поскольку структура адреса является входными данными для хеша, то при попытке нахождения хеша методом подбора, необходимо будет выбрать структуру $adrs$. Это исключает возможность воспользоваться преимуществами многоцелевых атак.

Алгоритм генерации открытого и секретного ключей определяется следующим образом.

Ключи формируются из n-байтовых значений sk_{seed}, sk_{prf}, pk_{seed} и pk_{root}, из которых первые три получаются с помощью некоторого криптографически стойкого генератора псевдослучайных чисел, а последний является вычисляемым открытым ключом гипердерева, т. е. открытым ключом (корневым узлом) единственного дерева XMSS на верхнем слое. Таким

образом, получаются открытый ключ $pk = pk_{seed} \parallel pk_{root}$ и секретный ключ $sk = sk_{seed} \parallel sk_{prf} \parallel pk_{seed} \parallel pk_{root}$.

Алгоритм формирования цифровой подписи определяется следующим образом.

Входными данными являются подписываемое сообщение M и секретный ключ sk.

Для генерации подписи вычисляется значение $R = TANBA(sk_{prf} ||opt||M, 8n)$, где opt – некоторое дополнительное случайное n-байтовое значение. Затем вычисляется $dg = TANBA(R||pk_{seed}||pk_{root}||M, 8m)$. Далее определяются md_{temp} – первые $\lfloor (k \log_2 t + 7)/8 \rfloor$ байтов dg, ti_{temp} – следующие $\lfloor (\mathrm{h} - \mathrm{h/d} + 7)/8 \rfloor$ байтов dg, li_{temp} – следующие $\lfloor (\mathrm{h/d} + 7)/8 \rfloor$ байтов dg. Далее определяются md – первые $k \log_2 t$ битов md_{temp}, ti – первые $\mathrm{h} - \mathrm{h/d}$ битов ti_{temp}, li – первые $\mathrm{h/d}$ битов li_{temp}. Используя полученные значения, последовательно определяются подпись FORS σ_{FORS} и подпись гипердерева σ_{HT}. В результате получается подпись $\sigma = R \parallel \sigma_{FORS} \parallel \sigma_{HT}$.

Алгоритм проверки цифровой подписи определяется следующим образом.

Входными данными являются сообщение M, подпись σ и открытый ключ pk.

Вычисляется $dg = TANBA(R||pk_{seed}||pk_{root}||M, 8m)$, определяются md_{temp} – первые $\lfloor (k \log_2 t + 7)/8 \rfloor$ байтов dg, ti_{temp} – следующие $\lfloor (\mathrm{h} - \mathrm{h/d} + 7)/8 \rfloor$ байтов dg, li_{temp} – следующие $\lfloor (\mathrm{h/d} + 7)/8 \rfloor$ байтов dg, затем определяются md – первые $k \log_2 t$ битов md_{temp}, ti – первые $\mathrm{h} - \mathrm{h/d}$ битов ti_{temp}, li – первые $\mathrm{h/d}$ битов li_{temp}. Далее выполняется вычисление кандидата на открытый ключ FORS и проверка подписи гипердерева на этом открытом ключе.

В целом, алгоритм TANBA-SPHINCS$^+$ представляет собой вариант реализации SPHINCS$^+$, в котором используется криптографическая хеш-функция TANBA, разработанная на базе модифицированной схемы Sponge с применением множества внутренних функций вместо одной. Научная значимость такого подхода заключается, в том, что он обеспечивает высокую устойчивость к криптоаналитическим атакам и адаптируемость под различные требования безопасности. Алгоритм сохраняет ключевые преимущества SPHINCS$^+$, включая долгосрочную стойкость к атакам на квантовых компьютерах, и расширяет возможности схемы за счет инновационного подхода к конструкции хеш-функций. Это делает алгоритм перспективным решением для задач постквантовой криптографии.

Дискуссия

SPHINCS$^+$ считается одним из самых устойчивых к квантовым атакам благодаря своей конструкции, основанной исключительно на свойствах хеш-функций. Криптостойкость схемы основывается на минимальных криптографических предположениях, относящихся к свойствам криптографических хеш-функций. Эти свойства считаются фундаментальными и широко изученными, что обеспечивает высокую надежность алгоритма даже в постквантовой среде. Квантовые алгоритмы, такие как алгоритм Гровера, могут ускорять поиск хеш-прообразов, но это дает лишь квадратичное ускорение. Это требует увеличения длины ключей и подписи для компенсации квантового ускорения. SPHINCS$^+$ учитывает это в своей конструкции, предлагая параметры, устойчивые даже к квантовым компьютерам. Алгоритм не полагается на задачи, которые могут быть решены алгоритмом Шора (например, факторизация или вычисление дискретного логарифма). Кроме того SPHINCS$^+$ устойчив к адаптивным атакам. Каждый одноразовый ключ подписи используется только один раз, что предотвращает утечку информации об основном закрытом ключе. А использование случайного выбора поддеревьев минимизирует вероятность предсказуемости используемых ключей. Использование в схеме гипердерева позволяет распределить вычисления и делает атаку на весь алгоритм менее эффективной. Детерминированный выбор путей и строгую привязку подписи к конкретному сообщению исключает возможность повторного использования или подмены ключей. Большой размер подписи (десятки килобайт) и сложная структура ключей затрудняют атаки. Однако, эти размеры могут быть проблемой для ресурсов

