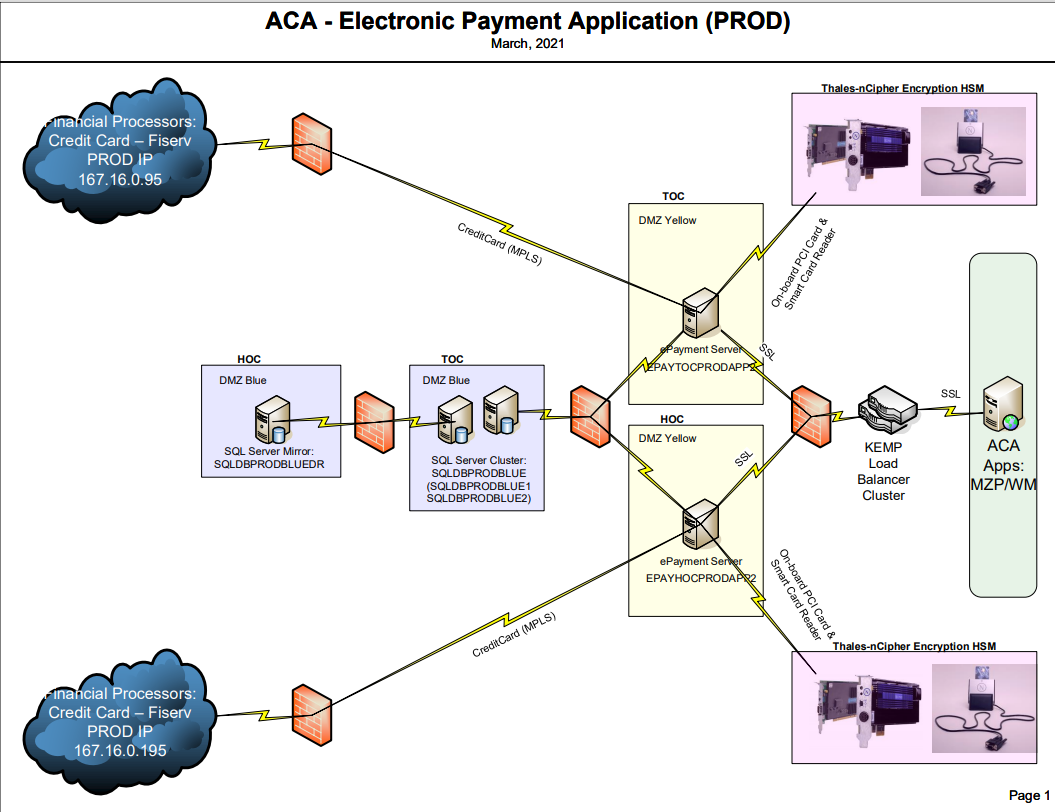
# Epayment Knowledge Transfer

## 1 – Introduction

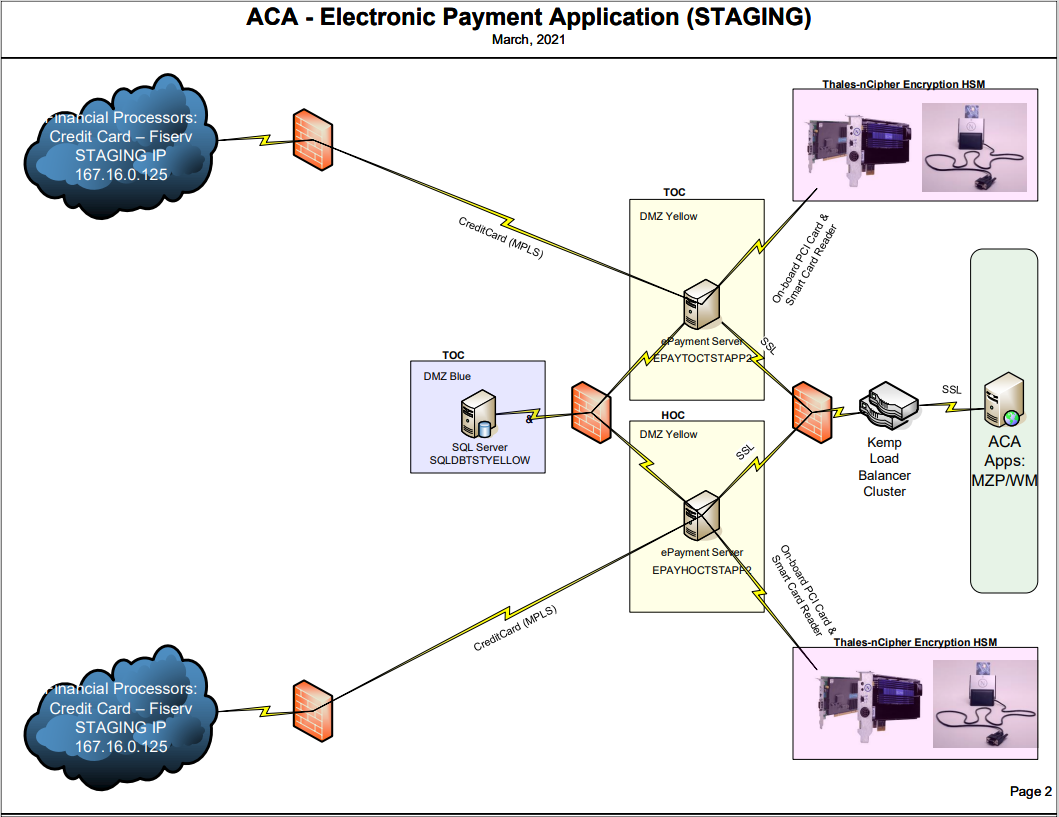
Please watch videos in videos folder and read the “doc/video\_summary.docx” for a quick review of content.

