ПАК «Звезда»
Общее описание

Оглавление

Список сокращений

АРМ	Автоматизированное рабочее место
ПАК	Программно-Аппаратный Комплекс
СКЗИ	Средство Криптографической Защиты Информации
УЦ	Удостоверяющий центр
ЭБ	Элемент Безопасности
ЭБКС	Элемент Безопасности Криптосервера
ЭП	Электронная подпись
API	Application Program Interface (Программный интерфейс приложения)
APDU	Application Protocol Data Unit
A-Proxy	Программный адаптер со стороны сервера приложений для интеграции криптосервиса в криптосервер
OSI	Open Systems Interconnection basic reference model
T-Proxy	Программный адаптер со стороны транспортной системы для интеграции криптосервиса в криптосервер

1. Общее описание ПАК «Звезда»

Программно-аппаратный комплекс "ЗВЕЗДА" предназначен для обеспечения криптографической защиты данных, передаваемых между конечными устройствами и сервером приложения в сетях интернета вещей. Криптографическая защита предполагает обеспечение целостности, конфиденциальности, аутентичности при отправке данных от клиента к серверу приложений и в обратную сторону.

ПАК «Звезда» ориентирован на сети, постороенные по топологии звезда, в которых один или несколько серверов приложений получает данные мониторинга и управляет множеством конечных устройств (далее *клиентов*). Например, к таким сетям относятся:

- Промышленные сети мониторинга и управления оборудованием (IIOT)
- Сети инфраструктуры умного города
- Сети управления умным домом

ПАК «Звезда» обладает следующими свойствами:

- Используется аппаратный элемент безопасности российского производства, отвечающий требованиям СКЗИ классов КС1, КС2, КС3, НР. Криптографические ключи на стороне клиента и криптосервера хранятся в элементе безопасности. Криптографические алгоритмы на стороне клиента и криптосервера полностью выполняются внутри элемента безопасности.
- Применяет российские криптографические алгоритмы последнего поколения, в том числе блочный алгоритм ГОСТ Р34.12-2015 «Магма» для шифрования и расчета имитовставки и ГОСТ Р34.10-2012 для расчета ЭП от передаваемых данных (см.[2] [3]).
- Аппаратный элемент безопасности на стороне устройства имеет компактный размер, низкое энергопотребление, и имеет возможность отключения питания при уходе устройства в режим сна, что позволяет использовать его в конечных устройствах с низким энергопотреблением и ограниченным вычислительным ресурсом.
- Использует компактный протокол защиты передаваемых данных CRISP, специально разработанный для Интернета вещей и стандартизованный в России в 2019 году. Этот протокол не требует предварительной установки сессии (т.е каждое сообщение содержит всю информацию, необходимую для его обработки).
- Обеспечивает удаленное управление ключевой информацией, что позволяет не демонтировать устройства для регламентной замены криптографических ключей.
- Подписывает сообщения полноценной электронной подписью на стороне конечного устройства, что позволяет пользователю (провайдеру приложения) проверять целостность и подлинность любых, в том числе архивных сообщений, без какой-либо зависимости от инфраструктуры передачи данных (провайдера связи и самого ПАК «Звезда»).
- Имеет модульную структуру и легко интегрируется как в конечные устройства, так и в серверную инфраструктуру.
- Решение совместимо с различными протоколами транспортного уровня – (NBIoT, LoRa, NB-Fi, TCP и т.д.)

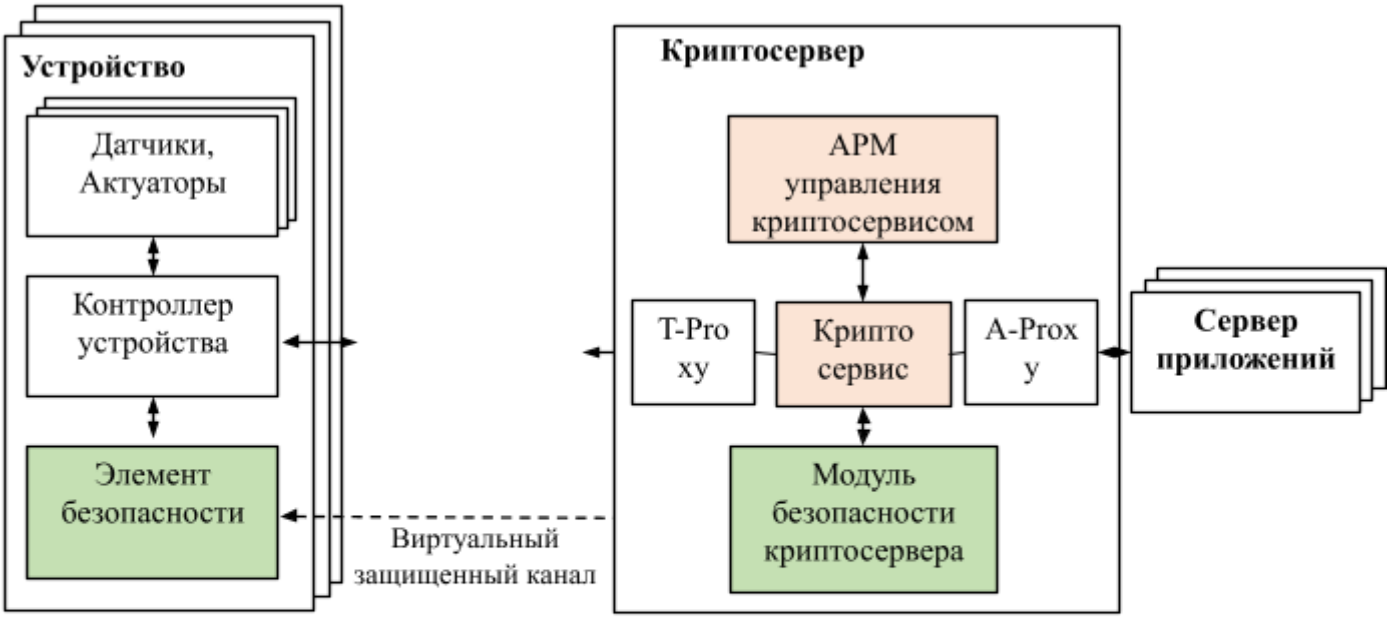
Настоящий документ содержит общее описание архитектуры компонентов ПАК «Звезда», правила взаимодействия между компонентами и встраивания их в инфраструктуру заказчика.

