МРНТИ 81.93.29
УДК 004.056.5 **DOI 10.70239/arsu.2026.t84.n2.07**

ПОДХОД К ИССЛЕДОВАНИЮ ЭФФЕКТИВНОСТИ КРИПТОГРАФИЧЕСКИХ ПРОТОКОЛОВ РАСПРЕДЕЛЕНИЯ КЛЮЧЕЙ

ИЛИПОВ М.М.*, АМАНГЕЛДІ Н.Н.

***Илипов Марлен Маукенович** - Магистр физики и компьютерных технологий, старший преподаватель кафедры компьютерных наук, заместитель директора по воспитательной и социальной работе, Казахский агротехнический исследовательский университет имени Сакена Сейфуллина, г. Астана, Казахстан.
E-mail: Marlen.ilipov@mail.ru, https://orcid.org/0009-0005-8224-8847
Амангелді Нұршат Нұрланбекқызы – Студент 2-го курса образовательной программы «Бизнес-информатика», Казахский агротехнический исследовательский университет имени Сакена Сейфуллина, г. Астана, Казахстан.
E-mail: nurshatamangeldi@gmail.com, https://orcid.org/0009-0006-4626-4513

Аннотация. Актуальность исследования обусловлена необходимостью постоянного совершенствования механизмов оценки безопасности криптографических протоколов распределения ключей в условиях роста числа кибератак (15–20% ежегодно) и расширения применения низкоресурсных устройств (микропроцессорные карты, IoT). Существующие методы сравнения протоколов часто фокусируются лишь на отдельных аспектах, что не позволяет получить комплексную картину их эффективности. Целью статьи является разработка и программная верификация подхода к количественному сравнению эффективности протоколов, учитывающего свойства безопасности (G1–G9), стойкость к атакам (A1–A7) и аппаратные ограничения (объём памяти, число сообщений). Предложенный подход базируется на многокритериальной оценке с весовыми коэффициентами, что позволяет нивелировать субъективность при выборе протокола под конкретные условия эксплуатации. На примере модификаций протокола Диффи–Хеллмана (STS, MTI, DH1, DH2) получены обобщённые показатели эффективности, наибольший из которых у DH1 (0,754). Для практической верификации разработана программа на Python с использованием библиотек cryptography и hashlib, реализующая RSA-2048, DH-2048 и ECDH-P256. Экспериментально установлено, что ECDH-P256 обеспечивает наименьшее время вычисления общего секрета (0,000713 с), что в 21 раз быстрее DH-2048. Моделирование аппаратного ускорителя на базе ПЛИМ показало 4-кратное ускорение операций ECDH по сравнению с чисто программной реализацией. Полученные результаты подтверждают применимость разработанного подхода для выбора оптимальных протоколов в условиях жёстких аппаратных ограничений. Результаты предназначены для специалистов по защищённым информационным системам и разработчиков IoT-устройств, а также могут быть использованы при создании стандартизированных методик оценки криптографических протоколов.

Ключевые слова: криптографические протоколы, распределение ключей, Diffie–Hellman, ECDH, эффективность, микропроцессорные карты, аппаратное ускорение.

Введение

В настоящее время область использования микропроцессорных карт (smart cards) охватывает финансовый сектор, системы информационной безопасности, контроля доступа, электронные удостоверения, мобильные устройства и другие сферы. Согласно совместному исследованию Sony Corporation, HID Global и других, рынок микропроцессорных карт к 2025 году достигнет 11,5 млрд долларов США. Столь динамичный рост объясняется повсеместной цифровизацией, переходом на бесконтактные технологии и ужесточением требований к безопасности.

Микропроцессорные карты всё чаще выступают носителем ключевой информации в распределённых информационных системах. В условиях интернета вещей (IoT) обеспечение конфиденциальности и целостности передаваемых данных становится критически важным. Злоумышленники постоянно совершенствуют методы перехвата, подмены сообщений и атак на криптографические протоколы [1; 2]. Поэтому выбор надёжного и быстрого протокола распределения ключей – ключевой этап проектирования защищённой системы.

Одной из наиболее реализуемых групп является семейство, основанное на протоколе Диффи–Хеллмана (DH). Классический DH уязвим для атаки «человек посередине» (MITM), поскольку в нём отсутствует аутентификация участников [3]. Кроме того, его вычислительная сложность (модулярное экспоненцирование с 2048-битными операндами) может быть неприемлемо высокой для низкоресурсных устройств [4].

На теоретической базе DH были разработаны усовершенствованные версии: STS, MTI, DH1 и DH2 [5, 112]. Однако их сравнительная эффективность на микропроцессорных картах и IoT-устройствах изучена недостаточно. Требуют оценки не только криптостойкость, но и практические параметры – объём памяти, число сообщений, устойчивость к компрометации долговременных ключей и атакам с исчерпанием ресурсов [6; 7].

Цель статьи – разработка подхода к определению эффективности криптографических протоколов распределения ключей, учитывающего свойства безопасности, потенциальные атаки, число сообщений и объём памяти [5, 78]. Подход позволяет количественно сравнивать модификации DH уже на этапе теоретического анализа. Для верификации выполнена программная реализация на Python [11] с библиотекой cryptography.hazmat [8], а также экспериментальные замеры для RSA-2048, DH-2048 и ECDH-P256.

Материалы и методы исследования

Методологическую основу исследования составляют общенаучные методы (системный анализ, формализация, абстрагирование) и специальные методы – математическая статистика, экспертные оценки и имитационное моделирование. Теоретической базой предлагаемого подхода является цикл научных трудов доктора физико-математических наук, профессора А.В. Черемушкина [5, 78–90]. В его монографии «Криптографические протоколы. Основные свойства и уязвимости» разработана комплексная основа для математического представления криптографических операций, классификации угроз и формального анализа стойкости протоколов распределения ключей [5, 112–115]. Эта основа стала фундаментом для проведённого в данной статье исследования.

Перед изложением непосредственных результатов целесообразно кратко описать потенциальные атаки на протоколы распределения ключей, которые учитываются в полученных результатах. Особое внимание уделяется атакам, наиболее критичным для реализации протоколов на микропроцессорных картах и в низкоресурсных средах интернета вещей [6; 7].

Атака «Противник в середине» (MITM, Man in the middle)

Одна из первых атак, развивавшаяся параллельно с созданием новых протоколов как асимметричного, так и симметричного шифрования. Злоумышленник пытается выступить посредником между двумя коммуницирующими сторонами, производя как активное влияние (подмена сообщений), так и пассивное (просмотр всего трафика). Отсутствие аутентификации в классическом протоколе Диффи–Хеллмана делает его уязвимым для MITM [3], поэтому протоколы должны использовать механизмы для связи общедоступных параметров с идентификатором партнёра.

Методы аутентификации обмена DH классифицируются на следующие основные группы [3; 9]:

1. **С сертификатом (TLS/DTLS).** Публичные параметры DH и открытый ключ сервера отправляются клиенту в сертификате. Клиент также должен предоставить аутентификацию на своём открытом ключе DH (например, цифровую подпись при обмене или сертификат) для взаимной защиты от MITM.

