Overview

Abbreviations and Acronyms

This section explains a meaning of key terms and concepts used in this document:

Abberations	Description
AC	Application Cryptogram
AHI	Account Holding Institution
ACS	Access Control Server
API	Application Programming Interface
ARQC	Authorization Request Cryptogram
BAU	Business as Usual
CDCVM	Consumer Device Cardholder Verification Method
CL	Contactless
CVM	Cardholder Verification Method
FCM	Firebase Cloud Messaging
HCE	Host Card Emulation
HVT	High Value Transaction
IBAN	Bank Account Number
JWE	Json Web Encryption
JWT	Json Web Token

LVT	Low Value Transaction
MCBP	MasterCard Cloud Base Payment
MDC	Mobile Data Core
MDES	MasterCard Digital Enablement Service
MPA	Mobile Payment Application
NFC	Near Field Communication
PAN	Primary Account Number
PbA	Pay by Account
PIN	Personal Identification Number
POS	Point of Sale
RNS	Remote Notification Service
SaaS	Software as a Service
SDK	Software Development Kit
SUK	Single Use Key
TAV	Tokenization Authentication Value
TVC	Token Verification Code
VCP/UCP	Verestro Cloud Payments/(Formerly uPaid Cloud Payments)
VPN	Virtual Private Network

Terminology

Name	Description
Customer	Institution which is using Verestro products.
User	User which uses MPA.
Payment Instrument	Common name for card and IBAN.
Payment Token	Token in context of MCBP.
Trusted Identity	Signed user identifier by Customer server. Used to proof that authentication was performed for given user.

Online PIN	Online PIN is a type of CVM where a PIN is inserted on the POS device and verified by the Issuer on the backend.
IBAN Id	Bank account number id which is sha256Hex of bank account number.

PbA high level

This diagram shows high level components which are involved in whole solution.

Image not found or type unknown

PbA key components

Component	Description
MPA	Android Mobile Payment Application provides frontend interface to the User and uses part of Verestro Wallet SDK which is responsible for Pay by Account.
Verestro Wallet Server	Provides the backend services to support Mobile Payment Application via Verestro Wallet SDK and is responsible for managing users, devices, IBANs, Payment Tokens and communication with MDES, communication with Zapp for retrieving CVC2 and ACS. Wallet Server acts as Token Requestor on behalf of Account Holding Institution in context of digitization.
Verestro Wallet SDK	Provides all functionalities needed for MPA in Pay by Account project.
MDES	Token Service Provider which supports digitization(transforming the IBAN into payment token) and is responsible for management, generation and provisioning of transaction credentials into mobile devices to enable simpler and more secure digital payment experience.
Remote Notification Service	Wallet Server communicates with the MPA also using Remote Notification Service. For Android is used Firebase Cloud Messaging.

Zapp Processing Platform	Zapp is bank account processing platform which provides capability to bank to digitize consumer’s bank account, process and enable contactless, e-com and QR payments via digitized bank account. Zapp also stores Static Payment Token with CVC2.
ACS	Component which is part of 3DS transaction to support cardholder authentication.
AHI	AHI holds information about bank accounts and integrates with Zapp and Wallet Server. Can have various services with business responsibilities inside.

Verestro Cloud Payments Solution

Verestro Cloud Payments is solution which had been developed to facilitate adopting cloud-based payments for the Customers. VCP provides functionalities for User identification and verification, Payment Instruments digitization and User data management.

Solution consists of:

- server components:
  - Wallet Server - backend component,
  - Wallet Admin Panel - frontend component,
- mobile components:
  - Wallet SDK - Android libraries.

Implementation model

VCP for PbA should be implemented in model where Customer is owner of MPA. Verestro provides Wallet SDK and Wallet Server. Customer is responsible for direct User authentication and passes the result of the authentication to Wallet SDK. Online operations which need to be performed by User using Wallet Server require valid session on Wallet Server. To obtain user online session with Wallet Server, Customer needs to pass Trusted Identity.

Architecture

This diagram shows big picture of Verestro Cloud Payments architecture.

Server Components

Server components are applications which need to be deployed on remote server to make possible to connect them by network.

Deployment Models

Verestro offers two deployment models of server components. On-premise and SaaS.

SaaS - Server components are designed to be deploy in SaaS model. In this case everything is deployed and configured on Verestro side. Verestro is responsible for maintaining infrastructure, deploying applications and monitoring.