ПАК "ЗВЕЗДА" состоит из следующих компонентов:

- Элемент безопасности (ЭБ) конечного устройства (клиента).
- Модуль безопасности криптосервера
- Криптосервис
- АРМ управления криптосервисом[1]

На Рисунок 1 изображена архитектурная схема компонентов ПАК «Звезда».

Рисунок 1. Общая архитектура ПАК «Звезда»



Канал передачи данных

Оранжевым цветом отмечены программные компоненты ПАК «Звезда»

Зеленым цветом отмечены аппаратные компоненты ПАК «Звезда»

Белым цветом обозначены части инфраструктуры пользователя, с которыми взаимодействуют компоненты ПАК «Звезда»

Компоненты ПАК «Звезда»:

Элемент безопасности – микросхема, обеспечивающая защиту информации на стороне клиентского устройства (Подробности см. 3)

Модуль безопасности криптосервера - это кластер микросхем на плате PCI-Express (или в ином исполнении), обеспечивающий хранение ключей и выполнение криптографических операций на стороне сервера (Подробности см. 4)

Криптосервис – программа, реализующая бизнес-логику серверной компоненты ПАК «Звезда» на стороне сервера (Подробности см. 5). Для хранения ключей и проведения криптографических операций криптосервис обращается к модулю безопасности.

[1] На схеме входит в состав криптосервера, но в общем случае может располагаться на другой машине

АРМ администрирования криптосервиса - это программный компонент с GUI интерфейсом для управления работой криптосервиса. АРМ администрирования необходим для:

- Управления криптосервисом
- Удаленного управления состоянием ЭБ на подключенных клиентах

Части инфраструктуры, в которую встраивается ПАК «Звезда»:

Клиент – это конечное устройство, которое необходимо защитить. На Рисунок 1 изображена типовая схема клиента, в которую предполагается встраивание элемента безопасности. Как правило, клиент обеспечивает управление взаимодействием с физическими датчиками и актуаторами. ***Датчики*** позволяют собрать определенного рода телеметрические данные о состоянии среды, а ***актуаторы*** выполняют определенные действия согласно предписаниям сервера. Нередко одно конечное устройство совмещает в себе и датчики, и актуаторы. Как правило, датчики/актуаторы физически встроены в устройство.

В состав ***клиента*** входит:

- Элемент безопаности (ЭБ) – микросхема, встроенная в конечное устройство, и обеспечивающая криптографическую защиту на его стороне (аппаратная компонента, поставляется в составе ПАК «Звезда»).
- Управляющий контроллер для взаимодействия с ЭБ, каналом передачи данных и датчиками.
- Прочие элементы для обеспечения работоспособности устройства (источник питания, физические датчики/актуаторы и т.д.)

Криптосервер - это физический сервер, работающий 24/7, на котором установлен МБКС и запущен криптосервис

Криптосервер, в зависимости от выбранной бизнес-модели, может быть:

- встроен в инфраструктуру заказчика. В этом случае заказчику проще обеспечить контроль защиты данных на промежутке от криптосервера к серверу приложений, т.к. они находятся на одном предприятии.
- установлен в инфраструктуре провайдера связи для оказания услуги криптографической защиты как сервиса. В этом случае заказчик получает сервис криптографической защиты без необходимости устанавливать/налаживать/обслуживать криптосервис, а также без необходимости подключаться к API криптосервиса, но теряет контроль над промежутком пути данных от криптосервера к серверу приложений.