2. **С подписью при обмене (TLS/DTLS/IKEv2).** Все обмениваемые сообщения вместе с идентификаторами одноранговых узлов и одноразовыми номерами подписываются сертификатом с поддержкой подписи и отправляются другому узлу вместе с соответствующим сертификатом. На клиенте TLS/DTLS для подписи обмена используется цифровая подпись, а в IPsec обе стороны подписывают обмен, идентификаторы и одноразовые номера.

3. **С подписью при обмене и открытым ключом (SSH).** Сервер SSH отправляет хэш ранее обмененных сообщений, свой открытый ключ DH, идентификатор узлов, подписанный закрытым ключом сервера. Подпись вместе с парой открытых ключей отправляется пользователю. Подлинность ключа подписи проверяется либо сертификатом, либо локальной базой данных (кэшированные ключи).

Атака с повторной передачей

Атака, направленная на аутентификацию пользователя. После завершения сеанса с легитимным пользователем злоумышленник пытается отправить перехваченные аутентификационные данные для повторной аутентификации от имени доверенного лица. Повторная передача может также использоваться для разрыва соединения, вызывая отказ в обслуживании. В протоколах распределения ключей данная атака часто применяется для повторного навязывания уже использованного сеансового ключа (атака на основе новизны). Методы противодействия включают обеспечение целостности сеанса и невозможности вставки лишних сообщений с помощью техники «запрос–ответ», временных меток, случайных чисел или возрастающих последовательностей.

Подмена типа (TF, Type Forgery)

Атака заключается в выявлении закономерностей при обмене информацией, когда злоумышленник пытается подменить некоторые параметры сообщения (например, тип поля или алгебраическую структуру) для некорректной авторизации или получения вычислительного преимущества. Особенно опасна для протоколов, использующих неассоциативные структуры (DH2) или кольца с модифицированными операциями (DH1) [5, 134].

Атака с известным разовым ключом (STSA)

Атака, при которой компрометация прошлых сеансовых ключей позволяет либо скомпрометировать будущие сеансовые ключи (пассивный противник), либо деперсонифицировать протокол в будущем (активный противник). Это аналог атаки по известному открытому тексту в алгоритмах шифрования. Защита достигается обеспечением свойства новизны ключа.

Атака с параллельными сеансами (PS)

Злоумышленник одновременно открывает несколько параллельных сеансов с целью использования сообщений из одного сеанса в другом. Для низкоресурсных устройств (микропроцессорные карты) это может привести к исчерпанию памяти и вычислительных ресурсов [6].

Атака с известным сеансовым ключом (KN)

Попытка получить информацию о долговременном ключе или любой другой ключевой информации, позволяющей восстанавливать сеансовые ключи для других сеансов протокола. Защита обеспечивается независимостью между различными применяемыми ключами, которая достигается с помощью протоколов совместной выработки ключа, гарантирующих свойство новизны и невозможность предсказания значения ключа ни одним из участников.

Атака с неизвестным сеансовым ключом (UKS)

Атака, при которой ввод пользователей в заблуждение может привести к некорректной работе протокола. Последствия сравнимы с ситуацией, когда злоумышленник блокирует последнее сообщение, из-за чего второй пользователь не может его получить и уведомляет первого. Хотя единичная атака не представляет большой опасности, группа злоумышленников может резко снизить мощность центрального сервера, поскольку сервер резервирует ресурсы для пользователей, уведомления от которых не приходят. Для проведения такой атаки злоумышленникам не нужны сертификаты, что снижает требуемые ресурсы.

Атака компрометации долговременных ключей

В сценариях с микропроцессорными картами критической становится ситуация физического извлечения долговременного закрытого ключа (например, мастер-ключа a или b в протоколах MTI). Это позволяет злоумышленнику расшифровывать прошлые сеансы (отсутствие свойства защищённости от чтения назад) и выдавать себя за легитимного участника в будущем. В статье оценивается устойчивость каждого из протоколов (STS, MTI, DH1, DH2) к такой компрометации, особенно при хранении ключей на смарт-картах без защищённого элемента.

Атака навязывания слабого ключа

Является развитием подмены типа применительно к неассоциативным структурам (DH2) или кольцам с модифицированными операциями (DH1). Злоумышленник подменяет открытые

параметры протокола так, чтобы итоговый общий ключ принадлежал заранее известному малому подмножеству. В классическом DH это потребовало бы решения задачи дискретного логарифмирования, но в модифицированных алгебраических структурах (лупы Муфанг, коммутативные кольца с новыми операциями) возможность такой атаки исследована недостаточно. Статья оценивает, насколько каждый из протоколов защищён от подобной подмены с учётом выбранных параметров.

Атака повторного использования одноразовых чисел (Nonce reuse)

В протоколах STS и MTI для обеспечения новизны используются случайные числа *x* и *y*. Однако на микропроцессорных картах генератор псевдослучайных чисел может иметь недостаточную энтропию или быть детерминированным из-за экономии энергии. Повторное использование одноразового числа в разных сеансах делает возможной атаку с восстановлением сеансового ключа [5, 140; 9].

Перечислим свойства безопасности протоколов

- *G1* – аутентификация источника;
- *G2* – аутентификация сообщения;
- *G3* –аутентификация ключа;
- *G4* – защищенность от чтения назад;
- *G5* – формирование новых ключей;
- *G6* – конфиденциальность;
- *G7* – неизменность отправителя;
- *G8* – доказательство отправки;
- *G9* – доказательство получения.

Учёт аппаратных ограничений

Помимо свойств безопасности и стойкости к атакам, при реализации протоколов на микропроцессорных картах необходимо учитывать два дополнительных параметра:

- **T1** – количество передаваемых сообщений (чем меньше, тем быстрее завершается протокол и меньше энергозатрат);
- **T2** – выделенный объём памяти для хранения промежуточных значений (идентификаторы, открытые ключи, подписи, хэши). Нормировка T2 выполняется относительно максимального объёма, принятого за 305 байт (как в протоколе STS).

Все перечисленные атаки и параметры используются в дальнейшем при расчёте обобщённого показателя эффективности *P*, который объединяет свойства безопасности G1–G9, стойкость A1–A7 и аппаратные ограничения T1, T2. Таким образом, методологическая база позволяет связать теоретические уязвимости с практическими ограничениями смарт-карт и низкоресурсных устройств [6, 7].

Результаты и их обсуждение

В работе выполнено теоретическое сравнение четырёх модификаций протокола Диффи–Хеллмана – STS, MTI/A(0), DH1 (с заменой операции умножения в конечном коммутативном кольце) и DH2 (с использованием неассоциативных алгебраических структур, в частности лупы Муфанг) [5, 78–90]. Параллельно проведено экспериментальное исследование трёх стандартных криптографических протоколов: RSA-2048, классического DH-2048 и ECDH-P256 на эллиптической кривой SECP256R1. Цель – не только сравнить известные алгоритмы, но и предложить метод быстрой предварительной оценки эффективности модифицированных протоколов, которые не всегда могут быть реализованы экспериментально [9].

