**Overview of Digital Payments in Mastercard :**

Digital Payments in Mastercard falls into two different categories –

1. Digital Wallet Solutions
2. MDES( Mastercard digital enablement services)
3. **Digital Wallet Solutions/Masterpass –**

Overview: As the name wallet suggests , it is used to store card details of a user securely in a digital way and transfer the card details to merchant during checkout for payment processing. All the digital wallets is a part of Masterpass umbrella.

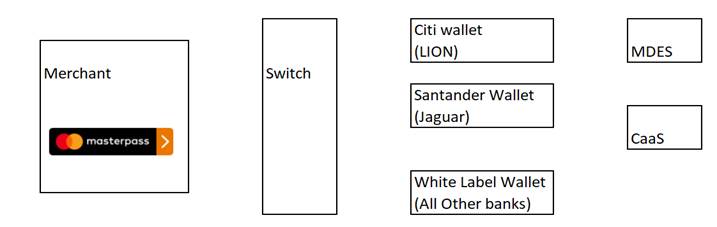
As part of Mastercard business strategy for wallet , Mastercard is B2B  with the bank and bank goes B2C with the customer . So Mastercard has built wallet solutions for different banks and user card data is stored in these wallets separately based on which issuer bank card is used. So e.g – citi bank customer data will be stored in Lion wallet created by Mastercard on behalf of Citi Bank .

When registered customer click on “Buy with Masterpass”  button in the merchant website , customer will be redirected to different based on which bank that customer belongs to.

**Technology Used –**

Front End – Angular JS

Backend – Each of these wallets are monolithic application developed in Spring Hibernate. F5 load balancer -> Apache web server ->  JBoss Server - > Oracle database.

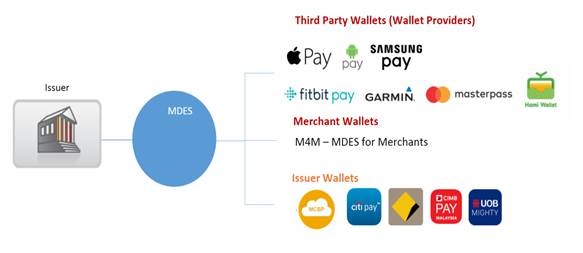


All the digital wallet makes API call to CaaS(Crypto as a Service) which is a HSM based crypto solution as all user data stored in the wallet will have to be encrypted.

All transaction via digital wallet will have to do card tokenization where 16 digit card number replaced by another 16 digit along with cryptogram (Similar to chip in our debit card).

1. **MDES–**

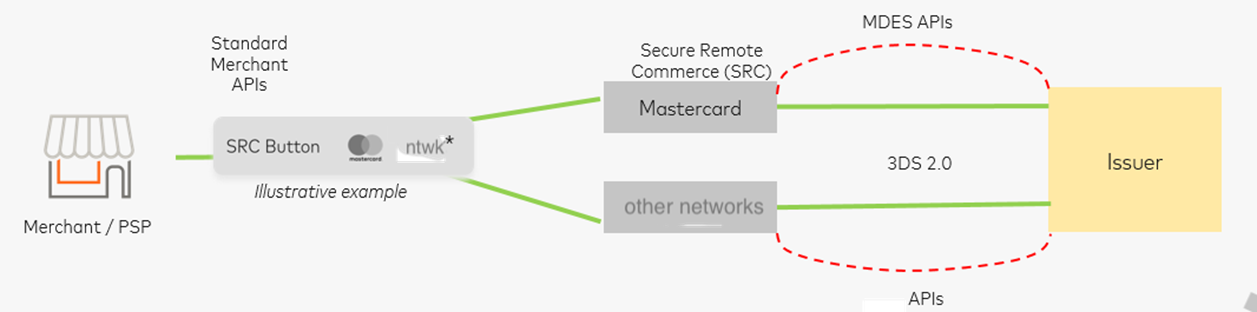
The MasterCard Digital Enablement Service (“MDES”) is a suite of on-behalf-of services that support the management, generation, and provisioning of digital payment credentials into mobile devices, to enable simpler, more secure digital payment experiences.



