Протокол приватности CafeApp

Offline-first архитектура для анонимного корпоративного питания

Whitepaper v2.1

Июнь 2025

Автор: Чивилев Яков Владимирович

Классификация: Публичный — Defensive Publication

Содержание

Кликните ПКМ по оглавлению и выберите «Обновить поля»

1. Аннотация

В данном whitepaper описывается протокол приватности CafeApp — бессерверная архитектура коммуникации на базе электронной почты, позволяющая двум изолированным приложениям (CafeApp для поставщиков питания и HR Client для корпоративных работодателей) управлять транзакциями корпоративного питания без обмена персональными данными сотрудников. Протокол гарантирует, что имена сотрудников никогда не покидают устройство работодателя, используя заводские UID NFC-карт в качестве единственного анонимного ключа соединения между двумя локальными базами данных. Вся коммуникация между приложениями осуществляется исключительно через стандартный email (SMTP/IMAP) с шифрованием TLS. Облачный бэкенд, общая база данных и постоянное сетевое соединение не требуются.

Ключевые слова: приватность по дизайну, минимизация данных, offline-first архитектура, анонимный идентификатор, NFC, корпоративное питание, GDPR, бессерверное мобильное приложение, протокол на базе email.

2. Введение и постановка задачи

Программы корпоративного питания традиционно требуют одной из двух архитектур:

1. Централизованная SaaS-платформа — сторонний сервер хранит идентификаторы сотрудников, истории транзакций и платежные данные. Это создаёт единую точку отказа, высокоценную мишень для атаки и потенциальную GDPR-ответственность.
2. Прямая интеграция (API) — POS-система кафе в реальном времени запрашивает данные у HR-базы компании. Это открывает имена сотрудников вендору, требует сложной API-безопасности и постоянного IT-обслуживания.

CafeApp решает эту задачу третьим способом: два полностью изолированных приложения, которые общаются асинхронно через текстовый email, используя в качестве межсистемного идентификатора только неперсональный UID NFC-карты. Имена сотрудников остаются на устройстве работодателя; записи транзакций — на устройстве вендора; единственные данные, пересекающие границу, настолько анонимны, что не требуют GDPR-согласия для вендора, но достаточны для работодателя, чтобы свести расходы.

3. Архитектура системы

3.1 Обзор компонентов

Компонент	Пакет	Роль	Сетевое взаимодействие
CafeApp	com.cafeapp	Приложение на стороне вендора (планшет/телефон/компьютер бариста). Считывает NFC-карты, записывает транзакции, отправляет отчёты.	SMTP (исходящие отчёты), IMAP (входящие заявки на регистрацию и запросы отчётов)
HR Client	com.clientapp	Приложение на стороне работодателя (телефон HR/администратора). Управляет списком сотрудников, сканирует NFC-карты, получает отчёты.	SMTP (исходящие заявки/запросы), IMAP (входящие отчёты по транзакциям)
Инфраструктура email	Любой IMAP/SMTP провайдер	Асинхронная шина сообщений между двумя приложениями.	Стандартный email с TLS-шифрованием

Критически важный инвариант: нет общего сервера, нет REST API, нет WebSocket, нет облачной базы данных, нет OAuth-провайдера. Два приложения намеренно спроектированы так, чтобы не быть способными к прямому общению в реальном времени.

3.2 Принцип разделения данных

Элемент данных	Где хранится	Куда НЕ отправляется	Обоснование
ФИО сотрудника (имя, отчество, фамилия)	HR Client (EmployeeEntity)	CafeApp	Вендор не должен знать, кто покупает еду.
Внутренний номер карты (#001, #TC-012)	CafeApp (CardEntity)	HR Client	Внутренняя нумерация вендора не нужна работодателю.
Цена транзакции (сырой построчный)	CafeApp (TransactionEntity)	HR Client (кроме агрегированных отчётов)	Через границу проходят только периодические сводки.
Email кафе	CafeApp (DataStore)	HR Client	Аутентификационные

Элемент данных	Где хранится	Куда НЕ отправляется	Обоснование
			секреты локальны для устройства.
Email-кредитеншиалы работодателя	HR Client (DataStore)	CafeApp	Аутентификационные секреты локальны для устройства.
Email для отчётов компании	CafeApp (CompanyEntity)	HR Client	Работодателю не нужно знать внутреннюю маршрутизацию вендора.

Единственный идентификатор, общий для обеих систем — это заводской UID NFC-карты (например, A1B2C3D4). Это установленный на заводе, доступный только для чтения аппаратный идентификатор NFC-тега. По своей природе он не является персональным: он не содержит имени, биометрии и не связан с личностью, если работодатель явно не создаст такую связь локально.