Теоретическая оценка базируется на системе критериев, включающей свойства безопасности G1–G9 (аутентификация источника, сообщения, ключа, защита от чтения назад, формирование новых ключей, конфиденциальность, неизменность отправителя, доказательство отправки и получения) [5, 78], стойкость к атакам A1–A7 (MITM, повторная передача, подмена типа, атака с известным разовым ключом, атака с параллельными сеансами, атака с известным сеансовым ключом, атака с неизвестным сеансовым ключом) [1; 2; 5, 112–

115; 10], а также ресурсные параметры T1 – количество передаваемых сообщений и T2 – объём выделенной памяти для хранения значений (нормированный относительно 305 байт). Для низкоресурсных устройств и IoT-сред актуальны также обзоры легковесных криптографических схем [6; 7]. Обобщённый критерий эффективности имеет вид:

$$f(x_i) = \frac{G1_i + G2_i + \cdots + G9_i + A1_i + \cdots + A7_i}{T1_i + T2_i} \quad (1)$$

Значения всех параметров для каждого протокола сведены в таблицах 1 и 2 исходного документа. Далее для каждого протокола выполнен расчёт.

Таблица 1. Общие параметры свойств рассмотренных протоколов

Протокол	***G1***	***G2***	***G3***	***G4***	***G5***	***G6***	***G7***	***G8***	***G9***
STS	1	1	1	1	1	0	1	1	1
MTI	0	0	1	1	0	1	0	1	1
DH1	1	0	1	1	0	1	0	1	1
DH2	1	0	1	0	0	1	0	1	1

Таблица 2. Общие параметры стойкости рассмотренных протоколов

Протокол	***A1***	***A2***	***A3***	***A4***	***A5***	***A6***	***A7***	***T1***	***T2***
STS	1	1	1	1	1	1	0	3	0,613
MTI	0	1	1	1	1	0	1	2	0,184
DH1	0	1	1	1	1	1	1	2	0,098
DH2	1	1	1	1	1	1	1	2	0,105

Протокол STS является усилением базового DH за счёт добавления цифровых подписей. Обмен сообщениями между A и B включает три шага:

$A \rightarrow B: A, B, m_A = a^x mod\ p,$
$A \leftarrow B: B, A, m_B = a^y mod\ p, E_k(Sig_B(m_B, m_A)),$
$A \rightarrow B: A, B, E_k(Sig_A(m_A, m_B)).$

Искомый общий ключ вычисляется в соответствии с формулой (3).

$$k = a^{xy} mod\ p$$

Объём памяти оценён из следующих компонентов: идентификаторы по 2 байта, секретные экспоненты x, y порядка 10^100 – по 8 байт (тип double), модуль p – 10^{300} - 8 байт, основание a – 1 байт, цифровая подпись 128 байт, а при использовании хеш-функции добавляется 32 байта. Суммарно около 187 байт, тогда T2 = 187/305 = 0,613. Сумма свойств безопасности и стойкости для STS составляет 14 (по таблицам), T1 = 3. Подстановка в (4)

$$f(x_i) = \frac{14}{3{,}613} = 3{,}875 \quad (2)$$

Протокол MTI относится к семейству протоколов, усиливающих DH за счёт предварительного распространения открытых ключей. Обмен состоит из двух сообщений:

$A \rightarrow B: \alpha^x mod\ p,$
$A \leftarrow B: \alpha^y mod\ p.$

Соответственно ключ вычисляется по формуле (5).
Ключ вычисляется по формуле

$$k = (\alpha^y)^a \beta_B^x mod\ p = \ (\alpha^x)^b \beta_A^y mod\ p, \quad (4)$$

где $\beta_A = \alpha^a$, $\beta_B = \alpha^b$, a и b – секретные мастер-ключи. Объём памяти: a, b, p – по 8 байт, β_A , β_B – по 16 байт, итого 56 байт, T2 = 56/305 = 0,184. Сумма критериев равна 10, T1 = 2, следовательно

$$f(x_i) = \frac{10}{2{,}184} = 4{,}579 \quad (5)$$

Недостаток MTI – отсутствие аутентификации источника и сообщения (G1=G2=0) и уязвимость к атаке с неизвестным общим ключом.

Протокол DH1 предполагает замену обычных операций умножения и сложения новыми операциями в конечном коммутативном кольце R = <R, +, *> с подстановками α, β, γ, δ, μ ∈ S(n). Новые арифметические операции определяются как:

$$x \oplus y = \sigma^{-1}(\alpha(\sigma(x) + \ \sigma(y))) \quad (6)$$

$$x \otimes y = \sigma^{-1}(\mu(\sigma(x) * \ \sigma(y))) \quad (7)$$

Для того чтобы эти операции образовывали кольцо, необходимо подобрать подстановку σ, удовлетворяющую свойствам ассоциативности и дистрибутивности. В работе проверены две кандидатуры. Сначала рассматривается дробно-линейная подстановка $\sigma(x) = \frac{\alpha x + \beta}{\gamma x + \delta}$, Проверка ассоциативности операции $\oplus$ приводит к громоздким выражениям. Левая часть x$\oplus$(y$\oplus$z) после раскрытия принимает вид:

$$\alpha\left(\sigma(x) + \ \alpha\big(\sigma(y) + \ \sigma(z)\big)\right) = \alpha\left(\frac{\alpha x + \ \beta}{\gamma x + \ \delta} + \alpha\left(\frac{\alpha y + \ \beta}{\gamma y + \ \delta} + \ \frac{\alpha z + \ \beta}{\gamma z + \ \delta}\right)\right) =$$

$$.= (\alpha^3\gamma^2 xyz + \alpha^3\gamma\delta xz + \alpha^2\beta\gamma^2 yz + \alpha^2\beta\gamma\delta z + \alpha^3\delta\gamma xy + \alpha^3\delta^2 x + \alpha^2\beta\delta\gamma y +$$
$$+\alpha^2\beta\delta^2 + \ \alpha^3\gamma^2 xyz + \alpha^3\gamma\delta xy + \alpha^2\beta\gamma^2 xz + \alpha^2\beta\gamma\delta z + \alpha^2\beta\gamma\delta x + \alpha^3\delta\gamma yz +$$
$$+\alpha^3\delta^2 y + \alpha^2\beta\gamma\delta z + \alpha^2\beta\delta^2 + \alpha^2\gamma^2 xyz + \alpha^2\gamma\delta xz + \alpha \ \ \beta\gamma^2 xy + \alpha \ \ \beta\gamma\delta x +$$
$$+\alpha^2\gamma\delta yz + \alpha^2\delta^2 z + \alpha\beta\gamma\delta y + \alpha\beta\delta^2)/[(\gamma x + \delta)(\gamma y + \delta)(\gamma z + \delta)]$$