систем с ограниченными вычислительными мощностями. Несмотря на сильную безопасность SPHINCS$^+$, возможные уязвимости могут возникнуть из-за ошибок при программировании алгоритма (например, утечки побочных каналов или некорректной работы генераторов случайных чисел). А также если будет найден эффективный способ взлома используемой хеш-функции, это может поставить под угрозу всю схему. В этом случае надо просто поменять её на новую безопасную хеш-функцию, тем самым сохраняя безопасность алгоритма цифровой подписи в целом. Таким образом, схема SPHINCS$^+$ предлагает чрезвычайно высокий уровень криптографической безопасности благодаря хеш-ориентированному подходу и минимальной зависимости от сложных предположений. Устойчивость к квантовым атакам достигается за счет оптимальных параметров и защиты от известных квантовых алгоритмов. Главным вызовом остается размер подписи и сложность вычислений, что ограничивает его применение в некоторых сценариях.

Замена стандартных хеш-функций, используемых в SPHINCS$^+$ на хеш-функции, основанные на схеме Sponge, требует проверки их соответствия, предъявляемым к хеш-функциям в SPHINCS$^+$, чтобы гарантировать, что замена не ослабляет алгоритм. К преимуществам схемы Sponge можно отнести простота конструкции, гибкость (поддержка переменной длины входа и выхода), высокую производительность для функций с длинными входами. Схема Sponge обеспечивает устойчивость к коллизиям при условии, что используемая внутренняя функция обладает высокой криптографической стойкостью. Хеш-функции, основанные на схеме, обладают стойкостью к нахождению прообраза и второго прообраза, если длина состояния и блока достаточна для компенсации атак квантового типа (например, алгоритма Гровера). Все это делает схему Sponge подходящей для применения в SPHINCS$^+$. При этом следует учитывать, что длина состояния (например, 256 бит) должна быть достаточно большой, чтобы обеспечить стойкость против классических и квантовых атак, а размер выхода должен соответствовать параметрам SPHINCS$^+$. Для обеспечения стойкости против квантовых атак длина выхода хеша должна быть увеличена (например, до 512 бит вместо 256 бит), а внутренняя функция должна быть устойчивой к известным атакам.

Хеш-функции, основанные на схеме Sponge, предоставляют определённые преимущества, такие как гибкость и высокая производительность для длинных сообщений, что может сделать SPHINCS$^+$ более эффективным в некоторых сценариях. Однако важно учитывать потенциальные уязвимости, связанные с новой конструкцией. Пример оптимального набора параметров SPHINCS$^+$ при использовании хеш-функции на основе схемы Sponge зависит от конкретных требований к безопасности, производительности и ресурсам.

Использования множества нескольких внутренних функций вместо одной в модифицированной схеме Sponge делает конструкцию более устойчивой к атакам, направленным на нахождение слабостей конкретной внутренней функции. А применение псевдослучайной функции выбора обеспечивает непредсказуемость последовательности применяемых внутренних функций. Это затрудняет проведение криптоанализа, поскольку злоумышленник не знает заранее, какие функции будут использоваться на каждом шаге. Если злоумышленник выявляет уязвимости в одной внутренней функции, криптоанализ становится практически не эффективным, так как другие функции могут иметь совершенно разные свойства. Набор внутренних функций может быть расширяемым. Возможно динамическое добавление новых функций без необходимости изменения общей конструкции. Злоумышленник не только должен анализировать одну функцию, но и учитывать динамический выбор функций, что значительно усложняет проведение успешного криптоанализа.

Однако следует учитывать, что наличие нескольких функций требует большего объёма памяти и увеличивает вычислительную сложность. Реализация схемы с несколькими функциями требует более тщательной проверки, чтобы избежать ошибок, которые могут снизить безопасность.

Предлагаемый подход многовариантных схем Sponge с динамическим выбором функций существенно повышает стойкость криптографических алгоритмов. Однако это также требует тщательной проверки безопасности каждой из внутренних функций и реализации функции выбора, чтобы избежать компрометации системы.