On-premise - Server components also can be deployed on Customer infrastructure. Applications are designed to be deployed using Kubernetes as system for automating deployment, scaling, and management of containerized applications. For more details please contact Verestro representative.

Mobile Components

Mobile components are dedicated for using on Android mobile devices.

Wallet SDK

Verestro provides Software Development Kit (SDK) called Wallet SDK which can be used in Mobile Payment Application. As a company, Verestro provides many products which can be used in single application. For that reason Wallet SDK is divided into separated modules which covers different functionalities. There are two main modules dedicated for Verestro Cloud Payments: MDC SDK and UCP SDK. MDC SDK is core Verestro module responsible for user data management: authentication, payment cards management - since these are main functionalities used in every product. UCP SDK is dedicated for performing digitization and payments using Payment Token. In payment context UCP SDK wraps Mastercard Cloud Based Payment SDK.

Requirements

Wallet SDK has some mandatory requirements to make it work:

- device cannot be rooted,
- Android OS image (ROM) should be genuine in version 6.0 (Marshmallow) or above,
- devices cannot have enabled debugging.

There are also some not mandatory requirements, but Customer needs to be aware of them to maintain functionalities:

- NFC module necessary for HCE payments,
- lock screen necessary for locally-verified user authentication.

Security

Wallet SDK was developed according to security requirements included in Security Guide MCBP SDK for Android. However Wallet SDK cannot guarantee full MPA protection and MPA must provide additional layer of security. More detailed information can be found in *Wallet SDK API.* Moreover all sensitive data are passed as chars or bytes arrays. Wallet SDK copies the arrays and clears that copies just after processing. MPA should clear provided sensitive data immediately after passing them to Wallet SDK.

MPA should provide mechanism for forcing application update in case of SDK security checks update.

Security Checks and Data Clearing

On Wallet SDK side are performed security checks which includes static code analysis protection and dynamic analysis protection. Security checks consists of:

- root access detection,
- hooking protection,
- debugging protection,
- custom ROM protection,
- data tampering protection,
- man in the middle protection.

Security checks are performed periodically, if Wallet SDK detects any of above things all data hold by Wallet SDK will be cleared and security report will be sent to Wallet Server. MPA will be informed about such detection.

Communication with Wallet Server

Communication from genuine applications which are installed on genuine devices is accepted by the Wallet Server. Wallet SDK at the very beginning performs authentication of application and device to Wallet Server. This authentication may takes advantage of Google Play Integrity which is 3-rd party trusted side in whole authentication. Google Play Integrity verifies device and sign information about device and application. Signed data from Play Integrity are sent to the Wallet Server. Wallet Server verifies data and allow or does not allow for further communication. Application is verified according preconfigured application certificate digest used for signing application.

Important: There is a limit of requests to Google Play Integrity API: 10 000 per day. If Customer predicts that there will be more installations per day then this limit needs to be increased during Google Project Setup.

Wallet SDK communicates with Wallet Server using TLS 1.2. Wallet SDK performs public key certificate pinning when it tries to establish connection with Wallet Server (similar with connection to MDES). Certificates for the pinning needs to be provided to SDK. Sensitive information are

additionally encrypted and/or signed.

Versioning

Wallet SDK uses semantic versioning. It means that every release has own version which is MAJOR.MINOR.PATCH, where:

- MAJOR version increases when SDK has incompatible API changes,
- MINOR version increases when new functionality is added in a backwards compatible manner,
- PATCH version increases when new backwards compatible bug fixes are introduced.

MAJOR versions are supported 6 months and Customer needs to migrate to new version if they want to maintain support.

Remote Notification Processing

There are several processes where server sends message to client. Remote Notification Service(FCM) is used to deliver such message. Wallet SDK is responsible for remote message processing, however MPA is responsible for obtaining FCM registration token, handling FCM token update and receiving remote messages. Before passing remote message to SDK, MPA needs to verify if given message is dedicated for SDK by checking sender Id. Sender Id is configured during onboarding. Verestro will create new FCM project and provide data needed to obtain FCM token for given project. Due to observing some issues with FCM token refresh notification from FCM service, additional check of new token availability is recommended(eg. on application start). See more in *Wallet SDK API.*

Access

Wallet SDK is stored as artifacts in maven repository. Access there is provided during onboarding by Verestro representative using pgp encryption.

Configuration

Whole product has configuration which needs to be fulfilled. This configuration also consists of data which are set in MDES. More details are described in *Wallet Configuration*.

