

ViPNet PKI Client Android SDK

Руководство разработчика

Версия продукта 2.1

О документе

Документ содержит инструкцию по работе с TLS ГОСТ ViPNet PKI Client — сервисом приложения ViPNet PKI Client Android для подключения к сайтам, использующим TLS ГОСТ, из сторонних приложений.

Документ предназначен для разработчиков, которые встраивают функции приложения ViPNet PKI Client Android в свои приложения. Предполагается, что читатель этого руководства имеет представление о базовых понятиях в области криптографии и PKI, а также имеет опыт разработки приложений для устройств на базе Android на языках Java/Kotlin.

О TLS ГОСТ ViPNet PKI Client

TLS ГОСТ ViPNet PKI Client — это связанная служба ([Bound Service](#)), предоставляющая сторонним приложениям без поддержки алгоритмов ГОСТ возможность подключения к веб-ресурсам, использующим алгоритмы ГОСТ для защиты соединений (далее — ГОСТ-ресурсам). Представляет собой локальный HTTP прокси-сервер, к которому привязываются (bind) сторонние приложения. При подключении к ГОСТ-ресурсам для передачи данных по открытым каналам связи используется TLS ГОСТ, а в пределах устройства — стандартный TLS (с зарубежными алгоритмами). Стороннее приложение использует TLS с понятными для него алгоритмами. Сертификаты ГОСТ-ресурсов проверяются средствами приложения ViPNet PKI Client Android. Также можно устанавливать TLS-соединения с аутентификацией пользователя по сертификату. Открытый HTTP-трафик и трафик, зашифрованный с использованием зарубежных алгоритмов, передается без изменений.

Во время привязки (bind) сервис запускает локальный HTTP прокси-сервер, если он не был запущен ранее вручную или из другого приложения. После этого средствами [AIDL \(Android Interface Definition Language\)](#) можно получить статус прокси-сервера, номер порта и корневой сертификат ViPNet PKI Client Android.

Для работы пользователям потребуется однократно выдать разрешение на использование TLS ГОСТ ViPNet PKI Client в стороннем приложении.

Для привязки (bind) к сервису в файле манифеста стороннего приложения должны быть указаны разрешение и пакет:

```
<manifest ...>
  <uses-permission android:name="ru.infotecs.pkiclient.permission.TLS_PROXY" />
  <queries>
    <package android:name="ru.infotecs.pkiclient" />
  </queries>
  ...
</manifest>
```

Для определения интерфейса взаимодействия со сторонними приложениями сервис использует [AIDL](#). AIDL-файлы из комплекта поставки должны быть добавлены в проект стороннего приложения.

Общий сценарий взаимодействия

Сценарий взаимодействия стороннего приложения с TLS ГОСТ ViPNet PKI Client выглядит следующим образом:

- 1 **Стороннее приложение привязывается (bind)** с помощью функции `bindService`.
Если прокси-сервер не запущен, то будут выполнены необходимые начальные проверки. Это займет несколько секунд, в зависимости от характеристик устройства. Учитывайте это при выборе контекста для привязки сервиса.
- 2 Если в процессе начальных проверок обнаружены проблемы, не позволяющие использовать прокси-сервер, пользователю нужно запустить приложение ViPNet PKI Client Android и устранить их. После того, как проблемы будут решены, начальные проверки будут продолжены. Отключение и повторное подключение к сервису не требуется.
- 3 После того, как прокси-сервер перешел в рабочее состояние, необходимо [получить корневой сертификат ViPNet PKI Client Android](#).
- 4 После получения сертификата необходимо [настроить используемый HTTP-клиент](#).

Привязка к сервису (bind)



Примечание. Основные понятия и принцип см. в документации на [сайте для разработчиков Android](#).