В состав сервера входят:

- Криптосервис для выполнения бизнес-логики серверной компоненты. Подробности см. 4 (программная компонента, поставляется в составе ПАК «Звезда»).
- Модуль безопасности криптосервера для безопасного хранения и использования криптографических ключей и выполнения криптографических операций (аппаратная компонента. поставляется в составе ПАК "Звезда").
- АРМ администрирования криптосервиса для управления ключами, клиентами и прочими сущностями в составе ПАК «Звезда» (программная компонента, поставляется в составе ПАК «Звезда»).
- прочие элементы для обеспечения работоспособности устройства (источник питания, периферийные устройства и т.д.)
- A-Proxy и T-Proxy для взаимодействия со средой передачи данных и сервером приложений (Подробности см. 4)

Сервер приложений - это потребитель данных от датчиков, и инициатор передачи управляющих команд датчикам. Сервер приложений получает данные от датчиков для дальнейшего сохранения, обработки и/или передачи во внешнюю среду. С точки зрения ПАК "Звезда" провайдер

приложения является конечным пунктом доставки данных, и логика его дальнейшей работы с данными выходит за рамки ПАК "Звезда".

2. Протокол взаимодействия Элемента Безопасности и криптосервиса

Для защищенного обмена данными между ЭБ клиента и криптосервисом был разработан протокол прикладного уровня CRISP-Z, являющийся модификацией индустриального протокола CRISP (см. [1]). Подробное описание протокола «Звезда» см. [4]

На стороне криптосервера протокол полностью реализован в криптосервисе. Для криптографических операций криптосервис использует модуль безопасности криптосервера (МБКС).

На стороне клиента протокол по большей части реализован в ЭБ, но часть функций по обработке ложится на контроллер усройства.

Далее в этой главе для удобства будем использовать упрощенную терминологию: ЭБ будем называть клиентом, а криптосервис – сервером.

Протокол обладает следующими свойствами:

Свойство	Техническое решение
Не требует установки сессии	Отсутствует процедура Handshake. Каждое сообщение содержит всю информацию, необходимую для его обработки (в том числе и информацию для аутентификации источника). Т.е. по сути, каждое сообщение – это мини-сессия.
Поддержка возможности подтверждения доставки сообщений в обе стороны	В протоколе предусмотрен режим синхронной сессии для передачи данных, в котором стороны обмениваются сообщениями строго последовательно.
Обеспечение целостности передаваемых данных	Каждое сообщение снабжается имитовставкой
Возможность конфиденциальной отправки данных	Поддерживаются различные алгоритмы шифрования. В заголовке сообщения есть идентификатор криптонабора, определяющего используемый алгоритм.
Возможность передачи ЭП клиентского устройства вместе с данными	В заголовке сообщения есть признак наличия ЭП в сообщении. Этот признак может быть использован в сообщениях от клиента к серверу.
Обеспечение взаимной аутентификации клиента и сервера при обмене сообщениями	Имитовставка ГОСТ 34.12-2015 (Магма), которой снабжается каждое сообщение, формируется на сессионном ключе, который выводится из уникального для каждого клиента мастер-ключа. Поскольку ключ известен только клиенту и серверу, успешная проверка имитовставки аутентифицирует отправителя.
Обеспечение защиты от повторной передачи	В заголовке каждого сообщения присутствует уникальные счетчики сообщений клиента и сервера (по 3 байта каждый). Поскольку данные заголовка включаются в имитовставку, это гарантирует защиту от переповторов при передаче в обе стороны.
Обеспечение возможности удаленного управления ключами клиента	Протокол CRISP-Z содержит служебные команды, позволяющие удаленно управлять ключами клиента. Все управляющие команды отправляются с подтверждением доставки

Протокол работает на прикладном уровне модели OSI. Предполагается, что транспортировка данных от клиента к серверу и обратно обеспечивается протоколами более низких уровней модели OSI.

Бизнес-логика работы с прикладными данными, переданными с помощью данного протокола, определяется логикой приложений и выходит за рамки данного документа.

Протокол предусматривает 2 режима работы:

Асинхронный режим – режим, в котором клиент и сервер обмениваются сообщениями независимо друг от друга. Данный режим не предусматривает подтверждения доставки и повторную отправку в случае потери сообщения. Если сообщения пришли не в том порядке, то будет обработано только сообщение, сформированное последним. Сообщение, которое было сформировано раньше, но пришло с опозданием, будет отвергнуто. Для обработки сообщения не нужно предварительно устанавливать соединение. Вся необходимая информация передается в заголовке сообщения.

Синхронный режим – режим, в котором клиент и сервер обмениваются сообщениями строго последовательно. Режим обеспечивает подтверждение доставки. После отправки каждого сообщения отправитель ждет ответного сообщения-подтверждения. Если ответное сообщение не пришло в течение некоторого времени, будет осуществлена повторная отправка. Синхронный режим используется криптосервисом для удаленного управления ключами, поскольку в этом случае необходимо подтверждение доставки управляющих команд. Также синхронный режим может быть использован для отправки пользовательских сообщений с подтверждением доставки как от устройства к серверу, так и в обратном направлении. Работу в синхронном режиме всегда инициирует клиент, и завершает криптосервис. В рамках синхронного режима может быть отправлено множество сообщений в обе стороны. Временной промежуток между началом работы протокола в синхронном режиме и завершением работы в синхронном режиме (переходом в асинхронный режим) далее будем называть синхронной сессией.

Сообщение состоит из следующих частей:

- обязательный заголовок [**16 байт**],
- опциональное поле данных [**0…480 байт**],
- обязательное поле CCS (имитовставка) [**4 байта**].