3.3 Принцип модифицируемости (для defensive publication)

Специалисту в области бессерверных протоколов будет очевидно, что описанная архитектура не зависит от конкретных языков программирования, платформ (Android, iOS, Windows, Linux, UNIX) или типов NFC-считывателей. Раскрытые принципы могут быть реализованы с использованием любых технологий, обеспечивающих локальное хранение данных и отправку/получение текстовых сообщений через email. В частности:

- Android может быть заменён на iOS с использованием Core Data и SKPSMTPMessage;
- NFC UID может быть заменён на любой аппаратный уникальный идентификатор (например, серийный номер Bluetooth-метки, UID RFID-карты стандарта MIFARE);
- Локальные базы данных SQLite могут быть заменены на Realm, ObjectBox или даже файлы CSV.

• Референсная реализация на Android (Kotlin + SQLite) может быть перенесена на Windows (C# / .NET / WPF / Electron), Linux (Python / Qt / GTK) или UNIX-системы (macOS, FreeBSD, BSD) с использованием любой реляционной базы данных или локального хранилища.

• HR Client может быть реализован как десктопное приложение для Windows, Linux или macOS с использованием внешнего USB-NFC-считывателя (например, ACR122U) или встроенного NFC (для Mac с Apple Silicon и macOS 13+).

• Протокол IMAP/SMTP поддерживается стандартными библиотеками на всех платформах (JavaMail, Python imaplib/smtplib, .NET MailKit, Node.js Nodemailer), что делает перенос тривиальным.

4. Email-протокол

Все сообщения между приложениями— текстовые письма. Все текстовые поля передаются в кодировке UTF-8. Шестнадцатеричные UID передаются в нижнем регистре (строчные буквы a-f). Специалисту в области email-инженерии очевидно, что могут использоваться альтернативные кодировки (например, CP-1251), однако UTF-8 является предпочтительной для универсальности.

Определены три типа сообщений.

4.1 COMPANY_REG — Заявка на регистрацию

Направление: HR Client → CafeApp

Назначение: Работодатель запрашивает статус партнёра-вендора или обновляет список авторизованных NFC-карт.

Формат темы: COMPANY_REG | <companyName>

Формат тела письма:

```
Company registration request
Company: Acme Corp
Cards: 3
TAG:acme_2024
A1B2C3D4
E5F67890
12345678
```

Правила парсинга:

- Строки, начинающиеся с #, являются комментариями и игнорируются.
- Строки, начинающиеся с TAG: (без учёта регистра), определяют уникальный тег компании (fallback: companyName).
- Все оставшиеся непустые строки интерпретируются как шестнадцатеричные UID NFC-карт.
- CafeApp сохраняет заявку как PendingRegistrationEntity в ожидании ручного одобрения баристой.

Гарантия приватности: Не передаются имена сотрудников, телефонные номера, внутренние ID сотрудников. Передаются только UID физических карт.

4.2 REPORT_REQ — Запрос отчёта

Направление: HR Client → CafeApp

Назначение: Работодатель запрашивает отчёт по транзакциям за конкретный расчётный период.

Формат темы: REPORT_REQ | <companyTag> | <yyyy-MM-dd> | <yyyy-MM-dd>

Пример: REPORT_REQ | acme-corp | 2025-06-01 | 2025-06-30

Формат тела: Текст на естественном языке отправителя (не машинопарсится).

Обработка: CafeApp парсит тему, разбивая по |, извлекает companyTag и диапазон дат, генерирует TextReport и отправляет его на fromEmail запроса.

Гарантия приватности: Запрос не содержит никаких данных сотрудников. Только тег компании и расчётный период.

4.3 CAFE_REPORT — Отчёт по транзакциям

Направление: CafeApp → HR Client

Назначение: Вендор отправляет периодическую сводку всех транзакций по данной компании.

Формат темы: <companyTag> | <periodLabel>

Пример: acme-corp | 1 Jun -- 30 Jun 2025

Тело письма содержит две секции: (а) человекочитаемая ASCII-таблица для прямого чтения, и (б) машинопарсимый блок, ограниченный ===CAFE_REPORT=== и ===END===.

```
===CAFE_REPORT===
COMPANY:Acme Corp
TAG:acme-corp
PERIOD_LABEL:1 Jun -- 30 Jun 2025
GENERATED:2025-06-13
CURRENCY:EUR
===TX===

UID|DATE|TIME|PRICE
A1B2C3D4|2025-06-01|09:15|3.50
E5F67890|2025-06-02|12:30|4.00
===END===
TOTAL:7.50
COUNT:2
```

Правила парсинга:

- Блок ===TX=== содержит одну транзакцию на строку.

- Поля разделены символом | (pipe).
- Порядок колонок: UID, DATE (yyyy-MM-dd), TIME (HH:mm), PRICE (десятичное, две цифры).
- HR Client парсит этот блок, сохраняет каждую строку в ReportRowEntity и разрешает UID в локальную EmployeeEntity по cardUid.

Десятичный разделитель — точка (.). В качестве альтернативы могут использоваться разделители , или ; с соответствующим указанием в заголовке SEPARATOR:;. Это очевидная модификация для локализации.