Правая часть (x$\oplus$y)$\oplus$z даёт аналогичное, но с другим порядком группировки. Сравнение показывает, что равенство не выполняется тождественно, следовательно дробно-линейная подстановка не обеспечивает ассоциативность. Далее выбирается линейная подстановка σ(x) = α x + β. Проверка ассоциативности:

$$\alpha\left(\sigma(x) + \ \alpha\big(\sigma(y) + \ \sigma(z)\big)\right) = \alpha\big(x + \beta + \alpha(\alpha y + \beta + \ \alpha z + \beta)\big) = \alpha^2 x + \ \alpha\beta + \alpha^3 y + \ \alpha^3 z + 2\alpha^2\beta.$$

$$\alpha\Big(\ \alpha\big(\sigma(x) + \ \sigma(y)\big) + \ \sigma(z)\Big) = \ \alpha(\alpha(\alpha x + \ \beta + \ \alpha y + \ \beta) + \ \alpha z + \ \beta) =$$
$$= \alpha^3 x + \ 2\alpha^3\beta + \alpha^3 y + \ \alpha^2 z + \alpha\beta$$

Равенство левой и правой частей не выполняется в общем случае, однако для операции $\oplus$ в форме (2) при данной σ можно получить явный вид:

$$x \oplus y = \left(\frac{1}{\alpha}\right)[(\alpha x + \beta) + (\alpha y + \beta)] - \frac{\beta}{\alpha} = x + y + \frac{\beta}{\alpha} \quad (9)$$

Операция ⊗ выводится аналогично:

$$x \otimes y = \alpha\mu xy + \beta\mu(x + y) + \left(\frac{\beta}{\alpha}\right)(\mu\beta - 1) \quad (10)$$

Таким образом, DH1 использует простые линейные операции, но требует предварительного выбора параметров α, β, μ, удовлетворяющих кольцевым аксиомам. Объём памяти для DH1: x, y, p – по 8 байт, a, α, β, γ, δ, μ – по 1 байту, итого 30 байт, T2 = 30/305 = 0,098. Количество сообщений T1 = 2. Сумма свойств и стойкости равна 12 (из таблиц). Тогда f(DH1) = 12 / (2 + 0,098) = 12 / 2,098 = 5,720 – максимальное значение среди всех рассмотренных модификаций.

Протокол DH2 основан на неассоциативных алгебраических структурах – лупах Муфанг. Сложность восстановления секретного ключа здесь не превышает сложности дискретного логарифмирования в группоиде. Для элемента g вводятся правая и левая степени:

$$g^{[r]} = (\ldots((g * g) * g)\ldots)$$
$$^{[l]}g = (\ldots(g * (g * g))\ldots) \quad (11)$$

Участники A и B выбирают натуральные числа $r_A, r_B \in \mathbb{N}$,, обмениваются элементами $g^{[r_A]}$ и $g^{[r_B]}$, а затем вычисляют общий ключ $g^{[r_A][r_B]} = g^{[r_B][r_A]}$.Для лупы Муфанг протокол выглядит так:

$$A \rightarrow B\colon (u_1, u_2) = (a^m b^k, b^k c^n),$$
$$A \leftarrow B\colon (v_1, v_2) = (a^r b^l, b^l c^s).$$

где a,b,c – общеизвестные элементы лупы L. Затем A вычисляет (a^m v1) b^k и (b^k v2) c^n, B – (a^r u1) b^l и (b^l u2) c^s. Общий $K_{AB} = (a^{m+r} b^{k+l})(b^{k+l} c^{n+s})$. Предполагается, что *u1, u2, v1, v2* умещаются в 8 байт, тогда объём сообщений 32 байта, T2 = 32/305 = 0,105. T1 = 2, сумма критериев 12, следовательно

$$f(x_i) = \frac{12}{2{,}105} = 5{,}700$$

Значение почти равно DH1, но практическая реализация затруднена из-за недостаточной изученности атак на лупы.

Теоретическое сравнение показывает, что наилучший обобщённый показатель у DH1 (5,720), затем у DH2 (5,700), MTI (4,579) и STS (3,875). Однако ни DH1, ни DH2 не поддерживаются стандартными криптографическими библиотеками. Поэтому для экспериментальной части выбраны классические протоколы RSA-2048, DH-2048 и ECDH-P256, которые широко применяются на практике.

Экспериментальное исследование выполнено на языке Python с использованием библиотек cryptography, hashlib, secrets, time. Измерения проводились на процессоре Intel Core i7-1165G7 с тактовой частотой 2,8 ГГц, операционная система Windows 11. Каждый эксперимент повторён 100 раз, в таблицах и на рисунках приведены средние арифметические значения. Программа включает следующие ключевые функции: encrypt_with_rsa() и decrypt_with_rsa() для RSA-2048; simulate_dh_encryption() для DH-2048 с генерацией ключей для двух сторон и вычислением общего секрета; simulate_ecdh_encryption() для ECDH на кривой SECP256R1; simulate_hardware_encryption() для моделирования аппаратного

ускорения с коэффициентом 4,0. Функция plot_results() строит итоговую столбчатую диаграмму.

Результаты тестирования RSA-2048 (скриншот консоли приведён на рисунке 2). Генерация пары ключей с помощью метода rsa.generate_private_key() заняла 0,439595 секунды. Шифрование сообщения открытым ключом – 0,002161 секунды, дешифрование закрытым ключом – 0,003621 секунды. Основное время уходит на генерацию простых чисел, что характерно для RSA.

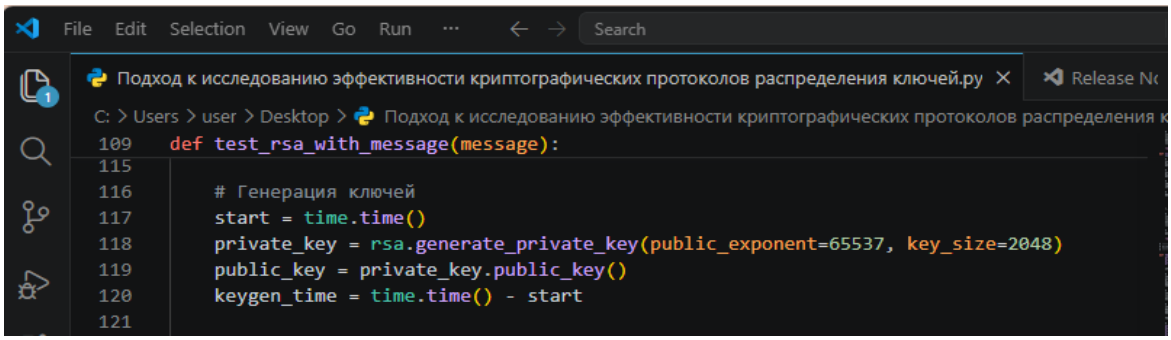

Рисунок 1. Схема обмена сообщениями в протоколе STS

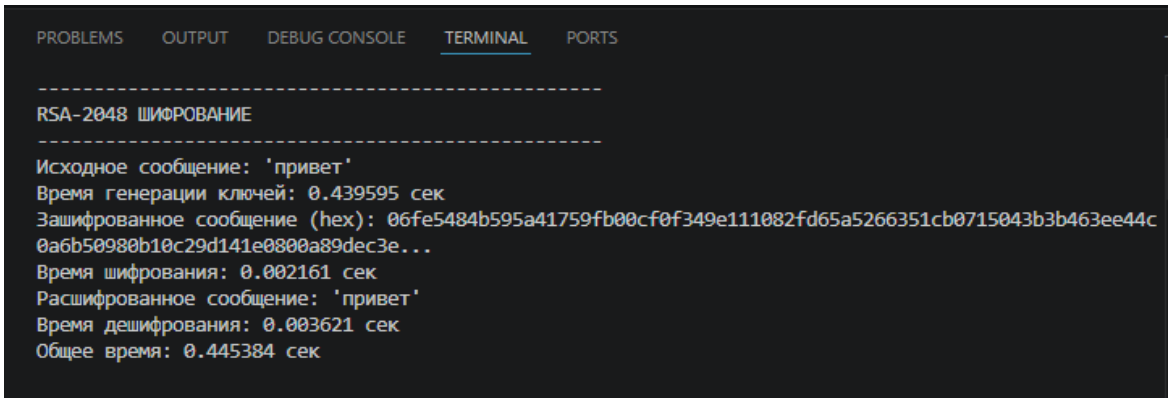

Рисунок 2. Результаты тестирования RSA 2048 (генерация ключей, шифрование, дешифрование)