Пример привязки к сервису:

```
var service: ITlsProxyService? = null

val listener: ITlsProxyServiceListener = object : ITlsProxyServiceListener.Stub() {
    override fun stateChanged(state: Int) {
        // Обработать изменение состояния сервера.
        // Будет гарантированно единожды вызван при успешном подключении к сервису.
        // Возможные значения:
        // STATE_ERROR - неустранимая внутренняя ошибка. Работа с сервисом невозможна.
        // STATE_NORMAL - сервер находится в нормальном рабочем состоянии.
        // STATE_STARTING - выполняются стартовые проверки.
        // STATE_INITIALIZATION_REQUIRED - не была выполнена первичная инициализация
        // ViPNet PKI Client Android.
        // STATE_ACTIVATION_REQUIRED - требуется выполнить активацию
        // ViPNet PKI Client Android.
        // STATE_INTEGRITY_ERROR - обнаружено нарушение целостности.
        // Пользователю требуется запустить <Product_name_short > и принять решение
        // о продолжении его использования.
        // STATE_PASSWORD_CHANGE_REQUIRED - требуется смена пароля пользователя.
        // STATE_ENTROPY_UPDATE_REQUIRED - требуется обновить последовательность
        // случайных чисел.
    }
}
```

```

        override fun portChanged(port: Int) {
            // Порт прокси-сервера. Будет гарантированно единожды вызван:
            // при успешном подключении к сервису;
            // при переходе прокси-сервера в рабочее состояние;
            // если прокси-сервер уже был в этом состоянии на момент подключения.
            // В дальнейшем будет вызван, если пользователь изменит порт в настройках
            // ViPNet PKI Client Android.
        }
    }

val connection: ServiceConnection = object : ServiceConnection {
    override fun onServiceConnected(name: ComponentName?, service: IBinder?) {
        service = ITlsProxyService.Stub.asInterface(service).also {
            try {
                it.addListener(listener)
            } catch (e: RemoteException) {
                // In this case the service has crashed before we could even
                // do anything with it; we can count on soon being
                // disconnected (and then reconnected if it can be restarted)
                // so there is no need to do anything here.
            }
        }
    }

    override fun onServiceDisconnected(name: ComponentName?) {
        // This is called when the connection with the service has been
        // unexpectedly disconnected -- that is, its process crashed.
        service = null
    }
}

...
val intent =
Intent("ru.infotecs.pkiclient.GostTlsProxyService.BIND").setPackage("ru.infotecs.pkiclient")
context.bindService(intent, connection, Context.BIND_AUTO_CREATE)

```

Получение корневого сертификата ViPNet PKI Client Android

Пример получения сертификата:

```

fun getCaCertificate (): X509Certificate? {
    val caBytes = try {
        service?.caCertificate
    } catch (e: RemoteException) {
        null
    }
    if (caBytes == null) {
        return null
    }

    val cf: CertificateFactory = CertificateFactory.getInstance("X.509")
    val caInput: InputStream = ByteArrayInputStream(caBytes)
    return caInput.use {
        cf.generateCertificate(it) as X509Certificate
    }
}

```

Метод `getCaCertificate` можно вызывать только после того, как прокси-сервер перешел в состояние `STATE_NORMAL`, в противном случае метод может вернуть `null`.

Настройка HTTP-клиента

Для настройки HTTP-клиента нужно:

- установить доверие к корневому сертификату ViPNet PKI Client Android;



Примечание. Подробнее об установлении доверия к сертификатам см. в документации на [сайте для разработчиков Android](#).

- задать адрес прокси-сервера.

Ниже приведен пример настройки для стандартного HTTP-клиента `HttpsURLConnection`.

```
val keyStoreType = KeyStore.getDefaultType()
val keyStore = KeyStore.getInstance(keyStoreType).apply {
    load(null, null)
    setCertificateEntry("ca", caCert)
}

val tmfAlgorithm: String = TrustManagerFactory.getDefaultAlgorithm()
val tmf: TrustManagerFactory = TrustManagerFactory.getInstance(tmfAlgorithm).apply {
    init(keyStore)
}

val context: SSLContext = SSLContext.getInstance("TLS").apply {
    init(null, tmf.trustManagers, null)
}

val proxy = Proxy(Proxy.Type.HTTP, InetSocketAddress("127.0.0.1", port))
val urlConnection = url.openConnection(proxy) as HttpsURLConnection
urlConnection.sslSocketFactory = context.socketFactory
```

При таких настройках HTTP-клиента возможно подключение к ГОСТ-ресурсам, сертификаты которых пройдут проверку в ViPNet PKI Client Android. При подключении к ГОСТ-ресурсам, сертификаты которых не прошли проверку в ViPNet PKI Client Android, или к ресурсам с зарубежными алгоритмами шифрования будет выброшено исключение `SSLHandshakeException`.



АО «ИнфоТеКС», 127083, Москва, улица Мишина, д. 56, стр. 2, этаж 2, помещение IX, комната 29

Телефон: +7 (495) 737-6192, 8 (800) 250-0260 — бесплатный звонок из России (кроме Москвы)

Сайт: infotecs.ru

Служба поддержки: hotline@infotecs.ru, телеграм-канал поддержки: t.me/vhd21

ФРКЕ.00175-02 33 06, версия продукта 2.1

© АО «ИнфоТеКС», 2025. ViPNet® является зарегистрированным товарным знаком АО «ИнфоТеКС».

Все названия компаний и продуктов, являющиеся зарегистрированными товарными знаками, принадлежат соответствующим владельцам.