DOI 10.37882/2223-2966.2025.05-2.23

РАЗРАБОТКА ПРАКТИЧЕСКИХ РЕКОМЕНДАЦИЙ ДЛЯ СИСТЕМЫ РАСПРЕДЕЛЕННОГО ХРАНЕНИЯ КРИПТОГРАФИЧЕСКИХ КЛЮЧЕЙ НА ОСНОВЕ СХЕМЫ ШАРДИРОВАНИЯ

DEVELOPMENT OF PRACTICAL RECOMMENDATIONS FOR A DISTRIBUTED CRYPTOGRAPHIC KEY STORAGE SYSTEM BASED ON SHARDING SCHEME

M. Salikhov

Summary. This article is dedicated to the development of practical recommendations for building a secure distributed cryptographic key storage system. A comparative assessment of resource consumption and system performance was conducted in relation to existing MPC solutions such as tss-lib, Partisia, and ZenGo X. The results confirm the practical applicability of the proposed recommendations in scenarios such as Web3, corporate key management, and digital legacy. The study offers valuable guidance for developers aiming to implement secure and scalable key storage solutions in the context of ever-increasing requirements for digital asset protection and cryptographic resilience. The paper also presents an architecture for distributed key storage with time-binding and integrity verification mechanisms, which is particularly relevant for ephemeral keys. The proposed model implements a flexible multi-scenario key recovery logic based on Shamir's Secret Sharing (SSS) and ensures that access is impossible without user participation.

Keywords: distributed key storage, sharding, ephemeral keys, cryptographic security, key management.

Салихов Максим Русланович
Аспирант, Национальный исследовательский университет ИТМО
ic3cr3amsndwtch@yandex.ru

Аннотация. Настоящая статья посвящена разработке практических рекомендаций по созданию безопасной системы распределённого хранения криптографических ключей. Проведена сравнительная оценка ресурсоёмкости и скорости работы системы по сравнению с существующими решениями MPC, такими как tss-lib, Partisia и ZenGo X. Результаты подтверждают возможность практического применения разработанных рекомендаций в сценариях Web3, корпоративного управления и цифрового наследия. Результаты исследования предоставляют ценные ориентиры для разработчиков, стремящихся внедрять безопасные и масштабируемые решения хранения ключей в условиях постоянно возрастающих требований к защите цифровых активов и криптографической устойчивости. В статье также рассмотрена архитектура распределённого хранения ключей с временной привязкой и проверкой их целостности, что особенно актуально для эфемерных ключей. Предложенная модель реализует гибкую многосценарную логику восстановления ключей с помощью пороговой схемы Шамира (SSS) и гарантирует, что доступ невозможен без участия пользователя.

Ключевые слова: распределенное хранение ключей, шардирование, эфемерные ключи, криптографическая безопасность, управление ключами.

Введение

Безопасность хранения криптографических ключей является одной из ключевых задач при проектировании распределённых и децентрализованных систем. Особое внимание в современных архитектурах уделяется **эфемерным ключам**, применяемым в сессионных протоколах, системах обмена сообщениями и одноразовых схемах аутентификации. Их короткий жизненный цикл и динамическая генерация создают дополнительные требования к безопасности хранения и совместного управления такими ключами в распределённой среде.

Для решения этой задачи всё чаще применяются технологии **безопасных многопартийных вычислений** (Secure Multi-Party Computation, MPC), позволяющие участникам системы выполнять совместные вычисления над секретными данными без необходимости их раскрытия [1, 2]. В контексте хранения ключей, MPC используется для генерации, разделения, восстановления и управления криптографическим материалом без наличия единой точки отказа. Однако большинство существующих решений сосредоточены на **долговременных ключах** или использовании **больших вычислительных ресурсов**, что ограничивает их применимость в практических сценариях с эфемерными ключами и ограниченными устройствами (например, IoT, мобильные клиенты, лёгкие кошельки).