Гарантия приватности:

- Отчёт не содержит имён сотрудников.
- Отчёт не содержит внутренних номеров вендора (#001, #TC-012).
- Отчёт не содержит первичных ключей транзакций или DB ID из CafeApp.
- Единственный межсистемный идентификатор — NFC UID, который работодатель разрешает в имя через свою локальную базу.

5. Схемы локальных баз данных

Специалисту в области баз данных будет очевидно, что типы полей могут варьироваться (например, price как INTEGER в центах вместо TEXT, timestamp как UNIX-время). Приведённые схемы являются минимальными и могут быть расширены полями sync_status, retry_count и т.п. без изменения сущности протокола.

5.1 База данных CafeApp (вендор) — cafe_db

Сущность	Ключевые поля	Примечания
CompanyEntity	name, reportEmail, uniqueTag, isActive	Одна запись на партнёрскую компанию.
CardEntity	uid, companyId, referenceNumber, isActive	referenceNumber — автоматически генерируемый последовательный код (например, #001), показываемый бариста вместо UID.
TransactionEntity	cardId, companyId, price, timestamp, includedInReport	Ссылки на CardEntity и CompanyEntity. Нет данных сотрудников.
ReportScheduleEntity	companyId, periodFromDay/Month, periodToDay/Month, sendOffsetDays, isEnabled	Определяет автоматическую ежемесячную генерацию отчётов.
SentReportEntity	companyId, periodLabel, sentAt, isManual, totalAmount, transactionCount	Журнал аудита всех отправленных отчётов.
PendingRegistrationEntity	companyName, fromEmail, uniqueTag, cardUidsJson, receivedAt	Входящие заявки COMPANY_REG, ожидающие одобрения.

5.2 База данных HR Client (работодатель) — client_db

Сущность	Ключевые поля	Примечания
EmployeeEntity	firstName, middleName, lastName, cardUid, companyId, isSynced, registeredAt	Связывает анонимный cardUid с личностью. Это единственное место, где имя связано с UID.
ReportEntity	periodLabel, receivedAt, totalAmount, isParsed, sourceEmail, currencyCode	Заголовок каждого полученного отчёта.

Сущность	Ключевые поля	Примечания
ReportRowEntity	reportId, cardUid, date, time, price	Отдельные строки транзакций из парсинга отчёта. Разрешается в EmployeeEntity локально при отображении.

6. Анализ безопасности и приватности

6.1 Модель угроз

Угроза	Митигация
Вендор узнаёт личности сотрудников	Невозможно по архитектуре. В базе данных CafeApp нет поля name и нет таблицы employee. Единственные входящие данные от работодателя — список UID.
Работодатель узнаёт внутреннюю нумерацию транзакций вендора	Невозможно по архитектуре. referenceNumber никогда не отправляется в отчётах.
Перехват email	Митигация через TLS 1.2/1.3 на SMTP и IMAP. Сам протокол не шифрует тело письма сверх транспортного уровня, потому что payload не содержит персональных данных.
Взлом сервера email-провайдера	Не применимо в части персональных данных — UID не являются PII. Однако метаданные (компания, суммы) могут быть скомпрометированы.
Кража устройства (CafeApp)	Вор получает доступ к журналам транзакций и UID, но UID на этом устройстве не связаны с именами. Имена отсутствуют.
Кража устройства (HR Client)	Вор получает соответствие имя↔UID. Митигация через стандартное шифрование устройства Android и контроль доступа на уровне приложения.
Роговая регистрация	Митигация через очередь PendingRegistrationEntity: все заявки COMPANY_REG требуют ручного одобрения баристы до привязки UID к компании.
Replay-атака на COMPANY_REG	Митигация через дубликат-гард: CompanyRepository.getByUniqueTag() проверяет существующие теги перед созданием новых компаний.

6.2 Соответствие GDPR / защите данных

По GDPR вендор (кафе) выступает в роли обработчика или совместного контроллера только анонимных данных, потому что:

1. UID сам по себе не является персональными данными, если не комбинируется с локальной таблицей работодателя.

2. Вендор никогда не владеет этой таблицей.

3. Следовательно, вендор не обрабатывает персональные данные сотрудников работодателя.

Работодатель остаётся единственным контроллером имён сотрудников. Это разделение явное и технически принудительное, а не только политическое.

6.3 Очевидные модификации для усиления приватности

Описанный протокол может быть усилен следующими дополнительными мерами, которые специалист по безопасности признал бы очевидными комбинациями раскрытых принципов:

- Шифрование тела CAFE_REPORT с использованием предварительно распределённого симметричного ключа (например, AES-GCM) — при этом UID остаются зашифрованными, но работодатель расшифровывает их локально.
- Хеширование UID на стороне кафе перед отправкой (например, SHA-256) с последующим сопоставлением работодателем — это разрывает прямую связь, но требует передачи соли.
- Использование отдельного email-ящика для каждой пары (кафе, компания) вместо общего — тривиальное масштабирование, не требующее изобретательного уровня.