Заголовок имеет фиксированный формат для всех сообщений независимо от направления отправки и типа сообщения. Накладные расходы на протокол Crisp составляют 20 байт на сообщение (16 на заголовок, 4 на CCS).

В протоколе предусмотрено несколько криптонаборов, определяющих используемые для формирования сообщения криптографические механизмы:

Наименование	Краткое описание
MAGMA-CTR-CMAC	• шифрование ГОСТ Р34.12-2015 (режим CTR) • имитовставка ГОСТ Р34.12-2015 (режим имитовставки)
MAGMA-NULL-CMAC	• шифрования нет • имитовставка ГОСТ Р34.12-2015 (режим имитовставки)
G28147-CFB-CMAC	• шифрование ГОСТ 28147-89 (режим CFB) • имитовставка ГОСТ 28147-89 (режим имитовставки)

Протокол не налагает каких-либо ограничений на длину отправляемых сообщений.

Однако, в асинхронном режиме работы ЭБ позволяет отправить сообщение с данными длиной не более 480 байт. Отправка данных, размер которых превышает 480 байт, возможна только в рамках синхронной сессии с помощью процедуры цепочечной передачи данных (*чейнинга*). Для использования режима чейнинга данные делятся на части, и последовательно отправляются принимающей стороне. После каждого сообщения ожидается подтверждение доставки, что значительно увеличивает накладные расходы протокола.

3. Элемент Безопасности

Элемент безопасности конечного устройства – микросхема, разработанная АО «НИИМЭ», и производимая ПАО «Микрон».

Элемент безопасности поставляется в корпусе LGA-40 (см. Рисунок 2) или в виде модуля SIM-карты на ленте:

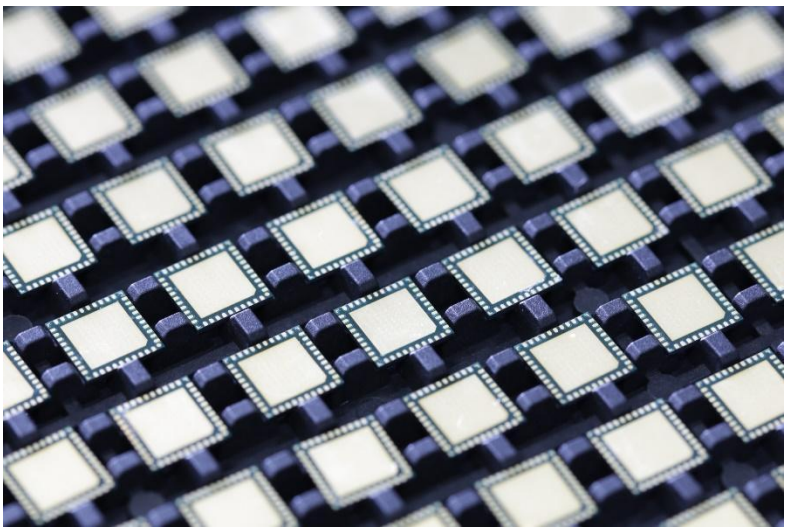


Рисунок 2. Микросхемы ЭБ

ЭБ предоставляет следующие функции для конечного устройства:

- Формирование исходящих CRISP-сообщений для отправки криптосерверу.
- Обработка входящих CRISP-сообщений от криптосервера
- Формирование ЭП от данных на стороне клиента

Администрирование ЭБ (управление ключами и настройками) обеспечивается прозрачно для контроллера конечного устройства посредством служебных CRISP-сообщений.

В асинхронном режиме работы ЭБ не хранит сессионной информации в оперативной памяти, что дает возможность отключать питание ЭБ на время ухода устройства в режим энергосбережения.

- При формировании CRISP-сообщения с использованием ЭБ могут быть использованы следующие доп. опции: Шифрование
- Подтверждение доставки
- Формирование ЭП от данных
- Режим чейнинга

Для взаимодействия с ЭБ предоставляется API уровня APDU (согласно ISO 7816-4). Описание API см. [6]

Описание порядка интеграции ЭБ в клиентское устройство см. [9]

Таблица 1. Технические характеристики ЭБ

Микроконтроллер	MIK51BC16
Корпус	LGA-40, ID1 (smart card), 2FF (SIM), 3FF (micro-SIM), 4FF (nano-SIM), MFF2 (eSIM)
Размер	см. корпус
Ток потребления	5 mA
Ресурс	10лет, 100 тыс. оп.
Производительность	100 ms/message (192 байта)
Интерфейс	UART (ISO 7816-3)

4. Модуль Безопасности Криптосервера

Модуль безопасности криптосервера (МБКС) используется криптосервисом для хранения ключей и выполнения криптографических операций с ключами.