## 2 – Architecture

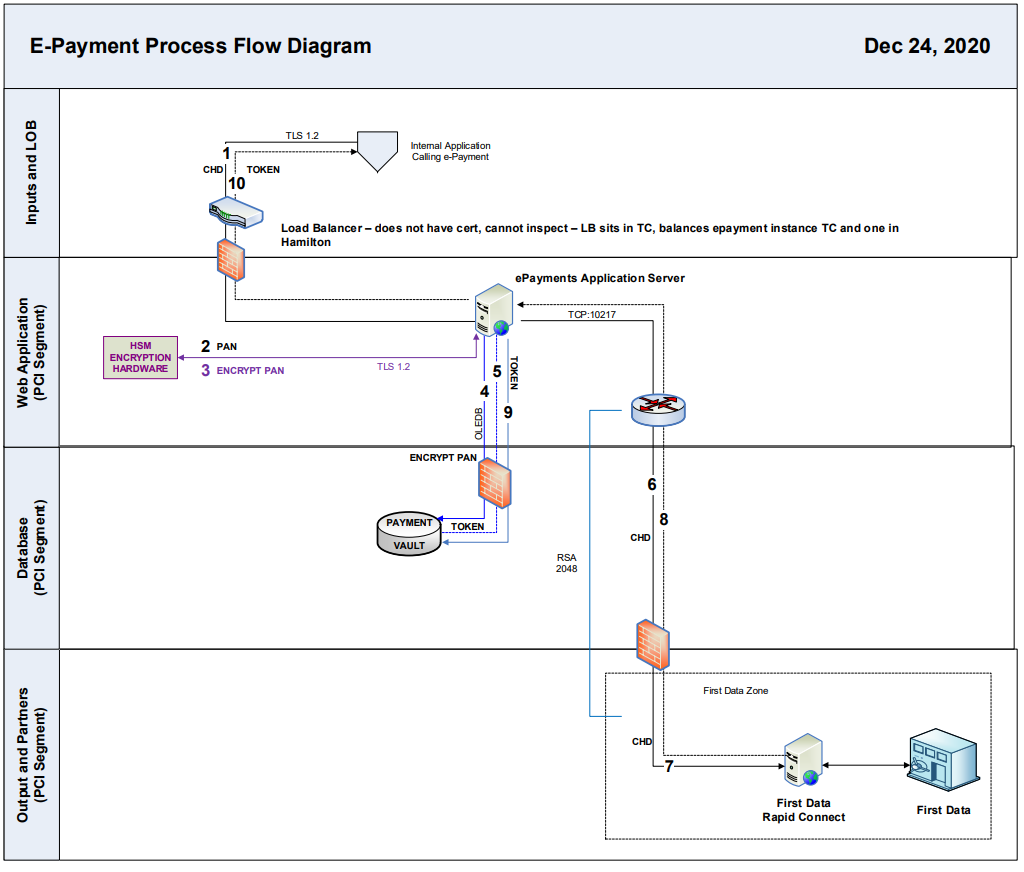
### Production Environment Diagram

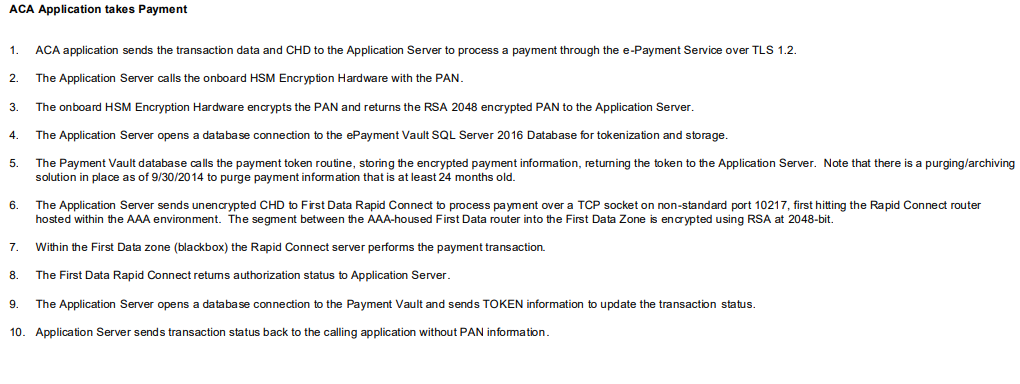


### Staging Environment Diagram:



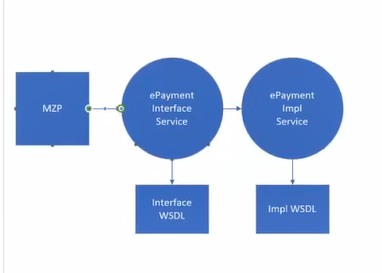
### Flow Diagram:



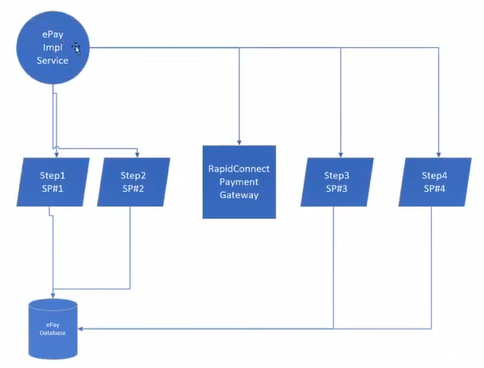


### View from client

For example from mzp

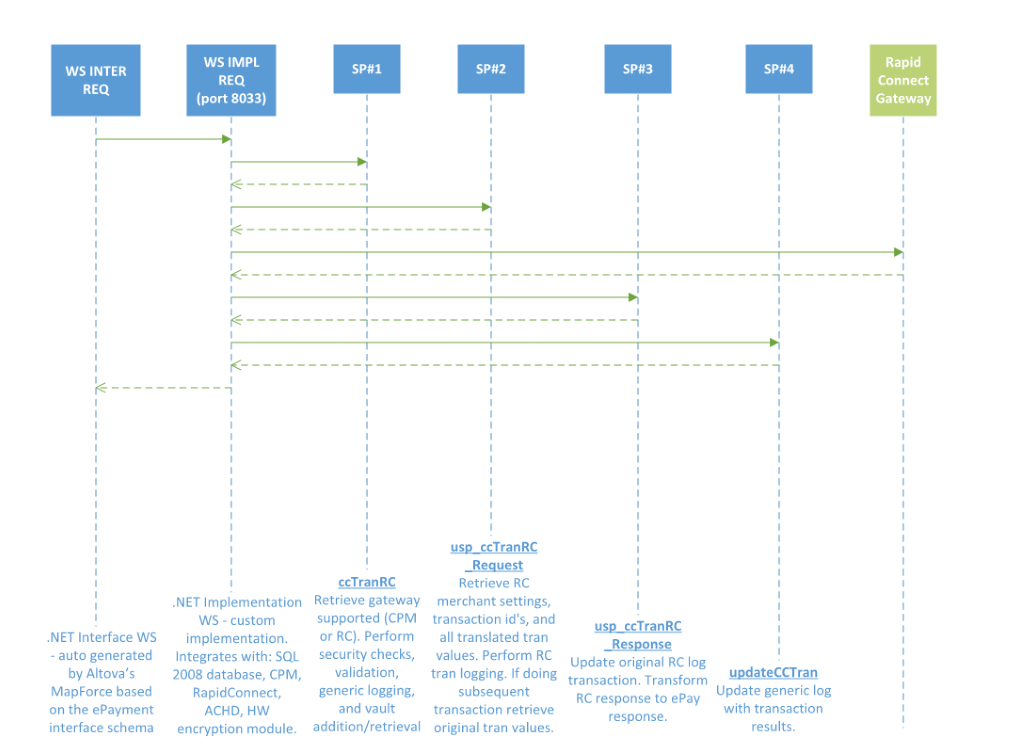


Inside implementation Service Application



Attention, please refer to 4.3 for Utility Transactions that are not going through all steps because there is no need.