7. Свойства и инварианты протокола

Следующие инварианты гарантируются протоколом и верифицируются кодом:

1. Уникальность UID: NFC UID — единственный межсистемный идентификатор. Ни одно другое поле не разделяется.
2. Отсутствие имени: Строки firstName, lastName, employeeName или любое человекочитаемое имя никогда не появляются в email, отправленном любым из приложений.
3. Отсутствие referenceNumber: referenceNumber, присвоенный CafeApp, никогда не появляется в исходящих письмах.
4. Локальное разрешение: Работодатель разрешает UID в имена исключительно через локальный поиск EmployeeDao.
5. Отсутствие центрального состояния: Нет базы данных, кэша или брокера, хранящего одновременно и имена, и цены транзакций.
6. Транспорт только по email: Единственный сетевой протокол между приложениями — IMAP/SMTP. Нет HTTP, gRPC, WebSocket или Bluetooth.
7. Человекопроверяемый payload: Все машинопарсимые блоки — plain text внутри человекочитаемого email. Нетехнический пользователь может прочитать точные данные, пересекающие границу.
8. Детерминированная генерация Device Signature: Все устройства генерируют подпись по одному и тому же алгоритму (HMAC-SHA256 с фиксированным секретом приложения), и специалисту очевидно, что возможны альтернативные алгоритмы (HMAC-SHA3, BLAKE2) при сохранении свойства детерминированности и устойчивости к подделке.
9. Отсутствие скрытых каналов: Протокол не использует никаких иных сетевых взаимодействий, кроме SMTP/IMAP. Очевидной модификацией было бы добавление резервного канала (SMS, Bluetooth) без изменения основного потока данных — это не является изобретением.

8. Референсная реализация

Референсная реализация — проект CafeApp Android, состоящий из двух Gradle-модулей:

- app — приложение вендора CafeApp (com.cafeapp)
- client_app — приложение работодателя HR Client (com.clientapp)

Ключевые исходные файлы:

```
app/src/main/java/com/cafeapp/data/local/entities/Entities.kt — схема базы данных CafeApp
client_app/src/main/java/com/clientapp/data/local/entities/Entities.kt — схема базы данных HR Client
app/src/main/java/com/cafeapp/email/TextReportGenerator.kt — протокол генерации отчётов
client_app/src/main/java/com/clientapp/email/TextReportParser.kt — протокол парсинга отчётов
app/src/main/java/com/cafeapp/email/ImapRegistrationChecker.kt — приём COMPANY_REG
app/src/main/java/com/cafeapp/email/ImapReportRequestChecker.kt — приём REPORT_REQ
client_app/src/main/java/com/clientapp/ui/screens/register/RegisterScreen.kt — отправка COMPANY_REG
client_app/src/main/java/com/clientapp/ui/screens/reports/ReportsScreen.kt — отправка REPORT_REQ и локальное разрешение имён
```

Хотя референсная реализация выполнена на Android (Kotlin + Gradle), протокол полностью независим от платформы. Идентичная логика может быть реализована на iOS (Swift + Core Data), Windows (C# / .NET + SQLite), Linux (Python + PostgreSQL) или macOS (Swift + Core Data). Единственные требования к платформе: возможность отправки и получения email через SMTP/IMAP, локальное хранилище данных и доступ к UID NFC-карты (или альтернативному аппаратному идентификатору).

9. Prior art и defensive publication

9.1 Явное указание на очевидность комбинаций

Специалисту в области построения бессерверных систем обмена данными будет очевидно, что описанные ниже элементы могут быть комбинированы с другими известными технологиями без изобретательного уровня:

- Device Signature (HMAC на основе Android ID) очевидным образом может быть заменён на подпись на основе MAC-адреса сетевого интерфейса, серийного номера устройства или UUID, сохранённого в защищённом хранилище.
- IMAP-фильтрация по BodyTerm может быть заменена на фильтрацию на стороне клиента (после загрузки всех писем) или с использованием пользовательских IMAP-флагов (если провайдер поддерживает).
- Типы сообщений (COMPANY_REG, REPORT_REQ, CAFE_REPORT) могут быть расширены дополнительными типами (например, CARD_SYNC_REQ, ERROR_REPORT) без изменения основного протокола — это очевидный шаг развития.
- Передача только UID в открытом виде может быть заменена на передачу HMAC(UID, per-company salt) — специалисту по анонимизации данных очевидно, что это усиливает приватность без изменения архитектуры.

9.2 Способы публикации

Данный документ публикуется как defensive publication через сервисы IP.com (идентификатор публикации: [будет присвоен]) и arXiv.org (идентификатор: [будет присвоен]), а также на официальном сайте CafeApp с временным штампом OpenTimestamp. Дата фиксации: июнь 2025.