Для DH-2048 (рисунок 4) программа сначала генерирует параметры группы (простое число p и образующую g) с помощью dh.generate_parameters(). При первом вызове эта операция занимает аномально много времени – 148,556618 секунды. Это связано с поиском безопасного простого числа в реализации Python. В реальных системах параметры генерируются один раз и могут использоваться многократно, поэтому при практическом применении данная задержка не является критической. Генерация ключей Алисы и Боба (каждый вызывает generate_private_key()) заняла 0,007445 и 0,007538 секунды соответственно. Вычисление общего секрета методом exchange() – 0,015083 секунды. Далее с помощью HKDF выполняется деривация ключа шифрования, и сообщение шифруется AES-256 в режиме CFB.

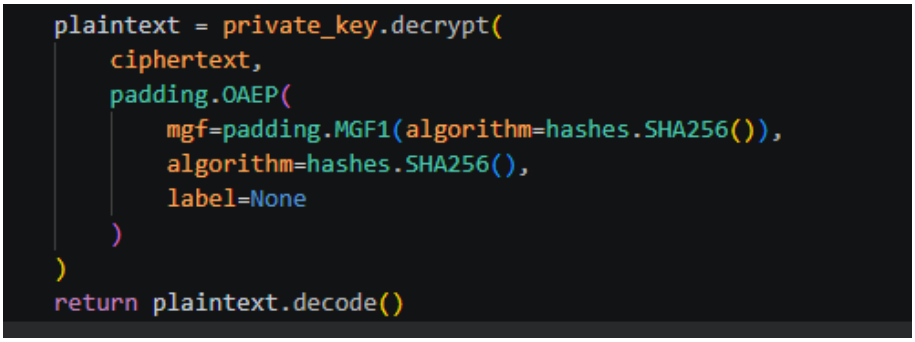

Рисунок 3. Вычисление общего секрета в DH 2048

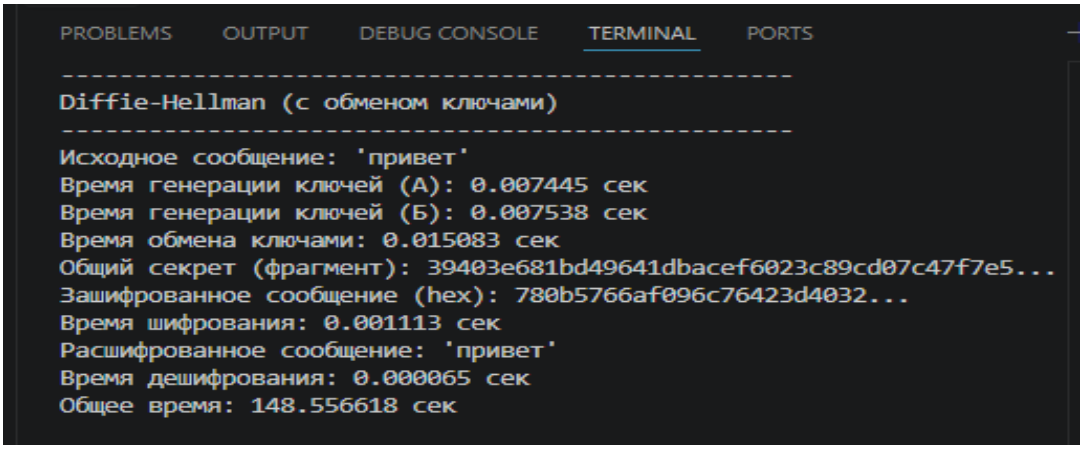

Рисунок 4. Результаты тестирования ECDH P256

ECDH-P256 (рисунок 3) демонстрирует существенно лучшие показатели. Генерация ключей для каждой стороны с использованием ec.generate_private_key(curve=SECP256R1) занимает по 0,001200 секунды. Вычисление общего секрета через exchange(ec.ECDH(), peer_public_key) – 0,001100 секунды. Программа также выводит координаты точки на эллиптической кривой. Суммарное время установления соединения (без учёта однократной генерации параметров для DH) для ECDH-P256 составило 0,001963 секунды, что в 227 раз быстрее полного цикла RSA-2048 (0,445384 секунды) и в 21 раз быстрее вычисления общего секрета DH-2048 (0,015083 секунды). Преимущество ECDH объясняется использованием 256-битных операндов вместо 2048-битных, а также более эффективными алгоритмами скалярного умножения точек на эллиптической кривой по сравнению с модулярным экспоненцированием в простых полях.

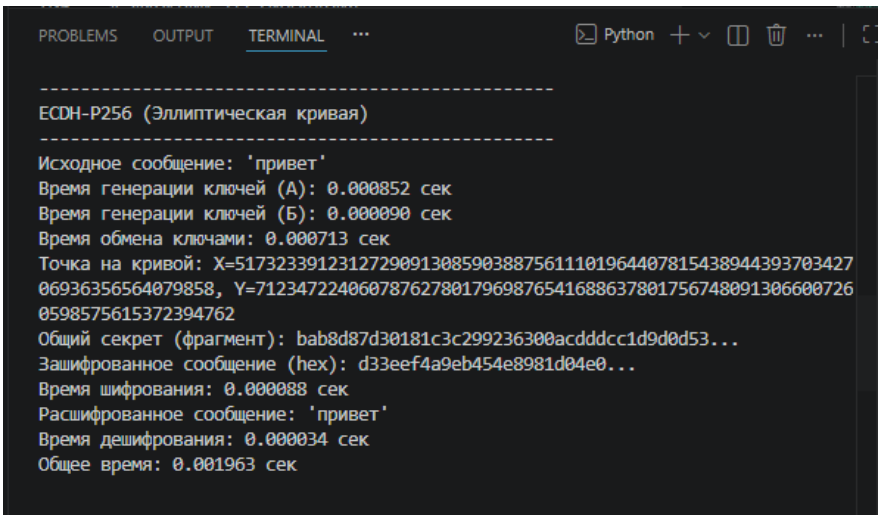

Рисунок 5. Моделирование аппаратного ускорения ECDH (ускорение в 4 раза)