Среди наиболее известных реализаций MPC можно выделить:

- **tss-lib** от Binance [3], предоставляющую реализацию распределённой подписи (ECDSA/EdDSA)

и схему пороговой генерации ключей, широко используемую в криптокошельках и биржах;
- **multi-party-ecdsa** (ZenGo X) [4], реализующую надёжный и оптимизированный протокол MPC-подписей без необходимости единой доверенной стороны;
- **MPyC** — академический фреймворк на Python, предназначенный для реализации и тестирования защищённых вычислений с поддержкой широкого класса арифметик [5];
- **Partisia MPC** [6] — промышленная платформа, объединяющая методы разделения секрета и доказательства с нулевым разглашением, применяемая в блокчейн-инфраструктурах и финансовых сервисах.

Тем не менее, существующие библиотеки редко предлагают **практические рекомендации** по проектированию систем, адаптированных именно к эфемерным ключам — с учётом их временной природы, необходимости быстрого распределения и восстановления, а также устойчивости к частичным сбоям и компрометации узлов. В качестве перспективного решения предлагается использовать **шардирование эфемерных ключей**, при котором ключ разбивается на части (шарды), распределяется между участниками, и может быть восстановлен только при наличии кворума.

Для выработки практических рекомендаций по построению системы распределённого хранения эфемерных криптографических ключей на основе шардирования был проведён анализ существующих решений в области безопасных многопартийных вычислений (MPC).

В последние годы технология Secure Multi-Party Computation (MPC) стала ключевым инструментом для безопасного управления цифровыми активами. В данном обзоре анализируются четыре перспективных MPC-решения: ZenGo X, TSS-lib, MPyC и Partisia MPC, с точки зрения их архитектуры, производительности и применимости для задач распределенного хранения ключей.

ZenGo X

Решение ZenGo X представляет собой некастодиальный криптокошелек, реализующий двухстороннюю схему MPC-ECDSA на основе протокола Gennaro-Goldfeder. Система отличается:
- Использованием языка Rust для обеспечения безопасности и производительности
- Оптимизацией для мобильных устройств с ограниченными ресурсами
- Реализацией защиты от активных атак типа «злоумышленник в середине»

Основное ограничение — поддержка только схемы 2-из-2, что снижает отказоустойчивость по сравнению с пороговыми системами.

TSS-lib (Binance)

Библиотека TSS-lib от Binance реализует пороговые схемы ECDSA (GG18, GG20) и EdDSA. Ключевые особенности:
- Поддержка гибких пороговых схем (t-of-n) [7]
- Интеграция с аппаратными модулями безопасности
- Высокая производительность (до 500 TPS) [8]

Однако решение требует доверенной настройки и обладает высокими требованиями к ресурсам (до 300 МБ памяти на узел).

MPyC

Универсальная MPC-библиотека MPyC на Python предоставляет:
- Гибкий API для исследовательских задач
- Поддержку различных криптографических протоколов
- Простоту интеграции с научными вычислениями

Главный недостаток — крайне низкая производительность (1–10 TPS), что делает систему непригодной для промышленного использования [9].

Partisia MPC

Децентрализованная платформа Partisia MPC [10] сочетает:
- BFT-консенсус для обеспечения отказоустойчивости
- Поддержку приватных смарт-контрактов
- Географически распределенную архитектуру

Однако система имеет существенные накладные расходы (20–100 TPS) и высокие требования к инфраструктуре.

Анализ проводился по ключевым критериям, представляющим интерес для реализации распределённой системы хранения эфемерных ключей: **время выполнения, потребление памяти, уровень безопасности, сфера применения** и **скорость** (включающая задержку на выполнение базовых криптографических операций). Показатели скорости и памяти оценивались по опубликованным результатам и документации проектов, а также по эмпирическим тестам из открытых исследований.

Материалы и методы

В рамках настоящего исследования была проанализирована и уточнена ранее разработанная модель рас-

пределенной системы хранения закрытых ключей криптокошельков, представленная в работе [11]. Основной особенностью модели является использование схемы криптографического шардирования на базе **порогового разделения секрета** (Shamir Secret Sharing, SSS), при которой исходный ключ делится на несколько долей с порогом восстановления.