10. Заключение

Протокол приватности CafeApp демонстрирует, что приватность по дизайну не требует продвинутой криптографии, доверенных сред выполнения или блокчейна. Он требует архитектурной дисциплины: строгой минимизации данных, чёткого разделения ответственности и коммуникационного протокола, проверяемого по построению. Делая NFC UID единственным мостом между двумя иначе изолированными устройствами, система гарантирует, что имена сотрудников никогда не попадают в домен вендора, при этом сохраняя полную финансовую сверку для работодателя.

По вопросам, предложениям интеграции или запросам на реализацию обращайтесь к команде разработки CafeApp.

11. Масштабирование и Device Signature (v2.0)

11.1 Проблема масштабирования N × M

Первоначальная архитектура v1.0 предполагала однозначное соответствие «одно кафе ↔ одна компания». При масштабировании возникают проблемы:

1. Несколько CafeApp (филиалы) используют один email-ящик — каждый видит все письма, включая чужие отчёты.
2. Несколько HR Client (менеджеры) работают с одной компанией — отчёты дублируются или теряются.
3. Автоматическая отправка отчётов по расписанию и автоматическая синхронизация карт без запроса требуют изоляции потоков.

11.2 Концепция Device Signature

Device Signature (DS) — криптографическая подпись каждого устройства, завязанная на Android ID и секрет приложения.

Алгоритм генерации (псевдокод для однозначного воспроизведения):

```
function generateDeviceSignature(androidId: String, appSecret: String): String {
  // appSecret для CafeApp: "cafe_app_v2_secret_2025"
  // appSecret для HR Client: "hr_client_v2_secret_2025"
  val key = appSecret.toByteArray(UTF-8)
  val message = androidId.lowercase().toByteArray(UTF-8)
  val hmac = HMAC_SHA256(key, message) // 32 байта
  val truncated = hmac.copyOf(8) // берём первые 8 байтов
  return truncated.joinToString("") { "%02x".format(it) } // 16 hex символов
}
```

Специалисту очевидно, что усечение до 8 байт (64 бит) является достаточным для предотвращения коллизий в данной модели угроз (максимум 10^4 устройств). Для больших систем может быть выбрано усечение до 12 или 16 байт без изменения логики.

Альтернативные идентификаторы устройства:

- Settings.Secure.ANDROID_ID может быть заменён на AdvertisingIdClient.getAdvertisingId() (Google Advertising ID) или уникальный идентификатор, генерируемый при первом запуске и сохраняемый в SharedPreferences;
- Для iOS аналогом является identifierForVendor.

• Для Windows аналогом является Machine GUID (из реестра HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid) или серийный номер системного диска/материнской платы.

• Для Linux аналогом является /etc/machine-id (systemd) или D-Bus UUID (/var/lib/dbus/machine-id).

• Для UNIX-систем (macOS, FreeBSD, Solaris) аналогом является IOPlatformUUID (macOS) или hostid (FreeBSD).

Форматы полей:

Поле	Описание	Формат	Пример
FROM_DEVICE	Android ID отправителя	16 hex chars	a3f7b2c1d9e4f5a6
FROM_SIG	HMAC-SHA256 подпись отправителя	16 hex chars	7d232defd8ca7450
TO_DEVICE	Android ID получателя	16 hex chars	c7a3f9e2b8c1d4f6
TO_SIG	HMAC-SHA256 подпись получателя	16 hex chars	a7b3e2c1d9f5b8a0

11.3 Протокол рукопожатия (4 шага)

Шаг 1: COMPANY_REG (HR → Cafe) — содержит FROM_DEVICE и FROM_SIG. CafeApp проверяет подпись и сохраняет в PendingRegistrationEntity.

Шаг 2: COMPANY_APPROVED (Cafe → HR) — содержит FROM_DEVICE, FROM_SIG, TO_DEVICE, TO_SIG. HR Client проверяет TO_DEVICE == свой ID и TO_SIG, сохраняет пару в ConnectedCafe.

Шаг 3: CAFE_REPORT (авто, Cafe → HR) — содержит FROM_DEVICE, FROM_SIG, TO_DEVICE, TO_SIG. HR Client фильтрует по TO_DEVICE + TO_SIG.

Шаг 4: COMPANY_REG update (авто, HR → Cafe) — содержит TO_DEVICE и TO_SIG. CafeApp фильтрует по TO_DEVICE + TO_SIG, обновляет карты без повторного одобрения.

Примечание: При отсутствии ответа на COMPANY_REG в течение 7 дней, HR Client может повторно отправить сообщение. Максимальное количество повторов — 3. Экспоненциальная задержка — очевидная модификация.