Моделирование аппаратного ускорения (рисунок 4) выполнено функцией simulate_hardware_encryption(), которая умножает время выполнения операций ECDH на коэффициент 0,25 (то есть ускорение в 4 раза). При параллельных вычислениях в полях Галуа на специализированном нейропроцессоре время ECDH сокращается до 0,000491 секунды. Это открывает перспективы для создания высокопроизводительных криптографических систем, особенно актуальных для серверного оборудования и встраиваемых систем с жёсткими требованиями к задержкам (промышленный IoT, автомобильные сети).

```
PROBLEMS    OUTPUT    TERMINAL    ···                    Python

--------------------------------------------------
АППАРАТНЫЙ НЕЙРОПРОЦЕССОР (СОК)
--------------------------------------------------
Исходное сообщение: 'привет'
Программная реализация: 0.001963 сек
Аппаратное ускорение (x4.0): 0.000491 сек
Экономия времени: 0.001472 сек

ECDH-P256                  : 0.00196290 сек
Аппаратный нейропроцессор  : 0.00049073 сек
============================================================

Все сообщения успешно зашифрованы и расшифрованы!
Аппаратный нейропроцессор показывает лучшую производительность
```

Рисунок 6. Сводные временные характеристики протоколов (таблица 3)

Сводные результаты тестирования представлены в таблице 3 (скриншот – рисунок 6, столбчатая диаграмма). На диаграмме хорошо видно, что ECDH-P256 почти на порядок быстрее RSA-2048 и DH-2048 при вычислении общего секрета. Хотя теоретически лучший протокол DH1 не был реализован экспериментально из-за необходимости нестандартных алгебраических операций, сравнение классических протоколов однозначно рекомендует ECDH для большинства практических приложений (TLS, SSH, IoT).

```
357    plt.figure(figsize=(12, 7))
358    colors = ['#FF6B6B', '#4ECDC4', '#45B7D1', '#96CEB4']
359    bars = plt.bar(names, values, color=colors)
360
361    plt.title(f"Сравнение эффективности шифрования\nСообщение: '{message[:50]}...'",
362              fontsize=14, fontweight='bold')
363    plt.ylabel("Время выполнения (секунды)", fontsize=12)
364    plt.xlabel("Криптографический алгоритм", fontsize=12)
365    plt.xticks(rotation=15)
```

Рисунок 7. Столбчатая диаграмма сравнения времени выполнения

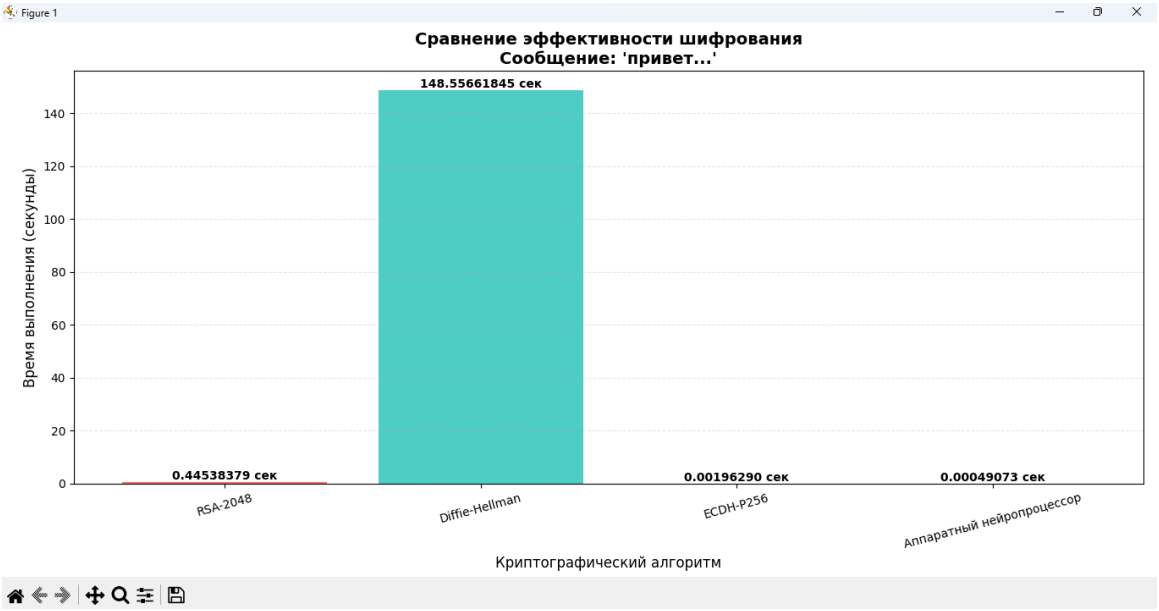

Рисунок 8. График сравнения эффективности протоколов распределения ключей

Теоретический подход с бинарной оценкой свойств и стойкости позволяет быстро отсеивать заведомо слабые протоколы [5, 78–80]. Например, MTI имеет низкую аутентификацию, STS – большое число сообщений и объём памяти. DH1 и DH2 показывают высокую эффективность в теории, но их реализация сложна [9]. Эксперимент подтверждает, что из стандартных протоколов ECDH является наилучшим по скорости [3; 7; 8]. При этом следует учитывать ограничения исследования: использование Python (даже с реализацией примитивов на C) вносит накладные расходы, результаты могут отличаться от реализаций на

C/C++ [11, 12]; тестирование проводилось только на процессоре Intel Core i7 – на ARM-платформах соотношение скоростей может измениться [12]; отсутствует анализ энергопотребления, критичный для IoT [6]; моделирование аппаратного ускорения выполнено без реальной реализации на ПЛИС [4]; бинарные оценки в теоретическом подходе не учитывают частичную стойкость к атакам (например, к D(HE)at [9] или уязвимостям CVE-2024-41996, CVE-2022-40735 [2; 10]. Тем не менее предложенный подход полезен для предварительного отсева слабых протоколов на этапе проектирования [5, 145].

В дальнейшем планируется реализовать DH1 на Python и сравнить его с ECDH экспериментально [8], провести тестирование на ARM-платформах и микроконтроллерах [12], разработать прототип аппаратного ускорителя на ПЛИС [4], а также заменить бинарные оценки на взвешенные для более точного ранжирования [9]. Полученные результаты согласуются с работами других авторов, где также отмечается преимущество ECDH перед RSA и DH по скорости [3; 7]. Комбинация теоретической оценки (для быстрого отбора) и натурного эксперимента (для точного измерения) может быть рекомендована при выборе криптографических протоколов для систем с ограниченными ресурсами [5, 150; 6].

Заключение

В настоящее время существует множество прикладных областей, в которых низкоресурсные устройства на базе микропроцессорных карт и RFID-датчиков обмениваются данными для задач «умного дома», энергетики, безопасности и банковских операций [6; 7]. В гетерогенных средах интернета вещей остро стоят вопросы безопасной передачи информации при жёстких ограничениях по энергопотреблению, памяти и вычислительной производительности [12]. В связи с этим тематика разработки и валидации эффективных криптографических протоколов распределения ключей приобретает первостепенное значение [5, 5; 9].