МБКС может быть реализован следующими способами:

Тип МБКС	Применение
плата PCI-Express (см. Рисунок 3)	Плата PCI-Express предназначена для стационарных серверов с большим количеством клиентов.
чипы в корпусе LGA-40	Для компактных серверных решений в возможностью установки корпуса на плату.
смарт-карты	для компактных серверных решений, снабженных считывателем смарт-карт
сим-карты 2FF (SIM), 3FF (micro-SIM), 4FF (nano-SIM), MFF2 (eSIM)	для компактных серверных решений, снабженных холдером сим-карт

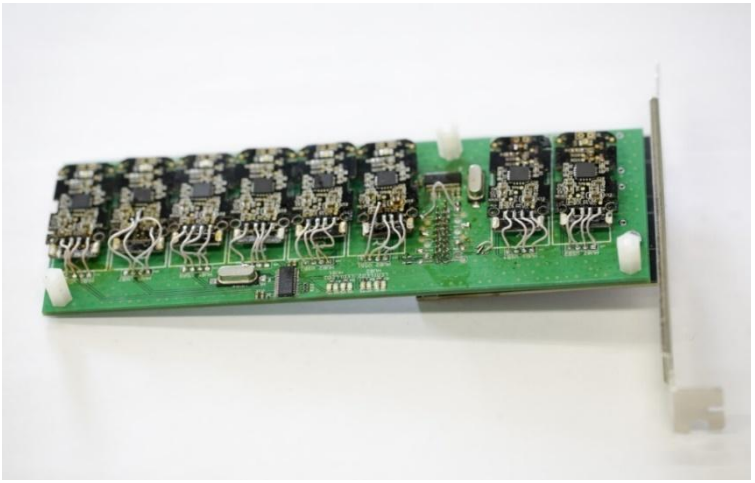


Рисунок 3. Модуль безопасности криптосервера в виде платы PCI-Express

МБКС включает в себя несколько Элементов Безопасности Криптосервиса (ЭБКС). Количество ЭБКС может быть различным. Тип МБКС PCI-Express содержит в своем составе несколько ЭБКС (см. Рисунок 3) .

При использовании типов МБКС с одной микросхемой (таких как LGA-40, смарт-карта, сим-карта), рекомендуется использовать не менее двух микросхем (ЭБКС). Несколько таких микросхем ЭБКС называются *кластер.*

Использование кластера ЭБКС дает следующие преимущества:

- Обеспечение отказоустойчивости. При использовании кластера криптосервис проводит дополнительные операции по резервному копированию определенных ключей модуля безопасности, потеря которых может привести к необходимости переперсонализации всех клиентских устройств. Тем самым, обеспечивается защита от выхода из строя одного или нескольких чипов кластера.
- Повышение производительности за счет параллельного выполнения криптографических операций

Для взаимодействия со всеми типами МБКС используется интерфейс PCSC. Каждое устройство ЭБКС в составе МБКС – отдельное PCSC-устройство. Таблица 2. Технические характеристики модуля безопасности

Интерфейс	PCI Express, ID1 (smart card), 2FF (SIM), 3FF (micro-SIM), 4FF (nano-SIM), MFF2 (eSIM)

Ресурс	10 лет
Макс. нагрузка	• PCI Express - 10 млн. сообщений. / сутки (при размере сообщения 192 байта) • ID1 (smart card), 2FF (SIM), 3FF (micro-SIM), 4FF (nano-SIM), MFF2 (eSIM) – 1.4 млн. сообщений / сутки (при размере сообщения 192 байта) на каждый ЭБКС

5. Криптосервис

Криптосервис представляет собой программу разработанную для ОС Windows в виде исполняемого файла (EXE)[2].

Криптосервис реализует следующие функции:

- Формирование исходящих сообщений для защищенной передачи прикладных данных клиенту.
- Обработка входящих сообщений от клиента: проверка целостности, расшифровка и извлечение прикладных данных
- Управление криптографическими ключами ***сервера***
- Удаленное управление криптографическими ключами ЭБ на ***клиентах***. Подписание данных клиента своей ЭП (для передачи серверу приложений)
- Интеграция клиентов в PKI (Генерация/Выгрузка запросов на сертификат/сертификатов ключей ***клиентов*** и ***сервера,*** Загрузка сертификатов ключей ***клиентов*** и ***сервера*** подписанных внешним УЦ и т.д.). Экспорт/импорт различных данных криптосервиса для передачи/копирования на другие устройства с обеспечением конфиденциальности и (опционально) целостности передачи секретных ключей

Подробное техническое описание криптосервиса (включая API) см. [5]

Криптосервис предоставляет программный интерфейс (API), посредством сокетов TCP/IP. API разделен на несколько групп:

Таблица 3. Группы интерфейсов криптосервиса

Название	Описание
RadioApi	Интерфейс для взаимодействия с клиентом. Обеспечивает шифрование/расшифрование CRISP-сообщений
AppApi	Интерфейс для взаимодействия с сервером приложений. Обеспечивает отправку/прием пользовательских данных
AdminApi	Интерфейс для администрирования. Обеспечивает управление клиентами, ключами, мониторинг и другие функции

Обращения к функциям API осуществляется согласно специально разработанному протоколу RPC (см.[8]).

На Рисунок 4 показана схема интеграции криптосервиса в физический криптосервер. Криптосервис может быть использован совместно с произвольными протоколами транспортного уровня. Для встраивания криптосервиса в цепочку обмена данными рекомендуется реализовать модули T-Proxy и A-Proxy.T-Proxy – адаптер, согласующий интерфейс транспортной системы с интерфейсом криптосервиса. В его функции также может входить:

- Защита канала взаимодействия с транспортной системой[3]
- Контроль доступа к интерфейсу RadioApi криптосервиса
- Буферизация сообщений[4].
- и т.д.