11.4 Изменения в базе данных

Новые сущности в CafeApp:

```
CREATE TABLE connected_device_entity (
    id TEXT PRIMARY KEY,
    company_id TEXT NOT NULL,
    hr_device_id TEXT NOT NULL,
    hr_device_sig TEXT NOT NULL,
    is_active INTEGER DEFAULT 1,
    approved_at INTEGER,
    FOREIGN KEY (company_id) REFERENCES company_entity(id)
);
```

Новые сущности в HR Client:

```
CREATE TABLE connected_cafe_entity (
    id TEXT PRIMARY KEY,
    company_id TEXT NOT NULL,
    cafe_device_id TEXT NOT NULL,
    cafe_device_sig TEXT NOT NULL,
    last_report_at INTEGER,
    is_active INTEGER DEFAULT 1,
    FOREIGN KEY (company_id) REFERENCES company_entity(id)
);
```

11.5 Фильтрация IMAP

Каждый CafeApp при проверке IMAP использует AND-комбинацию:

```
SearchTerm filter = new AndTerm(
    new SubjectTerm("COMPANY_REG"),
    new BodyTerm("TO_DEVICE:" + myDeviceId),
    new BodyTerm("TO_SIG:" + mySignature),
    new FlagTerm(new Flags(Flags.Flag.SEEN), false)
);
```

Если IMAP-сервер не поддерживает поиск по телу (например, некоторые старые реализации Courier), устройство должно загрузить все непрочитанные письма и выполнить фильтрацию локально. Этот fallback-механизм очевиден для любого инженера.

Пример проверки подписи локально:

```
fun isMessageForMe(message: MimeMessage, myDeviceId: String, mySig: String):
Boolean {
    val body = message.getContent() as String
    return body.contains("TO_DEVICE:$myDeviceId") && body.contains("TO_SIG:$mySig")
}
```

Преимущества: фильтрация на уровне сервера, каждый CafeApp загружает только свои письма. Проверка TO_SIG гарантирует, что подделка TO_DEVICE невозможна без секрета.

11.6 Сценарии масштабирования

1:N — одна компания, много филиалов: HR Client отправляет 3 COMPANY_REG (по одному на каждый филиал), каждый с TO_DEVICE конкретного CafeApp.

N:M — сеть кафе и холдинг: каждый CafeApp имеет N записей ConnectedDevice, каждый HR Client имеет M записей ConnectedCafe.

11.7 Безопасность v2.0

Специалисту в области криптографии очевидно, что HMAC-SHA256 может быть заменён на любой стойкий алгоритм аутентификации сообщений (CMAC, Poly1305). Выбор HMAC-SHA256 обусловлен его распространённостью и наличием в Android API.

Новые миграции:

- CafeApp-1 не видит отчёты CafeApp-2 — IMAP-фильтрация по TO_DEVICE + TO_SIG.
- Rogue HR Client не может отправить COMPANY_REG от чужого имени — без знания AppSecret и Android ID подпись неверна.
- Replay COMPANY_REG не работает — TO_DEVICE + TO_SIG уникальны для пары.
- Механизм отзыва (Revocation) — COMPANY_REVOKED с TO_DEVICE + TO_SIG удаляет ConnectedDevice.

Угроза компрометации AppSecret (извлечение из APK) митигируется использованием разных секретов для CafeApp и HR Client, а также возможностью удалённой смены секрета через обновление приложения. Более сильная митигация (например, использование аппаратного TEE) не требуется в рамках данной модели угроз.

11.8 Сравнение v1.0 и v2.0

Характеристика	v1.0	v2.0
Уникальность отправителя	Нет	Да — Device ID + Signature
Авто-отчёты без запроса	Нет	Да — по расписанию с TO_DEVICE
Авто-обновление карт	Нет	Да — для известных TO_DEVICE
N филиалов, 1 компания	Невозможно	Возможно через ConnectedCafe
M HR-менеджеров	Невозможно	Возможно через ConnectedDevice
Человекочитаемость	Да	Да — подпись в отдельных полях
Криптография	Нет	HMAC-SHA256, 64 бита

12. Приложение A: Полный пример сообщения CAFE_REPORT с Device Signature

Ниже приведён пример реального письма для иллюстрации формата:

```
From: cafe@example.com
To: hr@example.com
Subject: acme-corp | 1 Jun -- 30 Jun 2025

Dear HR department,

Please find the report for June 2025 attached in plain text.

===CAFE_REPORT===
COMPANY:Acme Corp
TAG:acme-corp
PERIOD_LABEL:1 Jun -- 30 Jun 2025
GENERATED:2025-06-13
CURRENCY:EUR
FROM_DEVICE:a3f7b2c1d9e4f5a6
FROM_SIG:7d232defd8ca7450
TO_DEVICE:c7a3f9e2b8c1d4f6
TO_SIG:a7b3e2c1d9f5b8a0
===TX===

UID|DATE|TIME|PRICE
A1B2C3D4|2025-06-01|09:15|3.50
E5F67890|2025-06-02|12:30|4.00
===END===
TOTAL:7.50
COUNT:2

Best regards,
CafeApp
```

Специалисту очевидно, что порядок полей внутри ===CAFE_REPORT=== может быть произвольным (парсер должен быть устойчив к перестановке).