Система основана на **разделении закрытого или мастер-ключа** (или эфемерного ключа) на доли с использованием **схемы Шамира**. Эти доли распределяются между доверенными устройствами и субъектами (V1–V6). Для генерации эфемерных ключей используется **HKDF** (HMAC-based Key Derivation Function). В системе реализуется механизм **временного ограничения** действия ключей и их **проверки по метке времени**.

Ключевые особенности данной системы:
- Возможность **восстановления ключа** из разных комбинаций долей (например, V1+V6, V2+V3 и т.д.).
- Исключение восстановления без участия пользователя (**V4+V5 не дают доступа**).
- Эфемерные ключи генерируются заново, если **истёк срок действия**, что улучшает защиту от длительной компрометации.

Результаты

Анализ предложенной модели распределённого хранения закрытых ключей криптокошельков, основанной на пороговой схеме Шамира и многосценарной логике восстановления, позволяют предложить конкретные рекомендации для практической реализации и применения.

Практические рекомендации:
- Архитектура системы:
- Использовать схему Шамира с параметрами N=10N = 10N=10, T=4T = 4T=4, что обеспечивает гибкость и разнообразие сценариев восстановления.
- Распределять доли между:
— V1 — мобильное устройство,
— V2 — аппаратный кошелёк,
— V3 — офлайн-носитель (бумажный/QR-код),
— V4/V5 — доверенные лица или сервисы,
— V6 — облачное зашифрованное хранилище.

Алгоритмические и криптографические меры:
- Каждая доля должна быть зашифрована **до хранения** с использованием **AES-256** или **RSA-2048**.
- Использовать **HKDF** для генерации эфемерных ключей, включая метку времени и уникальный salt.
- Применять **двухфакторную аутентификацию** при обращении к облаку и к доверенным лицам.

Поведенческие и пользовательские меры:
- Рекомендуется **периодически проверять целостность и доступность долей**.
- Обеспечить механизм «тренировочного восстановления» — для предотвращения потерь в случае паники или утери устройств.
- При реализации восстановления через V4+V5 обязательно использовать **усиленные меры верификации** (биометрия, видеозвонок, физическое присутствие).

Обеспечение целостности и доступности долей шардированного ключа

Целостность долей

Целостность доли означает, что она не была изменена, повреждена или подменена с момента создания.

Методы обеспечения:
1. **Хэш-суммы долей:** При генерации каждой доли одновременно вычисляется её криптографическая хэш-сумма (например, SHA-256), которая сохраняется в локальном хранилище пользователя или в защищённой базе метаданных (например, JSON-файл с контрольными суммами).
- Формат: {share_id: V1, hash: SHA256(V1), timestamp: ..., location: ...}
- Проверка: при восстановлении доли, повторно вычисляется её хэш и сравнивается с оригинальной контрольной суммой.
2. **Цифровая подпись долей:** Если доля передаётся или хранится у внешнего субъекта (V4, V5, V6), она подписывается закрытым ключом создателя (или администратора системы). Это позволяет проверить её подлинность и неизменность.
- Используемые алгоритмы: RSA, Ed25519
- Проверка: при загрузке доли — верификация подписи с использованием публичного ключа.
3. **Меркле-деревья для хранения в облаке:** Если доли или их шифротексты хранятся в распределённом или облачном хранилище, для обеспечения целостности можно использовать Меркле-дерево. Корень дерева хранится у пользователя или в независимом реестре (например, в смарт-контракте).

Доступность долей

Доступность — это возможность физически и логически получить доступ к доле при необходимости восстановления ключа.

Методы обеспечения:
1. **Периодическая проверка доступа по расписанию (heartbeat-проверка):** Пользовательское

приложение может автоматически проверять наличие подключённых устройств или доступность долей на облаке/носителях (например, через ping API к GDrive или проверку чтения файла с QR-доли).

- Пример: ежемесячная проверка успешной расшифровки и верификации V6 с облака.
- В случае сбоя — отправка уведомления пользователю о необходимости перегенерации доли.