### Implementation Service Sequence Diagram



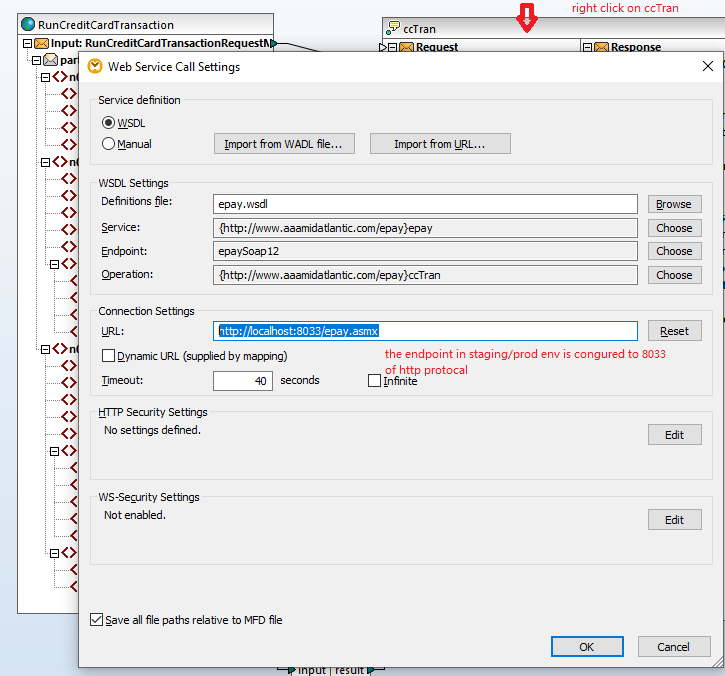
## 3 - Project Life Cycle

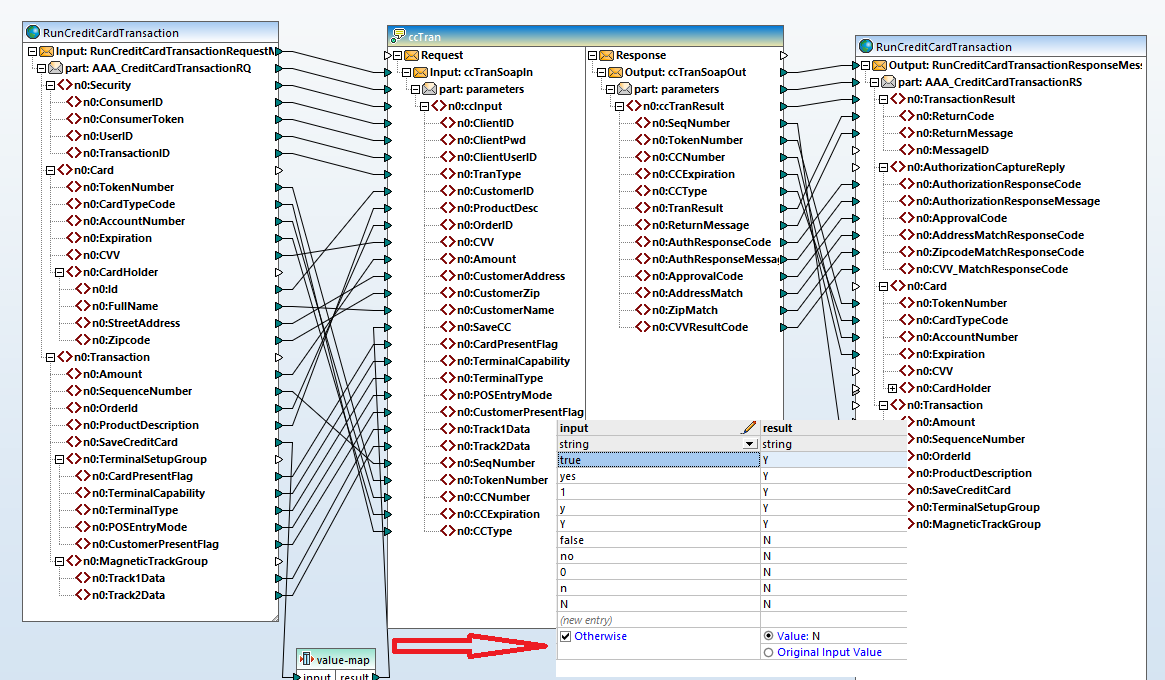
Please refer to documentation: “doc/Rapid Connect User Manual.pdf” for how to work with FiServe to setup sandbox and create project, and how to create test scripts. (I assume it is a soapui project that Ed created). The test cases should be completed for certification and then the project is moving along the stages and eventually be completed.

## 4 - Development

### 4.1 Map

Please read documentation under “doc/map\_interface\_code\_generation.docx” for how to generate the map code into the .net c# project.





### 4.2 Interface Application

After generating the code into .net application with steps in 4.1, modify the web.config in the .net application and deploy it to the staging/production environment. The endpoint should be like <http://localhost:8033/epay.asmx>, since in prod/staging, the port number for implementation service is configured to be http with 8033. This http port is not configured to be accessible to outsiders, but since impl/interface applications are on the same server, it should be ok.

This interface application/serves only for the transformation purpose. Some of the clients are using the implementation service directly. Preethi and I talked about the possibility of rewrite the application into a .net application directly to get rid of the map. She doesn’t want WSO2 solution because she wants to avoid the issue of PCI scope.

### 4.3 Implementation Application

Please refer to implementation Service sequence Diagram for an overview.