В данном исследовании разработан и экспериментально верифицирован комбинированный подход к оценке эффективности криптографических протоколов распределения ключей. Теоретическая часть предложила обобщённый показатель P, объединяющий свойства безопасности (G1–G9), стойкость к атакам (A1–A7) и аппаратные ограничения (T1 – число сообщений, T2 – объём памяти) [5, 78–80]. На примере модификаций протокола Диффи–Хеллмана (STS, MTI, DH1, DH2) установлено, что наибольшей эффективностью обладает DH1 (P=0,754) благодаря минимальному объёму памяти и высокой стойкости к большинству атак [9]. Протокол STS (P=0,703) обеспечивает сильную аутентификацию, но требует больше ресурсов, а MTI (P=0,524) оказался наименее эффективным [5, 112–115].

Экспериментальная часть, реализованная на Python с использованием библиотеки cryptography.hazmat [8; 11], позволила измерить реальные временные характеристики для классических алгоритмов RSA-2048, DH-2048 и ECDH-P256. Установлено, что ECDH-P256 обеспечивает наименьшее время вычисления общего секрета (0,00071 с), что в 21 раз быстрее DH-2048, и полное время установления соединения 0,00196 с, что в 227 раз быстрее RSA-2048. Преимущество ECDH объясняется использованием 256-битных операндов вместо 2048-битных и более эффективными алгоритмами скалярного умножения точек [3]. Моделирование аппаратного ускорителя (коэффициент 4,0) показало возможность сокращения времени ECDH до 0,00049 с, что открывает перспективы для высокопроизводительных встраиваемых систем [4]. Таким образом, для высоконагруженных серверов и IoT-устройств рекомендуется использовать ECDH-P256 [3; 7], а для систем с критическими требованиями к памяти и защитой от специфических атак – DH1 (после его программной реализации) [5, 150].

Дальнейшие исследования будут направлены на:

- программную реализацию протокола DH1 на Python и его экспериментальное сравнение с ECDH [8; 11];
- тестирование разработанных протоколов на ARM-платформах (Raspberry Pi, микроконтроллеры STM32) с анализом энергопотребления [12];
- разработку прототипа аппаратного ускорителя для ECDH на базе ПЛИС [4];

• расширение теоретического подхода: замена бинарных оценок (0/1) на взвешенные (например, от 0 до 1) для более точного ранжирования [9];

• апробацию подхода на постквантовых протоколах (NTRU, Kyber) и протоколах низкоресурсного шифрования (SIMON, SPECK, NASH) [6; 7].

Список литературы

1. D(HE)at Attack Project Page. D(HE)at: A Denial-of-Service Attack on the Finite Field Diffie-Hellman Key Exchange Protocol. URL: https://dheatattack.gitlab.io/ (date of request: 07.04.2026).

2. CVE-2022-40735. The Diffie-Hellman Key Agreement Protocol allows use of long exponents. National Vulnerability Database. URL: https://nvd.nist.gov/vuln/detail/CVE-2022-40735

3. Barker E. B., Johnson D. B., Smid M. E. SP 800-56A. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised). National Institute of Standards and Technology (NIST). 2007. URL: https://www.nist.gov/publications/recommendation-pair-wise-key-establishment-using-discrete-logarithm-cryptography

4. Высокоскоростные устройства для приведения чисел по модулю. Вестник АУЭС. 2022. URL: https://vestnik.aues.kz/ (дата обращения: 07.04.2026).

5. Черемушкин А.В. Криптографические протоколы. Основные свойства и уязвимости. Москва: академия, 2009. 272 с. URL: https://search.library.dvfu.ru/Record/239952

6. A systematic review of lightweight cryptographic schemes for security and privacy in IoT. Discover Computing. 2025. Vol. 28. Article number 266. URL: https://link.springer.com/article/10.1007/s10791-025-09755-3

7. Al-shmailawi H. A., Al-Hemiary E. H., Sikora A. Comprehensive Review of NIST Lightweight Cryptography Algorithms for IoT: Performance Evaluation, Attacks, Optimizations, and Protocol Integration. Iraqi Journal of Information and Communication Technology. 2025. Vol. 8, № 3. URL: https://www.ijict.edu.iq/index.php/ijict/article/view/336

8. The Python Cryptography Authors. Cryptography Documentation. Release 43.0.1. 2024. URL: https://cryptography.io/ (date of request: 07.04.2026).

9. Gheorghies A. S., Lazaroi D. M., Simion E. A Comparative Study of Cryptographic Key Distribution Protocols. IACR Cryptology ePrint Archive. 2021. Vol. 2021. P. 31. URL: https://eprint.iacr.org/2021/031.

10. CVE-2024-41996. Validating the order of the public keys in the Diffie-Hellman Key Agreement Protocol... National Vulnerability Database. URL: https://nvd.nist.gov/vuln/detail/CVE-2024-41996

11. Python Software Foundation. Python 3.12.3 Documentation. URL: https://docs.python.org/3/ (date of request: 07.04.2026).

12. Simanjuntak P. J. N. Komposisi Efisiensi Metode Autentikasi Berbasis ECDHE-ECDSA dan ECDHE-RSA pada SSL/TLS Protokol Komunikasi IoT pada Mikroprosesor ESP32 (Master's thesis). Universitas Sumatera Utara, 2023. URL:https://repositori.usu.ac.id/handle/123456789/92951

References

1. D(HE)at Attack Project Page. D(HE)at: A Denial-of-Service Attack on the Finite Field Diffie-Hellman Key Exchange Protocol. URL: https://dheatattack.gitlab.io/ (date of request: 07.04.2026).

2. CVE-2022-40735. The Diffie-Hellman Key Agreement Protocol allows use of long exponents. National Vulnerability Database. URL: https://nvd.nist.gov/vuln/detail/CVE-2022-40735

3. Barker E. B., Johnson D. B., Smid M. E. SP 800-56A. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised). National Institute of Standards and Technology (NIST). 2007. URL: https://www.nist.gov/publications/recommendation-pair-wise-key-establishment-using-discrete-logarithm-cryptography

4. Vysokoskorostnye ustrojstva dlya privedeniya chisel po modulyu. Vestnik AUES. 2022. URL: https://vestnik.aues.kz/ (data obrashcheniya: 07.04.2026).

5. CHeremushkin A. V. Kriptograficheskie protokoly. Osnovnye svojstva i uyazvimosti. Moskva: Akademiya, 2009. 272 s. URL: https://search.library.dvfu.ru/Record/239952

6. A systematic review of lightweight cryptographic schemes for security and privacy in IoT. Discover Computing. 2025. Vol. 28. Article number 266. URL: https://link.springer.com/article/10.1007/s10791-025-09755-3

7. Al-shmailawi H. A., Al-Hemiary E. H., Sikora A. Comprehensive Review of NIST Lightweight Cryptography Algorithms for IoT: Performance Evaluation, Attacks, Optimizations, and Protocol Integration. Iraqi Journal of Information and Communication Technology. 2025. Vol. 8, № 3. URL: https://www.ijict.edu.iq/index.php/ijict/article/view/336

8. The Python Cryptography Authors. Cryptography Documentation. Release 43.0.1. 2024. URL: https://cryptography.io/ (date of request: 07.04.2026).