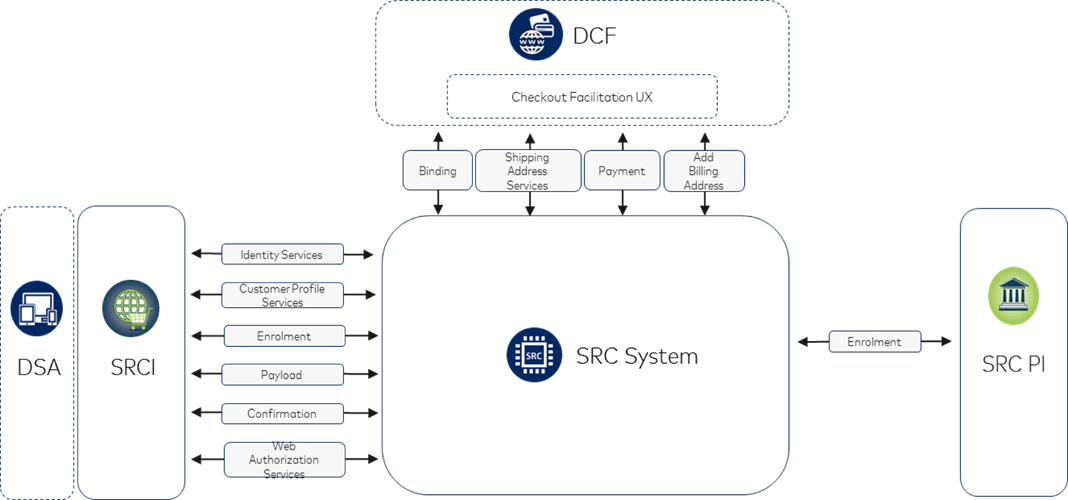
MDES connects issuers to digital wallets and used by Samsung pay, Google pay, Apple pay.

MDES maps the token to the PAN , validates the token status: state and expiry date , validates the transaction data: E.g. Cryptogram, indicates the result to the issuer.

**SRC Overview**



**Secure Remote Commerce (SRC)** is a common approach that provides security and interoperability to deliver the safest card payment experience in a remote environment. It aims to enable the secure exchange of payment information through common interfaces between participating entities, which may include, for example, merchants and issuers.



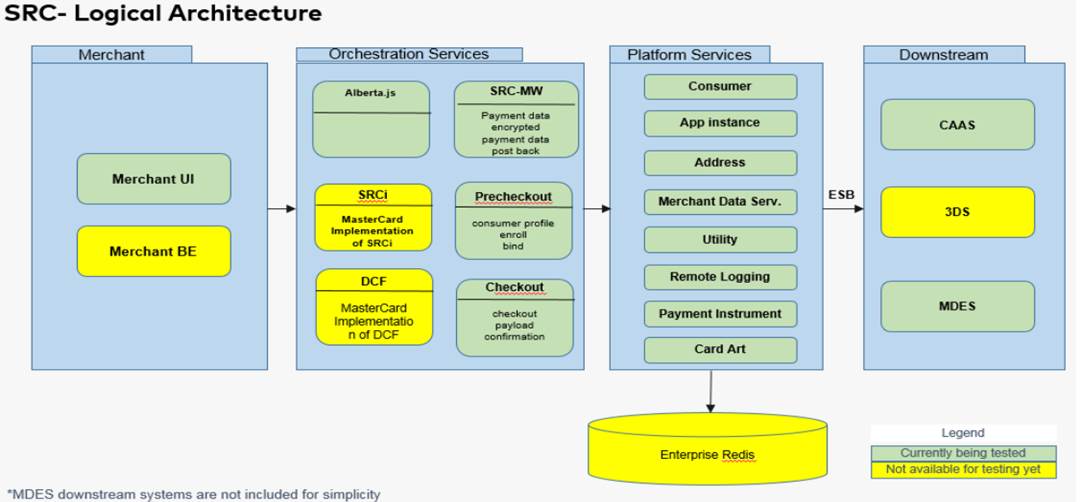
An SRC System provides access to Checkout Services that enable Profile, Enrolment, Binding and Checkout events, facilitating the storage of Cardholder’s payment information and its transmission to an **SRC Initiator (SRCi)**.