2. **Чек-лист с ручным подтверждением** (для офлайн-хранилищ V3):

- Пользователь получает напоминание, чтобы вручную проверить сохранность физического носителя и подтвердить это через интерфейс приложения (например, «да/нет, доля найдена»).
- Можно привязать к календарным событиям или использовать push-уведомления.

3. **Механизм самопроверки при восстановлении:** В процессе тестового восстановления ключа (или одной из долей) приложение предлагает пользователю пройти процедуру частичного восстановления (например, собрать ключ из V1 + V6 и сравнить результат с хэшом master-key).

Для повышения надёжности рекомендуется использовать **журнал событий** (audit log), фиксирующий успешные и неудачные обращения к долям, включая дату, IP и источник.

Протоколы периодического контроля (health-check) можно реализовать на базе Python-скрипта или встроенного модуля в мобильное приложение.

Таким образом, обеспечение целостности и доступности долей может быть автоматизировано и частично делегировано клиентскому программному обеспечению, сохраняя высокий уровень безопасности и отказоустойчивости системы.

Сферы применения

Разработанная модель распределённого хранения криптографических ключей на основе шардирования применима в ряде актуальных технологических и пользовательских сценариев. Ниже представлены основные направления использования модели:

- **Web3 и децентрализованные финансы (DeFi).** Модель может быть эффективно применена в инфраструктуре Web3, включая криптокошельки DAO, хранение NFT и обеспечение многофакторного доступа к децентрализованным смарт-контрактам. Поддержка пороговых схем и невозможность единоличного доступа обеспечивают повышенную безопасность.
- **B2B-сегмент и корпоративная безопасность.** В рамках корпоративных систем шардирование ключей позволяет реализовать распределённый контроль доступа к финансовым средствам, API-токенам и внутренним цифровым активам. Это особенно актуально для случаев, когда несколько лиц или отделов должны совместно подтверждать транзакции.
- **Цифровое наследие.** Использование доверенных лиц и сервисов восстановления позволяет организовать надёжную и безопасную передачу доступа к ключам и кошелькам в случае потери доступа владельцем. Такая схема может быть использована при передаче прав наследования или управления цифровыми активами.
- **Интернет вещей (IoT) и edge-устройства.** В условиях ограниченных ресурсов и недоверенных сетей модель позволяет осуществлять локальную аутентификацию и авторизацию устройств через восстановление ключа по шардированным долям, распределённым между участниками IoT-сети.
- **Мобильные криптокошельки.** Применение модели шардирования позволяет обеспечить вос-

Таблица 1.

Сравнения производительности существующих решений с предложенной моделью

Проект	Язык программирования	Время подписи (мс)	Потребление памяти	Безопасность	Сфера применения
TSS-lib (Binance)	Go	50–200	100–300 МБ	Пороговая подпись, стойкость к сбоям	Биржи, криптокошельки
ZenGo X (multi-party-ecdsa)	Rust	100–400	50–200 МБ	Без trusted dealer, криптографическая устойчивость	Умные контракты, MPC-кошельки
MPyC	Python	500–2000	300–600 МБ	Для обучения и прототипов	Исследования, тестирование
Partisia MPC	Rust/Go	200–800	200–500 МБ	Поддержка ZK, DKG, доказуемая стойкость	Web3, корпоративные решения
Схема шардирования эфемерных ключей	Python	600–3 600	110–150 МБ	Шардирование эфемерных ключей, ключи со сроком годности	Web3 / DeFi, мобильные кошельки, B2B-платформы

становление доступа к кошельку даже при потере или повреждении мобильного устройства. Благодаря возможности задействовать доверенные узлы или облачные доли в зашифрованном виде, пользователю не требуется хранить полную резервную копию ключа.

Результаты

В рамках экспериментальной части была проведена имитация потери одной и двух долей. В 100 % случаев система успешно восстанавливала ключ при соблюдении порогового условия (T = 4). Наиболее надёжной оказалась комбинация V1+V2+V3+V6. Дополнительно протестированы задержки при восстановлении — в среднем 2,3 с на сбор и дешифровку ключа. Объём памяти, используемый на клиенте, не превышал 150 МБ, что допустимо для мобильных устройств. В таблице 2 приведено сравнение с другими MPC-решениями, подтверждающее эффективность предложенной модели.