A-Proxy - адаптер, согласующий интерфейс сервера приложений с интерфейсом криптосервиса. В его функции также может входить:

- Защита канала взаимодействия с сервером приложений

[2] На последующих этапах возможна разработка аналогичного сервиса для ОС семейства UNIX

[3] Криптосервис не обеспечивает контроль доступа к своим API. Предполагается, что данный функционал при необходимости должен быть реализован в T-Proxy и A-Proxy.

[4] Криптосервис не буферизует сообщений и данных. Все принятые через RadioApi сообщения от клиента после расшифрования сразу отправляются в интерфейс AppApi, а переданные через AppApi данные после запаковки в сообщения сразу отправляются в RadioApi. Исключение составляет отправка данных в синхронном режиме. В этом случае данные помещаются в очередь до момента установления синхронной сессии с клиентом.

- Контроль доступа к интерфейсу AppApi криптосервиса
- Буферизация даных
- и т.д.

Криптосервис не допускает одновременного подключения нескольких соединений по одному и тому же типу интерфейса (т.е. максимальная конфигурация обслуживаемых соединений - один A-Proxy, один T-Proxy и один АРМ администрирования)

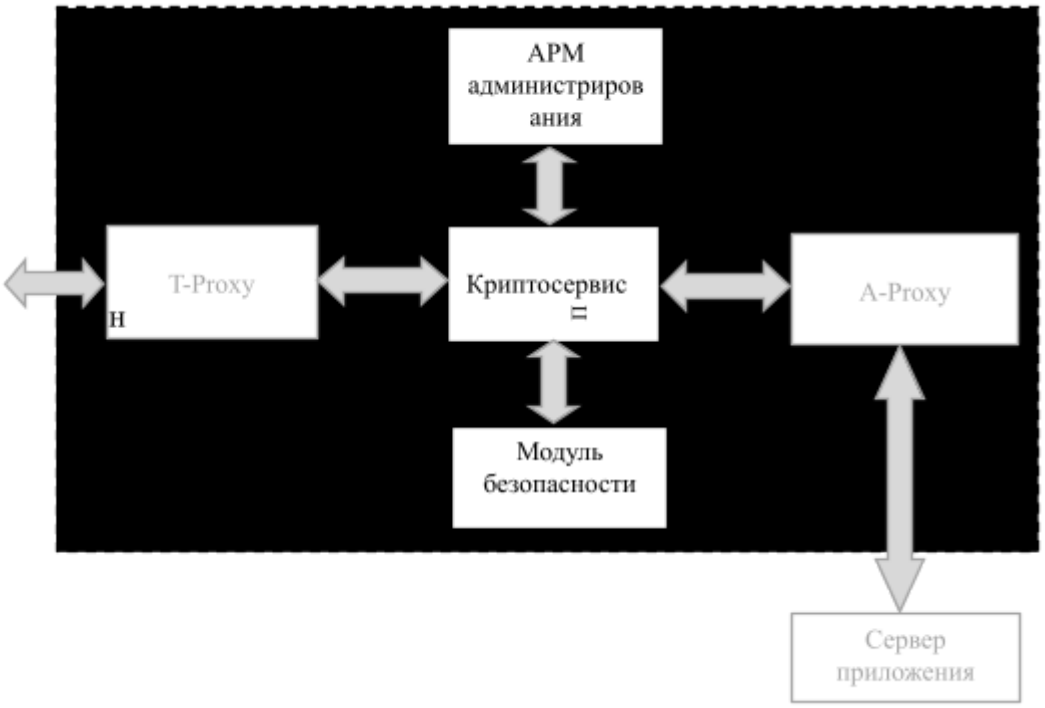


Рисунок 4. Схема встраивания криптосервиса в криптосервер

Разграничение доступа к криптографическим функциям и функциям управления ключами криптосервиса обеспечивается средствами криптосервиса.

Вопросы разграничения доступа к интерфейсам и к функциям, не связанным с криптографией, а также вопросы защиты канала между средой функционирования и криптосервисом выходят за рамки данного документа, и должны при необходимости обеспечиваться средой функционарования криптосервиса.

6. АРМ Администрирования криптосервиса

АРМ администрирования криптосервиса, далее АРМ, представляет собой программу на базе платформы .NET Framework для мониторинга и управления состоянием Криптосервиса (модуль системы ПАК «Звезда»). АРМ не расширяет возможностей Криптосервиса и позволяет делать только то, что разрешено в API Криптосервиса.

Для взаимодействия с Криптосервисом АРМ использует интерфейс AdminApi. Безопасность передаваемых данных по этому каналу связи возлагается на среду окружения и не рассматривается в рамках этого документа.

АРМ позволяет:

- Создавать, удалять и изменять ключи.
- Создавать, удалять и изменять параметры клиентов Криптосервиса.
- Создавать, удалять и изменять параметры пользователей АРМ.
- Управлять сертификатами Криптосервиса.
- Отслеживать ошибки и события Криптосервиса.
- Отслеживать нагрузку на Криптосервис.
- Изменять настройки Криптосервиса.