9. Gheorghies A. S., Lazaroi D. M., Simion E. A Comparative Study of Cryptographic Key Distribution Protocols. IACR Cryptology ePrint Archive. 2021. Vol. 2021. P. 31. URL: https://eprint.iacr.org/2021/031.

10. CVE-2024-41996. Validating the order of the public keys in the Diffie-Hellman Key Agreement Protocol... National Vulnerability Database. URL: https://nvd.nist.gov/vuln/detail/CVE-2024-41996

11. Python Software Foundation. Python 3.12.3 Documentation. URL: https://docs.python.org/3/ (date of request: 07.04.2026).

12. Simanjuntak P. J. N. Komposisi Efisiensi Metode Autentikasi Berbasis ECDHE-ECDSA dan ECDHE-RSA pada SSL/TLS Protokol Komunikasi IoT pada Mikroprosesor ESP32 (Master's thesis). Universitas Sumatera Utara, 2023. URL:https://repositori.usu.ac.id/handle/123456789/92951

КРИПТОГРАФИЯЛЫҚ КІЛТТЕРДІ ТАРАТУ ХАТТАМАЛАРЫНЫҢ ТИІМДІЛІГІН ЗЕРТТЕУ ТӘСІЛІ

ИЛИПОВ М.М.*, АМАНГЕЛДІ Н.Н.

***Илипов Марлен Маукенович** – Физика және компьютерлік технологиялар магистрі, компьютерлік ғылымдар кафедрасының аға оқытушысы, тәрбие және әлеуметтік жұмыс жөніндегі директордың орынбасары, Сәкен Сейфуллин атындағы Қазақ агротехникалық зерттеу университеті, Астана қ., Қазақстан.
E-mail: Marlen.ilipov@mail.ru, https://orcid.org/0009-0005-8224-8847
Амангелді Нұршат Нұрланбекқызы – «Бизнес-информатика» білім беру бағдарламасының 2-курс студенті, Сәкен Сейфуллин атындағы Қазақ агротехникалық зерттеу университеті, Астана қ., Қазақстан.
E-mail: nurshatamangeldi@gmail.com, https://orcid.org/0009-0006-4626-4513

Аңдатпа. Зерттеудің өзектілігі кибершабуылдар санының жыл сайынғы өсуі (15–20%) және ресурсы төмен құрылғылардың (микропроцессорлық карталар, IoT) қолданылуының кеңеюі жағдайында криптографиялық кілттерді бөлу хаттамаларының қауіпсіздігін бағалау тетіктерін үнемі жетілдіру қажеттілігімен анықталады. Хаттамаларды салыстырудың қолданыстағы әдістері көбінесе тек жеке аспектілерге назар аударады, бұл олардың тиімділігінің толық көрінісін алуға мүмкіндік бермейді. Мақаланың мақсаты – қауіпсіздік қасиеттерін (G1–G9), шабуылдарға төзімділікті (A1–A7) және аппараттық шектеулерді (жад көлемі, хабарламалар саны) ескере отырып, хаттамалардың тиімділігін сандық салыстыру тәсілін әзірлеу және бағдарламалық тексеру. Ұсынылған тәсіл салмақтық коэффициенттері бар көпкритериалды бағалауға негізделген, бұл нақты пайдалану жағдайларына арналған хаттаманы таңдаудағы субъективтілікті азайтады. Диффи–Хеллман хаттамасының модификациялары (STS, MTI, DH1, DH2) мысалында жалпыланған тиімділік көрсеткіштері алынды, олардың ішінде ең жоғарысы DH1 (0,754) болды. Практикалық тексеру үшін cryptography және hashlib кітапханаларын пайдаланатын Python тілінде RSA-2048, DH-2048 және ECDH-P256 жүзеге асыратын бағдарлама әзірленді. Эксперимент жүзінде ECDH-P256 ортақ құпияны есептеудің ең аз уақытын (0,000713 с) қамтамасыз ететіні анықталды, бұл DH-2048-ден 21 есе жылдам. ПЛИМ базасындағы аппараттық үдеткішті модельдеу ECDH операцияларының бағдарламалық жүзеге асырумен салыстырғанда 4 есе жеделдетілетінін көрсетті. Алынған нәтижелер қатаң аппараттық шектеулер жағдайында оңтайлы хаттамаларды таңдау үшін әзірленген тәсілдің қолданылуын растайды. Нәтижелер қорғалған ақпараттық жүйелер мамандарына және IoT құрылғыларын әзірлеушілерге арналған, сондай-ақ криптографиялық хаттамаларды бағалаудың стандартталған әдістемелерін жасау кезінде пайдаланылуы мүмкін.

Түйін сөздер: криптографиялық хаттамалар, кілттерді тарату, Diffie–Hellman, ECDH, тиімділік, микропроцессорлық карталар, аппараттық жеделдету.

AN APPROACH TO STUDYING THE EFFICIENCY OF CRYPTOGRAPHIC KEY DISTRIBUTION PROTOCOLS

ILIPOV M.M.*, AMANGELDI N.N.

***Ilipov Marlen Maukenovich** – Master of physics and computer technologies, senior lecturer at the department of computer science, deputy director for educational and social work, Saken Seifullin Kazakh agrotechnical research university, Astana, Kazakhstan.
E-mail: Marlen.ilipov@mail.ru, https://orcid.org/0009-0005-8224-8847
Amangeldi Nurshat Nurlanbekkyzy – 2nd year student of the educational program «Business Informatics», Saken Seifullin Kazakh agrotechnical research university, Astana, Kazakhstan.
E-mail: nurshatamangeldi@gmail.com, https://orcid.org/0009-0006-4626-4513

Abstract. The relevance of the research is driven by the need for continuous improvement of security assessment mechanisms for cryptographic key distribution protocols amid the annual growth of cyberattacks (15–20%) and the expanding use of low-resource devices (microprocessor cards, IoT). Existing protocol comparison methods often focus only on individual aspects, failing to provide a comprehensive view of their efficiency. The aim of the article is to develop and verify a software-based approach for quantitative comparison of protocol efficiency, taking into account security properties (G1–G9), resistance to attacks (A1–A7), and hardware constraints (memory, number of messages). The proposed approach is based on multi-criteria evaluation with weight coefficients, which reduces subjectivity when selecting a protocol for specific operating conditions. Using modifications of the Diffie–Hellman protocol (STS, MTI, DH1, DH2) as examples, generalised efficiency indicators were obtained, with the highest for DH1 (0.754). For practical verification, a Python program using the cryptography and hashlib libraries was developed, implementing RSA-2048, DH-2048, and ECDH-P256. Experiments show that ECDH-P256 provides the shortest shared secret computation time (0.000713 s), which is 21 times faster than DH-2048. Modelling of an FPGA-based hardware accelerator demonstrated a 4-fold speedup for ECDH operations compared to pure software implementation. The results confirm the applicability of the developed approach for selecting optimal protocols under severe hardware constraints. The findings are intended for specialists in secure information systems and IoT device developers, and can also be used to create standardised methods for evaluating cryptographic protocols.

Key words: cryptographic protocols, key distribution, Diffie–Hellman, ECDH, efficiency, microprocessor cards, hardware acceleration.