User Experience for Contactless Transactions

MDES offers a few options for customers for defining user experience for contactless transactions. Final option is the choice of CDCVM type (Mobile PIN, Locally-verified CDCVM) and CVM model (Always, Flexible, Card-like).

CDCVM Types

There are two types of Consumer Device Cardholder Verification Method (CDCVM) which are supported by MDES.

Mobile PIN CDCVM - A PIN value (4–8 digits) that the cardholder enters on the mobile device and that is validated online by MDES during the transaction authorization process. Since Locally-verified is mostly preferable option, Mobile PIN is out of scope.

Locally - verified CDCVM (custom) - A CDCVM entered on and validated by the consumer’s mobile device, for example system device PIN, pattern, password or biometric methods (such as fingerprint, iris or facial recognition). Swipe (slide to unlock) is not a valid cardholder verification method and must not be supported. These methods are commonly associated with a device unlock process and are validated on the cardholder’s mobile device. From SDK perspective there is only requirement to pass information whether transaction is authenticated or not but process of authentication is performed on Mobile Payment Application level. A Locally-verified CDCVM applies to all the payment tokens of a given Mobile Payment Application instance (“Wallet-level”). In some parts of system this type is also named as Custom.

CDCVM Models

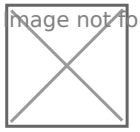
For two CDCVM types customer can apply different user experience CVM Models.

- **Always**

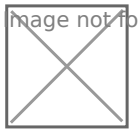
In this model the card profiles supplied to the SDK are configured to indicate that the mobile device supports on-device cardholder authentication. When transactions are performed on a POS supporting CDCVM, the POS will delegate cardholder authentication to the mobile device the terminal will not request an Online PIN on the terminal. In POS which does not support CDCVM cardholder authentication is required using Online PIN. This model requires the cardholder’s mobile device to authenticate the cardholder for all transactions (LVT, HVT, Transit). CDCVM can be performed using either a Mobile PIN or a Locally-verified CDCVM. MPA is expected to decline any transaction for which cardholder authentication is not performed or is unsuccessful.

Below are presented sample diagrams which show how the transactions can look like:

Always LVT, HVT - single tap



Always LVT, HVT - double tap

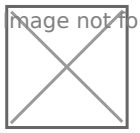


- **Flexible**

In this model the card profile also indicates that the mobile device supports on device cardholder authentication Mobile PIN or Locally-verified CDCVM. However rather than applying authentication for every transaction the MPA defines flexible criteria such as allowing multiple transactions between each authentication. This criteria are often names as Lost & Stolen options or velocity checks. For transit transactions cardholder authentication is not expected.

Below are presented sample diagrams which show how the transactions can look like:

Flexible LVT



Flexible HVT - single tap

Flexible HVT, LVT with Velocity counters - double tap

Card-like

In this model the card profiles supplied to the SDK are configured to indicate that the mobile wallet is not capable of supporting on device cardholder verification. This means that when transactions are performed with a Point of Sale (POS) terminal, the POS will treat the transaction in the same way as a card transaction. Typically this means that low value transactions (LVTs) will be processed without additional user authentication and if supported, high value transactions will require an online PIN to be provided on POS. This model is put here just for general information, however it is not preferred for issuer wallets.

Below are presented sample diagrams which show how the transactions can look like:

Card like LVT

Card like HVT

Image not found or type unknown

Lost & Stolen options

The Lost & Stolen options are dedicated to control performing transactions allowed before requiring cardholder authentication. This limits fraud risk if the cardholder’s mobile device is lost or stolen. Lost & stolen options can be applied only for Flexible CDCVM. These options also are known as velocity check counters. Wallet SDK provides interface which is invoked during transaction. Transaction information like range, rich transaction type, amount are provided within this interface. MPA can implement various checks to support velocity check counters using transaction information. MPA for example can count LVT transactions and allow only some predefined number of LVT transactions without cardholder authentication.

Transit transactions

Transit transactions are transactions with given Merchant Category Code performed e.g. on traffic gates like: underground. It is up to Customer if wants to enable such transactions or not (option selected during MDES onboarding). Transit transactions are enabled in every CVM model, however for Always CDCVM needs to be performed for every transaction, for Card-Like and Flexible CDCVM can be skipped.

Revision #12
Created 17 March 2023 11:25:31 by Jagoda Mazurek
Updated 27 April 2023 15:01:35 by Krzysztof Boczek