13. Приложение B: Обработка ошибок и крайних случаев

Хотя whitepaper фокусируется на типовом потоке, специалист в области разработки надёжных систем без труда дополнит протокол следующими очевидными механизмами:

Ситуация	Ожидаемое поведение
Непарсимый блок в CAFE_REPORT	Письмо помечается как has_errors, человекочитаемая часть сохраняется, уведомление администратору
Дубликат COMPANY_REG от того же HR Client	Игнорируется, если pending-запись уже существует
TO_DEVICE или TO_SIG не совпадают	Письмо молча игнорируется (не скачивается, если фильтрация на сервере)
Истёкший IMAP-сеанс	Автоматический перелогин с экспоненциальной задержкой
Потеря интернета при отправке отчёта	Письмо сохраняется в исходящую очередь и отправляется при восстановлении соединения
Некорректный UID (не 16 hex)	Строка пропускается, логируется ошибка
Отсутствует блок ===TX===	Отчёт считается пустым, TOTAL и COUNT из заголовка используются как есть

Все перечисленные механизмы являются стандартной практикой и не требуют изобретательства.

Приложение C: Технические требования к оборудованию

Ниже приведены минимальные технические требования к устройствам и NFC-считывателям для развёртывания CafeApp и HR Client на различных платформах. Требования являются консервативными; протокол будет работать и на менее производительном оборудовании при условии поддержки IMAP/SMTP и доступа к NFC UID.

C.1 Требования к вычислительным устройствам

Компонент	Платформа	Минимальные характеристики	Сеть	Примечания
CafeApp (вендор)	Android 8.0+, iOS 13+, Windows 10+, Linux (Ubuntu 20.04+), UNIX (macOS 12+, FreeBSD)	2 GB RAM, 100 MB свободного места, экран 7" (рекомендуется сенсорный)	Wi-Fi или Ethernet, доступ в интернет (SMTP/IMAP с TLS 1.2+)	Может работать на планшете, POS-терминале или десктопе. Для Windows/Linux требуется внешний NFC-считыватель.
HR Client (работодатель)	Android 8.0+, iOS 13+, Windows 10+, Linux, macOS 12+	2 GB RAM, 50 MB свободного места, экран 5" (рекомендуется)	Wi-Fi, Ethernet или мобильный интернет (SMTP/IMAP с TLS 1.2+)	Может быть мобильным приложением или десктоп-админкой. NFC нужен только для первичной регистрации карт.
Email-сервер	Любой IMAP/SMTP провайдер	Не применимо (сторонний сервис)	TLS 1.2 или 1.3 обязательны	Gmail, Yandex, Outlook, корпоративный Exchange, самостоятельный Postfix/Dovecot.

C.2 Требования к NFC-считывателям и картам

Тип считывателя	Совместимость	Требования к UID	Рекомендуемые модели /

			интерфейсы
Встроенный NFC (планшет/телефон)	Android 8.0+ с NFC, iOS 13+ с NFC (CoreNFC для чтения UID)	UID должен быть доступен в raw-формате (hex, 4–10 байт)	Samsung Galaxy Tab, iPad (ограниченно), Google Pixel, Xiaomi Pad
USB NFC Reader (внешний)	Windows 10+, Linux (kernel 3.0+), macOS 10.15+	Передача UID через PC/SC или проприетарный драйвер; длина UID 4–10 байт (hex)	ACR122U, ACR1252U, ACR1281U, HID Omnikey 5022, Identiv uTrust 3700 F
Bluetooth NFC Reader	Android 8.0+, iOS 13+, Windows 10+, Linux, macOS	Передача UID по BLE в raw-формате (hex); поддержка Serial Port Profile (SPP) или BLE GATT	ACR1255U-J1, TimeAttendance TA-BT, шлюзы BLE→USB
MIFARE Classic / DESFire	Все перечисленные считыватели	UID (4 байт для MIFARE Classic, 7 байт для DESFire) — только UID, без чтения секторов/ключей	Карты MIFARE Classic 1K/4K, MIFARE DESFire EV1/EV2, ISO 14443 Type A/B
NFC Forum Type 1–5 теги	Все перечисленные считыватели	UID (4–10 байт) в hex-формате; допускается перезаписываемый UID, если он стабилен в рамках жизни карты	NTAG213/215/216, MIFARE Ultralight, смарт-метки NFC

C.3 Требования к программному окружению и библиотекам

Компонент	Минимальные требования	Рекомендуемые решения	Платформы
SMTP-клиент	TLS 1.2+, AUTH LOGIN/PLAIN или OAuth2	JavaMail (Android), MailKit (.NET), Python smtplib, Node.js Nodemailer, Swift Mailcore	Все
IMAP-клиент	TLS 1.2+, IDLE или опрос с интервалом ≤ 15 мин	JavaMail (Android), MailKit (.NET), Python imaplib, Node.js imap, Swift Mailcore	Все