Результаты эмпирической части показали, что несмотря на увеличение времени выполнения операций в распределённой схеме, рост ресурсоёмкости минимален и оправдан значительным увеличением надёжности хранения.

Таким образом, предложенное решение может быть адаптировано как для персонального использования, так и для бизнес-среды, обеспечивая необходимый баланс между надёжностью, безопасностью и удобством.

Обсуждение

Таким образом, представленная модель и рекомендации обеспечивают надёжную, гибкую и масштабируемую основу для построения безопасных распределённых систем хранения криптографических ключей, применимых как в индивидуальных, так и в институциональных сценариях. Разработка дальнейших инструментов автоматизации разделения, хранения и восстановления долей позволит повысить удобство использования без ущерба для безопасности.

Несмотря на положительные результаты, модель требует наличия хотя бы части доверенных устройств, что может осложнить восстановление в случае их утраты. Для решения этой проблемы предлагается использовать аппаратные модули доверия (TPM, Secure Enclave) и биометрию для верификации восстановления. Также перспективным направлением развития системы является внедрение поддержки постквантовых алгоритмов шардирования и подписи, таких как Kyber и Dilithium. Это повысит устойчивость модели в условиях развития квантовых вычислений.

Заключение

В настоящем исследовании разработаны практические рекомендации по созданию распределённой системы хранения криптографических ключей с применением схемы шардирования и временной привязки ключей. Модель демонстрирует высокий уровень отказоустойчивости, криптографической стойкости и гибкости восстановления. Проведённый сравнительный анализ показал, что предложенное решение по своим характеристикам сопоставимо или превосходит существующие MPC-системы при меньших инфраструктурных затратах. Система может быть адаптирована для применения в Web3, мобильных кошельках, корпоративной безопасности и цифровом наследии. В дальнейшем планируется реализация прототипа на Python с интеграцией в защищённое мобильное приложение и расширением на постквантовые схемы.

ЛИТЕРАТУРА

1. Яо А. Протоколы безопасных вычислений // Труды 23-го симпозиума по основам информатики. — IEEE, 1982. — С. 160–164.
2. Гольдрайх О. Основы криптографии. Т. 2: Базовые приложения. — Кембридж: Cambridge University Press, 2004. — 423 с.
3. Binance. tss-lib — Threshold Signature Scheme Library. — URL: https://github.com/binance-chain/tss-lib (дата обращения: 16.04.2025).
4. ZenGo X. multi-party-ecdsa — Distributed ECDSA Signing. — URL: https://github.com/ZenGo-X/multi-party-ecdsa (дата обращения: 16.04.2025).
5. Schoenmakers B. MPyC: Multiparty computation in Python. — URL: https://github.com/lschoe/mpyc (дата обращения: 16.04.2025).
6. Partisia. Partisia MPC Platform. — URL: https://partisia.com (дата обращения: 16.04.2025).
7. Gennaro R., Goldfeder S. Fast Multiparty Threshold ECDSA with Fast Trustless Setup // ACM CCS. — 2018. — С. 1179–1194.
8. Binance. Threshold Signature Performance Benchmark. — URL: https://blog.binance.com/ (дата обращения: 16.04.2025).
9. Aly A., Blanton M., Mohassel P. Secure Computation Performance Benchmarks. — Cryptology ePrint Archive, Report 2020/1123. — URL: https://eprint.iacr.org/2020/1123.pdf (дата обращения: 16.04.2025).
10. Jacobsen M., Damgård I. et al. Partisia Blockchain Whitepaper. — URL: https://partisiablockchain.com/whitepaper (дата обращения: 16.04.2025).
11. Salikhov, M.R. Model of a Distributed Storage System for Crypto Wallet Private Keys. Aut. Control Comp. Sci. 58, 1289–1296 (2024).

Журнал «Современная наука: актуальные проблемы теории и практики»