Подробную информацию и руководство пользователя см. [7]

7. Основные сценарии работы

В данном разделе приведены основные сценарии работы ПАК «Звезда»

7.1. Отправка данных от датчиков к серверу приложения без подтверждения доставки

1. Датчик передает контроллеру клиента телеметрические данные.
2. Контроллер обращается к криптомодулю для формирования криптографически защищенного сообщения (отправка командного APDU согласно ISO 7816-4) без установки бита синхронной сессии.
3. Криптомодуль формирует защищенное сообщение и возвращает его в ответном APDU согласно ISO 7816-4.
4. Контроллер отправляет сообщение в канал передачи данных. На этом логика на стороне клиентского устройства заканчивается, и устройство может, например, уйти в режим энергосбережения
5. Сообщение проходит через сеть, и приходит на API Proxy для транспортного протокола
6. API Proxy для для транспортного протокола через API криптосервиса отправляет сообщение на криптосервис
7. Криптосервис обращается к ЭБКС чтобы расшифровать и проверить сообщение (отправка командного APDU согласно ISO 7816-4)
8. ЭБКС расшифровывает и проверяет сообщение, и возвращает открытые данные в ответном APDU согласно ISO 7816-4.
9. Криптосервис отправляет данные в API Proxy для приложения (через интерфейс криптосервиса)
10. API Proxy для приложения отправляет данные приложению

7.2. Отправка данных от датчиков к серверу приложения с подтверждением доставки

1. Датчик передает контроллеру клиента телеметрические данные.
2. Контроллер обращается к криптомодулю для формирования криптографически защищенного сообщения (отправка командного APDU согласно ISO 7816-4) с установкой бита синхронной сессии.
3. Криптомодуль формирует защищенное сообщение и возвращает его в ответном APDU согласно ISO 7816-4.
4. Контроллер отправляет сообщение в канал передачи данных. После отправки данных клиент должен ожидать подтверждения доставки.
5. Сообщение проходит через сеть, и приходит на API Proxy для транспортного протокола
6. API Proxy для для транспортного протокола через API криптосервиса отправляет сообщение на криптосервис
7. Криптосервис обращается к ЭБКС чтобы расшифровать и проверить сообщение (отправка командного APDU согласно ISO 7816-4)
8. ЭБКС расшифровывает и проверяет сообщение, и возвращает открытые данные в ответном APDU согласно ISO 7816-4.
9. Криптосервис отправляет данные в API Proxy для приложения (через интерфейс криптосервиса)
10. API Proxy для приложения отправляет данные приложению
11. Криптосервис обращается к ЭБКС чтобы сформировать сообщение-подтверждение приема (отправка командного APDU согласно ISO 7816-4)
12. ЭБКС формирует сообщение и возвращает его в ответном APDU согласно ISO 7816-4.
13. Криптосервис отправляет сообщение-подтверждение приема в API Proxy для транспортного протокола.
14. API Proxy для транспортного протокола отправляет сообщение в канал передачи данных.

15. Сообщение проходит через сеть и приходит в контроллер клиента.
16. Контроллер обращается к ЭБ для обработки сообщения-подтверждения (отправка командного APDU согласно ISO 7816-4).
17. ЭБ проверяет подтверждение
18. Контроллер фиксирует подтверждение доставки данных (в данном сценарии предполагаетсся что криптосервис не отправляет управляющих команд, а только подтверждение. В обшем случае криптосервис мог бы отправить данные в ответ, и не завершать синхронную сессию).
19. На этом логика на стороне клиентского устройства заканчивается, и устройство может, например, уйти в режим энергосбережения

7.3. Отправка данных от сервера приложения клиенту без подтверждения доставки

1. Провайдер приложения передает данные для отправки датчику в API Proxy для приложения.
2. API Proxy для приложения через API криптосервиса отправляет данные на криптосервис.
3. Криптосервис формирует защищенное сообщение (посредством обращения к ЭБКС) и передает его в API Proxy для транспортного протокола.
4. API Proxy для транспортного протокола отправляет сообщение в канал передачи данных.
5. Сообщение проходит через сеть и приходит в контроллер клиента
6. Контроллер обращается к ЭБ для расшифровки сообщения (отправка командного APDU согласно ISO 7816-4).
7. ЭБ расшифровывает сообщение и возвращает контроллеру открытые данные в ответном APDU.
8. Контроллер выполняет бизнес-логику по обработке данных