Локальная база данных	SQL-совместимое хранилище, ACID-транзакции	SQLite (все платформы), Core Data (iOS/macOS), Room (Android), PostgreSQL (Windows/Linux сервер)	Все
NFC / PC/SC API	Доступ к UID в raw hex	Android NFC API, iOS CoreNFC, Windows WinSCard / .NET PC/SC, Linux libnfc / pcsc-lite, macOS CryptoTokenKit	Зависит от платформы
HMAC-SHA256 (Device Signature)	Реализация в стандартной библиотеке или BouncyCastle/OpenSSL	javax.crypto.Mac (Android/Java), CommonCrypto (iOS), .NET System.Security.Cryptography, Python hmac	Все

Примечание: При использовании десктопных платформ (Windows, Linux, macOS) с внешним USB-NFC-считывателем необходимо установить соответствующие драйверы PC/SC (например, libnfc + pcsc-lite для Linux, ACS Unified Driver для Windows). Для большинства современных Linux-дистрибутивов драйверы входят в состав пакетов pcsc-lite и libccid.

Приложение D: Список используемых источников

Ниже приведён перечень нормативных документов, стандартов, технических спецификаций и программных продуктов, упомянутых в настоящем whitepaper и использованных при проектировании протокола CafeApp.

1. ISO/IEC 14443 — Identification cards — Contactless integrated circuit cards — Proximity cards. Стандарт на NFC Type A/B, лежащий в основе чтения UID MIFARE и NTAG. URL: https://www.iso.org/standard/73598.html

2. NXP MIFARE Classic / MIFARE DESFire EV1/EV2 — Product data sheets. NXP Semiconductors, 2024. URL: https://www.nxp.com/products/rfid-nfc/mifare

3. NTAG213/215/216 — NFC Forum Type 2 Tag — Product data sheet. NXP Semiconductors, 2024. URL: https://www.nxp.com/products/rfid-nfc/ntag

4. RFC 5321 — Simple Mail Transfer Protocol (SMTP). J. Klensin. IETF, 2008. URL: https://datatracker.ietf.org/doc/html/rfc5321

5. RFC 9051 — Internet Message Access Protocol (IMAP) — Version 4rev2. A. Melnikov, B. Leiba. IETF, 2021. URL: https://datatracker.ietf.org/doc/html/rfc9051

6. RFC 2104 — HMAC: Keyed-Hashing for Message Authentication. H. Krawczyk, M. Bellare, R. Canetti. IETF, 1997. URL: https://datatracker.ietf.org/doc/html/rfc2104

7. FIPS 198-1 — The Keyed-Hash Message Authentication Code (HMAC). National Institute of Standards and Technology (NIST), 2008. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf

8. RFC 8446 — The Transport Layer Security (TLS) Protocol Version 1.3. E. Rescorla. IETF, 2018. URL: https://datatracker.ietf.org/doc/html/rfc8446

9. Regulation (EU) 2016/679 — General Data Protection Regulation (GDPR). European Parliament and Council, 2016. URL: https://gdpr.eu

10. Android Developers — NFC basics. Google. URL: https://developer.android.com/develop/connectivity/nfc/nfc

11. Apple Developer — Core NFC. Apple Inc. URL: https://developer.apple.com/documentation/corenfc

12. SQLite — In-memory / file-based relational database engine. SQLite Consortium, 2024. URL: https://www.sqlite.org

13. PC/SC Workgroup — Interoperability Specification for ICCs and Personal Computer Systems. URL: https://www.pcscworkgroup.com

14. libnfc — Open source NFC library. URL: https://github.com/nfc-tools/libnfc

15. pcsc-lite — PC/SC Smart Card Middleware for Linux. URL: https://pcsclite.apdu.fr

16. ACR122U — USB NFC Reader. Advanced Card Systems Ltd. (ACS). URL: https://www.acs.com.hk

17. JavaMail API — Oracle Corporation / Eclipse Foundation. URL: https://eclipse-ee4j.github.io/mail/

18. MailKit — Cross-platform .NET mail-client library. Jeffrey Stedfast, 2024. URL: https://github.com/jstedfast/MailKit

19. Python imaplib / smtplib — Standard Library. Python Software Foundation. URL: https://docs.python.org/3/library/imaplib.html

20. Nodemailer — Node.js email sending library. Andris Reinman, 2024. URL: https://nodemailer.com

21. OpenTimestamp — Trusted timestamping with Bitcoin. Peter Todd, 2017. URL: https://opentimestamps.org

22. Creative Commons Attribution 4.0 International (CC BY 4.0). URL: https://creativecommons.org/licenses/by/4.0

23. Android Developers — Settings.Secure.ANDROID_ID. Google. URL: https://developer.android.com/reference/android/provider/Settings.Secure#ANDROID_ID

24. Apple Developer — identifierForVendor. Apple Inc. URL: https://developer.apple.com/documentation/uikit/uidevice/1620059-identifierforvendor

25. systemd — machine-id. Freedesktop.org. URL: https://www.freedesktop.org/software/systemd/man/latest/machine-id.html

CafeApp

Приватность по дизайну

contact@cafeapp.dev