Заключение

Обсуждаемый в данной работе алгоритм цифровой подписи TANBA-SPHINCS$^+$ является новым вариантом реализации схемы SPHINCS$^+$. Схема SPHINCS$^+$ - это схема постквантовой цифровой подписи, основанной на хешировании, разработанная для предоставления долгосрочного безопасного решения, которое может противостоять будущим атакам квантовых компьютеров. Для обеспечения криптографической стойкости схемы применяется подход, состоящий в использовании необходимого уровня криптостойкости криптографических хеш-функций. Это позволяет проектировать алгоритмы цифровой подписи с любой безопасной криптографической хеш-функцией. Схема SPHINCS$^+$ стала основой одного из стандартов проекта по стандартизации алгоритмов постквантовой криптографии. Это подтверждает важность SPHINCS$^+$ в постквантовой криптографии и обеспечивает его широкое распространение. Схема SPHINCS$^+$ дает возможность для построения различных алгоритмов цифровой подписи, используя различные криптографические хеш-функции. В оригинальной схеме SPHINCS$^+$ используются криптографические хеш-функции SHA2, SHAKE и Haraka, а в версии схемы, изложенной в стандарте FIPS 205, оставили варианты с использованием алгоритмов SHA2 и SHAKE. Существуют примеры реализации SPHINCS$^+$ с использованием и других криптографических хеш-функций. Рассмотренный в данной работе алгоритм цифровой подписи TANBA-SPHINCS$^+$ использует криптографическую хеш-функцию TANBA, построенную на основе модифицированной схемы Sponge. Схема Sponge является основой для построения множества различных современных криптографических алгоритмов хеширования, в том числе и стандартов. В модифицированной схеме вместо одной внутренней функции используется множество внутренних функций. Такой подход повышает устойчивость к атакам, направленным на компрометацию одной внутренней функции, и обеспечивает дополнительную гибкость и адаптивность алгоритма. Благодаря этому, TANBA-SPHINCS$^+$ сочетает в себе ключевые преимущества оригинальной схемы SPHINCS$^+$ и новые возможности, предоставляемые модифицированной схемой Sponge. Это делает алгоритм перспективным решением для задач постквантовой криптографии, позволяя адаптировать его под конкретные требования безопасности и производительности. Кроме того, использование множества внутренних функций в сочетании с псевдослучайным выбором их последовательности усложняет криптоанализ и делает схему еще более устойчивой к современным угрозам. TANBA-SPHINCS$^+$ демонстрирует, как эволюция криптографических подходов может открывать новые горизонты в обеспечении долгосрочной безопасности цифровых систем.

Благодарность

Работа выполнена при финансовой поддержке КН МНВО РК, грант № AP23486901.

Список использованных источников

[1] NIST Releases First 3 Finalized Post-Quantum Encryption Standards, Released August 13, 2024, Updated August 26, 2024 (https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards)

[2] National Institute of Standards and Technology (2024) Module-Lattice-Based Key-Encapsulation Mechanism Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 203. https://doi.org/10.6028/NIST.FIPS.203.

[3] National Institute of Standards and Technology (2024) Module-Lattice-Based Digital Signature Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 204. https://doi.org/10.6028/NIST.FIPS.204.

[4] National Institute of Standards and Technology (2024) Stateless Hash-Based Digital Signature Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 205. https://doi.org/10.6028/NIST.FIPS.205.

[5] Aumasson, et al. SPHINCS+ – Submission to the 3rd round of the NIST post-quantum project., 10 June 2022.

[6] Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. 2019. The SPHINCS+ Signature Framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19). Association for Computing Machinery, New York, NY, USA, 2129–2146. https://doi.org/10.1145/3319535.3363229.

[7] Kiktenko E., Bulychev A., Karagodin P., Pozhar N., Anufriev M., Fedorov A. Sphincs+ postquantum digital signature scheme with streebog hash function // AIP Conference Proceedings. vol. 2241, p. 020014. AIP Publishing LLC, 2020.

[8] ГОСТ 34.11-2018. Межгосударственный стандарт. Информационная технология. Криптографическая защита информации. Функция хэширования

[9] Sim M., Eum S., Song G., Kwon H., Jang K., Kim H., Kim H., Yang Y., Kim W., Lee W.K., et al. K-XMSS and K-SPHINCS+: Hash based signatures with korean cryptography algorithms // Cryptology ePrint Archive, 2022, https://eprint.iacr.org/2022/152.pdf

[10] Ospanov R.M., Seitkulov Ye.N., Yergaliyeva B.B. A cryptographic hash function based on a modified SPONGE scheme. Eurasian Journal of Mathematical and Computer Applications, 10(2), pp. 55-70, 2022.

[11] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G. Sponge functions // Ecrypt Hash Workshop 2007 (May 2007), http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html

[12] Bertoni G., Daemen J, Peeters M., Van Assche G. Cryptographic sponge functions. Version 0.1, January 14, 2011, https://keccak.team/files/CSF-0.1.pdf.