7.4. Отправка данных от провайдера приложения датчику с подтверждением доставки

1. Провайдер приложения передает данные для отправки датчику в API Proxy для приложения.
2. API Proxy для приложения через API криптосервиса отправляет данные на криптосервис.
3. Криптосервис буферизирует данные до момента выхода клиента-адресата в эфир.
4. Криптоссервис ждет начала синхронной сессии от клиента для передачи буферизованных данных (если клиент вышел на связь, но отправил сообщение в асинхронном режиме, то криптосервис шлет запрос синхроной сессии и продолжает ожидание открытия синхронной сессии)
5. После получения от клиента синхронного сообщения (подробности см. 7.2, п 1-8), криптосервис формирует сообщение с установленным битом синхронной сессии и через API Proxy для транспортного протокола отправляет сообщение в сеть.
6. Сообщение проходит через сеть и приходит в контроллер клиента
7. Контроллер обращается к ЭБ для расшифровки сообщения (отправка командного APDU согласно ISO 7816-4).
8. ЭБ расшифровывает сообщение и возвращает контроллеру открытые данные в ответном APDU.
9. Контроллер отправляет данные датчику
10. Контроллер обращается к ЭБ для формирования сообщения-подтверждения (отправка командного APDU согласно ISO 7816-4).
11. ЭБ формирует сообщения-подтверждения и возвращает его в ответном APDU согласно ISO 7816-4.
12. Контроллер отправляет сообщение в сеть
13. Сообщение проходит через сеть и приходит на API Proxy для транспортного протокола
14. API Proxy для транспортного протокола через API криптосервиса отправляет сообщение на криптосервис
15. Криптосервис обращается к ЭБКС чтобы расшифровать и проверить сообщение (отправка командного APDU согласно ISO 7816-4)

16. ЭБКС расшифровывает и проверяет сообщение, и возвращает подтверждение в ответном APDU согласно ISO 7816-4.
17. Криптосервис фиксирует успешную доставку данных клиенту и оповещает API Proxy для приложения (через интерфейс криптосервиса)
18. Криптосервис отправляет сообщение закрытия синхронной сессии клиенту или возвращается на п-т 5 если есть еще данные для отправки.

7.5. Удаленное управление криптомодулем клиента

1. Оператор подает в криптосервис (через API криптосервиса) команду удаленного управления ЭБ клиента.
2. Криптосервис буферизирует административную команду до момента выхода клиента-адресата в эфир.
3. Криптоссервис запрашивает синхронную сессию и ждет от клиента инициации синхронной сессии для передачи команды (если клиент вышел на связь, но отправил сообщение в асинхронном режиме, то криптосервис шлет запрос синхроной сессии и продолжает ожидание открытия синхронной сессии)
4. После получения от клиента синхронного сообщения (подробности см. 7.2, п 1-8), криптосервис формирует сообщение с установленным битом синхронной сессии для передачи управляющей команды и через API Proxy для транспортного протокола отправляет сообщение в сеть.
5. API Proxy для транспортного протокола отправляет сообщение в канал передачи данных.
6. Сообщение проходит через сеть и приходит в контроллер клиента
7. Контроллер передает ЭБ административную команду (отправка командного APDU согласно ISO 7816-4).
8. ЭБ обрабатывает административную команду, производит необходимые действия и возвращает контроллеру ответное административное сообщение (в ответном APDU).
9. Контроллер отправляет ответное сообщение в канал передачи данных
10. Сообщение проходит через сеть и приходит на API Proxy для транспортного протокола
11. API Proxy для транспортного протокола через API криптосервиса отправляет сообщение на криптосервис
12. Криптосервис обращается к ЭБКС чтобы расшифровать и проверить сообщение (отправка командного APDU согласно ISO 7816-4)
13. ЭБКС расшифровывает и проверяет сообщение, и возвращает ответную административную команду в ответном APDU согласно ISO 7816-4.
14. Криптосервис обрабатывает ответную административную команду, и, в зависимости от типа административного сообщения, либо заново формирует новое административное сообщение, либо завершает операцию удаленного управления.
15. После завершения операции удаленного управления криптосервис оповещает АРМ администрирования об успешном выполнении операции.
16. Криптосервис отправляет сообщение закрытия синхронной сессии клиенту или продолжает синхронную сессии, например, если есть пользовательские данные для отправки (см. 7.4).

Список источников

[1] Протокол защищенного обмена для индустриальных систем (CRISP 1.0). Проект. Технический комитет по стандартизации «Криптографическая защита информации»

[2] ГОСТ Р 34.12 - 2015. Блочные шифры.

[3] Р 50.1.113 2016. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования

[4] ПАК «Звезда» - Протокол обмена

[5] ПАК «Звезда» - Техническое описание криптосервиса

[6] Операционная система Trust 3.30i. Руководство программиста.

[7] ПАК «Звезда» -Руководство пользователя АРМ

[8] Технология удаленного вызова процедур

[9] ПАК «Звезда» -Руководство по интеграции ЭБ

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Версия	Дата	Автор	Содержание изменения
1	01.03.2019	Сергеев В.Л.	Создание документа
2	06.05.2019	Сергеев В.Л.	Редактирование общей схемы ПАК Редактирование схем протокола Мелкие правки текста
3	20.05.2019	Сергеев В.Л.	Редактирование базовых криптографических алгоритмов
4	18.10.2019	Сергеев В.Л.	Описание режима чейнинга Описание формата сертификатов
5	13.01.2020	Сергеев В.Л.	Добавление параметра ServerQuota Запрос синхронного режима становится битом а не отдельным служебным сообщением. Описание восстановления после сбоя синхронного режима и чейнинга.
6	20.10.2020	Сергеев В.Л.	Переработка документа. Оставлено только общее описание. Все детали перенесены в другие документы.
7	07.12.2020	Сергеев В.Л.	Правки в соответствие с замечаниями.