#### 4.3.1 - Main flow description

1. First stored proc call to return merchant gateway supported in addition to security checks, validation, epay logging, and vault extraction

2. If merchant still uses CPM continue with existing flow

3. If merchant uses RC then call new stored proc passing all values to get back RC merchant settings, transactional id's, and all translated tran values. Perform RC logging. If doing subsequent transaction then retrieve original tran values.

4. Build RC XML request message

5. Send request to RC

6. If timeout received (35 sec) issue RC Timeout Reversal transaction up to 3 times if can not complete then process exception.

7. If RC response received process it by calling another stored proc to update original RC log transaction. Transform RC response to ePay response.

8. Continue flow with calling existing second ePayment stored proc to update ePayment log

9. Respond to client

#### 4.3.2 - Utility Flow description.

Not every type of transaction are reaching out to Rapid connect. SAVE\_CC etc only works internally. Please refer to 5.2 for Ed’s certified soapui project for more details. Under some situation the utility transaction only calls ccTranRC, the 1st stored procedure, and exit out. (LOOK\_UP\_BY\_TOKEN for example)

#### 4.3.3 - Credit card encryption

Please refer to “doc/HowToRotateNCipherCryptoKeys.doc” for how to rotate the encryption key for credit card.

Alone with hardware, there is a site deployed to the physical staging/production server. <http://localhost:8099/cryptoWS.asmx>

Need to login to the physical server and access the site to create a new container and key.

In local development env, the implementation application is using the windows encryption method to encrypt the credit card info, and save the value into the vault. The row created into vault table has the column to store the container name. When retrieving the value from vault and decrypt it locally, the windows decryption method is used. But with a token number in hand, the implementation application has no way to tell whether the token is created by NCipher or windows. So in summary, local env should only decrypt the vault value created locally.

In web.config, <add key="useNCIPHER" value="False"/> is used to control the encryption method.

If a token was created before the container rotation, the old container name can be retrieved with the vault value and the implementation application can detect this and try to use the new container name to encrypt it again and restore it back in updateCCTran.

If a token was created before, but when cc number is passed in with an existing token number, the new passed in cc information will take priority and be used to update the vault value back in updateCCTran. The new credit card will be used to connect with RC. Please check specific scenarios and see how it is handled.

#### 4.3.5 – Rapid Connect

In web.config there is a switch to control whether reach out to rapid connect to complete the transaction. Under certain situation, mzp client wants to get a rejection for credit card transaction.

Rapid connect is consumed by tcp socket connection. To do that, we need to create an xml object based on the schema (v2.0.8 API) provided by Fiserve, “doc/UMF Schema V2.10a.6V4.xsd”. Ed was using a utility in windows to generate the cs code and included into the implementation application as **UMFSchema.cs**.

##### Original command used by Ed:

C:\Program Files\Microsoft SDKs\Windows\v7.0A\bin\xsd.exe "c:\temp\out\UMFSchema.xsd" /classes /language:CS /outputdir:c:\temp\out\

The above xsd.exe is not on my local machine (according to my previous note with Ed, it looks like ‘a .net tool (based on .net framework 2)”, instead I found it here but didn’t try it yet.

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin\NETFX 4.6.1 Tools\xsd.exe

We need to regenerate the cs file if Rapid connect forces an upgrade to a newer schema, or if clients want a new feature.