The resulting data from the successful completion of a Checkout Service is payment data that a merchant can use to initiate an authorisation request through its designated payment provider. A merchant or a payment service provider, acting as an SRC Initiator, facilitates access and integration to Checkout Services.

The integration of a Checkout Service is determined by the SRC Programme in coordination with the **SRC Initiator(SRCi)** and **Digital Shopping Application(DSA)**. The SRC System interactions occur based on triggered events. The specification does not prescribe any one-implementation approach, but describes the interactions between the roles, functions, messages and the SRC System and SRC Participants.

As part of SRC , Alberta will go live on June 2019 which will replace traditional Masterpass and other wallets. Alberta wallet will be a single wallet replacing all other bank wallets eventually.

**Alberta:**



In Alberta , wallet architecture is broken down into 14 microservices.

Platform services-

These services will store encrypted user data . Like Address will store encrypted billing address , consumer will store encrypted user email, mobile number etc. These services use CaaS for encrypt/decrypt .  Permanent storage is  Mongo DB as well as Redis cache. Payment instrument service stores card token data and card cryptogram.

Orchestration services –

Merchant UI or Merchant Backend will call the API endpoints of one or more orchestration services . These services communicates with one or more platform services to get stored user data .

**Technology Stack –**

**Front End –** Angular JS 4.

**Back End -**  All the microservices are deployed in Mastercard on-premises installation of pivotal cloud foundry . Enterprise Redis and Mongo DB is used which is hosted outside of PCF . Platform services talks to Downstream componets like CaaS/MDES via IBM ESB . IBM ESB in mastercard by all the application.

**Inter Service Communication – one microservice communicates with other microservice via Synapse.**

**Synapse –**

Synapse is a home grown framework/tool based on Zuul API gateway.

It's comprised of multiple components encompassing

* Distributed dynamic routing.
* Client-side loadbalancing.
* Fault tolerance to failover with no client-side errors.
* A new security framework allowing group control over service consumption.
* Operational dashboards for monitoring and maintaining the framework and deployed services.

*The goal of Synapse is to* provide developers with easy-to-use tools and SDKs that allow them to deploy their services in a distributed manner and consume other services deployed in the framework. This enables us to run services in a microservice architecture at MasterCard.

***PRISM***

Consumer-side java based library that handles service discovery, client-side authorization for provider calls using PDP & client-side load balancing. This is based on Netflix project Ribbon.

**LOCUS**

Provider-side java based library, that handles service registration to the registry, server-side authorization & API health-checks. The purpose of Locus is to register a service with the Synapse framework. A developer includes the Locus dependency in their project and configures it. When the service starts up, it is Locus that registers it to the Synapse framework. This is similar to Eureka Client in Netflix Project

**ZOOKEEPER**

Service registry, which stores the meta data about the service. All other synapse components talk to ZooKeeper to get data about the registered service.

**POLICY POINTS (PAP, PEP, PDP)**

For a service to register or consume in Synapse framework, a valid security policy has to be on-boarded. This policy defines who can consume what service on how method. There are 3 components that handle the policy management within **Synapse –**

PAP (Policy Administration Point) – API that allows internal users to CRUD security policies on API.

PEP (Policy Enforcement Point) – Intercepts incoming API calls to a provider & verifies with PDP to see if the consumption is allowed.

PDP (Policy Decision Point) – API that actually fronts the PAP to provide run-time decision about access control based on policies.

**AKER**

This is implementation of Netflix Zuul API Gateway . Aker is between XML GW & PCF / DMZ2 services & is vital for exposing any synapse API's externally via XML GW. Aker is also responsible for translating the SAML tokens from XML GW to JWT, that synapse API's understand.