[13] Genêt, A. (2023). On Protecting SPHINCS+ Against Fault Attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2023(2), 80–114. https://doi.org/10.46586/tches.v2023.i2.80-114.

[14] Hülsing, A., & Kudinov, M. (2023). Recovering the Tight Security Proof of SPHINCS+. In S. Agrawal, & D. Lin (Eds.), Advances in Cryptology – ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV (pp. 3-33). (Lecture Notes in Computer Science (LNCS); Vol. 13794). Springer Nature. https://doi.org/10.1007/978-3-031-22972-5_1.

[15] Q. Berthet, A. Upegui, L. Gantel, A. Duc and G. Traverso, "An Area-Efficient SPHINCS+ Post-Quantum Signature Coprocessor," 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 2021, pp. 180-187, doi: 10.1109/IPDPSW52791.2021.00034.

[16] Bertoni G., Daemen J, Peeters M., Van Assche G. The Keccak reference. SHA-3 competition (round 3), 2011, https://keccak.team/sponge_duplex.html.

[17] National Institute of Standards and Technology (2015) SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 202. https://doi.org/10.6028/NIST.FIPS.202.

References

[1] NIST Releases First 3 Finalized Post-Quantum Encryption Standards, Released August 13, 2024, Updated August 26, 2024 (https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards)

[2] National Institute of Standards and Technology (2024) Module-Lattice-Based Key-Encapsulation Mechanism Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 203. https://doi.org/10.6028/NIST.FIPS.203.

[3] National Institute of Standards and Technology (2024) Module-Lattice-Based Digital Signature Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 204. https://doi.org/10.6028/NIST.FIPS.204.

[4] National Institute of Standards and Technology (2024) Stateless Hash-Based Digital Signature Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 205. https://doi.org/10.6028/NIST.FIPS.205.

[5] Aumasson, et al. SPHINCS+ – Submission to the 3rd round of the NIST post-quantum project., 10 June 2022.

[6] Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. 2019. The SPHINCS+ Signature Framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19). Association for Computing Machinery, New York, NY, USA, 2129–2146. https://doi.org/10.1145/3319535.3363229.

[7] Kiktenko E., Bulychev A., Karagodin P., Pozhar N., Anufriev M., Fedorov A. Sphincs+ postquantum digital signature scheme with streebog hash function // AIP Conference Proceedings. vol. 2241, p. 020014. AIP Publishing LLC, 2020.

[8] GOST 34.11-2018. Mezhgosudarstvennyi standart. Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Funktsiya kheshirovaniya. [Interstate standard. Information technology. Cryptographic protection of information. Hash function] (In Russian)

[9] Sim M., Eum S., Song G., Kwon H., Jang K., Kim H., Kim H., Yang Y., Kim W., Lee W.K., et al. K-XMSS and K-SPHINCS+: Hash based signatures with korean cryptography algorithms // Cryptology ePrint Archive, 2022, https://eprint.iacr.org/2022/152.pdf

[10] Ospanov R.M., Seitkulov Ye.N., Yergaliyeva B.B. A cryptographic hash function based on a modified SPONGE scheme. Eurasian Journal of Mathematical and Computer Applications, 10(2), pp. 55-70, 2022.

[11] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G. Sponge functions // Ecrypt Hash Workshop 2007 (May 2007), http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html

[12] Bertoni G., Daemen J, Peeters M., Van Assche G. Cryptographic sponge functions. Version 0.1, January 14, 2011, https://keccak.team/files/CSF-0.1.pdf.

[13] Genêt, A. (2023). On Protecting SPHINCS+ Against Fault Attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2023(2), 80–114. https://doi.org/10.46586/tches.v2023.i2.80-114.

[14] Hülsing, A., & Kudinov, M. (2023). Recovering the Tight Security Proof of SPHINCS+. In S. Agrawal, & D. Lin (Eds.), Advances in Cryptology – ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV (pp. 3-33). (Lecture Notes in Computer Science (LNCS); Vol. 13794). Springer Nature. https://doi.org/10.1007/978-3-031-22972-5_1.

[15] Q. Berthet, A. Upegui, L. Gantel, A. Duc and G. Traverso, "An Area-Efficient SPHINCS+ Post-Quantum Signature Coprocessor," 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 2021, pp. 180-187, doi: 10.1109/IPDPSW52791.2021.00034.

[16] Bertoni G., Daemen J, Peeters M., Van Assche G. The Keccak reference. SHA-3 competition (round 3), 2011, https://keccak.team/sponge_duplex.html.

[17] National Institute of Standards and Technology (2015) SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publications (FIPS) NIST FIPS 202. https://doi.org/10.6028/NIST.FIPS.202.