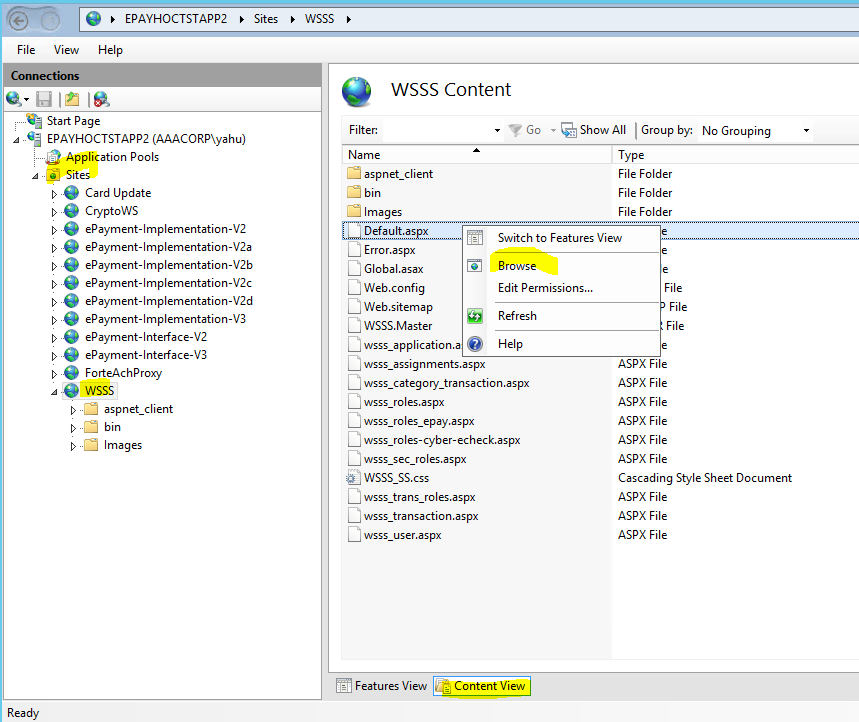
The way to use the schema is to refer to the documentation: “doc\Universal\_Message\_Format\_2014.01.22\_7292.pdf” for specific column meaning and definition, and then refer to the schema a little. Most of the data preparation and transformation happens in stored procedure “usp\_ccTranRC\_Request”. Before exit the stored procedure, the data got stored in the CC\_TRANS\_RC table. The primary key of this row will be returned as well as sequence number.

#### 4.3.4 – Web.config

|  |  |
| --- | --- |
| MPWeb.services.TransactionService | ignore , echeck related |
| PGWeb.services.TransactionService | ignore , echeck related |
| PGWeb.services.IDVerifyService | ignore , echeck related |
| ecStoredProcStep1 | ignore , echeck related |
| ecStoredProcStep2 | ignore , echeck related |
| ecSuccCodeSP | ignore , echeck related |
| ecSuccCodeOut | ignore , echeck related |
| ecSuccMsg | ignore , echeck related |
| IDVerifyEnabled | ignore , echeck related |
| IDVerifyEnabledOnValidate | ignore , echeck related |
|  |  |
| useNCIPHER | Local need to set it to false, it can encrypt with microsoft way instead of nicipher |
| cryptoKeySize | 2048, |
|  |  |
| tranContainerName | EPAY\_TEST\_4 |
| userContainerName | EPAY\_TEST\_4 |
|  |  |
| BypassDBValidation | Can by pass any data validation inside database when set to true.  This one is used in ccTranRC to see whether to validate the data for this transaction. |
| BypassAPPValidation |  |
|  |  |
| CCAmountTestMode |  |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\D6851C18.tmp | AMEX related |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\C15191E6.tmp | Address validation related. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\83B15EE4.tmp | Email related. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\EC2AFA92.tmp | ignore , echeck related |
| EpaySrvName | This stores information about which .net server processed the transaction |
| rcStoredProcStep1 | usp\_ccTranRC\_Request |
| rcStoredProcStep2 | usp\_ccTranRC\_Response |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\48754870.tmp | Determine whether rc xml is going to be stored or not. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\E40E77FE.tmp | Sp to store rc related xml |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\D255E4BC.tmp | Sp to handle when calling rc and time out etc. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A620F62A.tmp | Related to send msg to rc. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\2382FFC8.tmp | RC Test mode related. If set to true, will not send to rapid connect, instead the standard xml will be send back. |
| C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\71402116.tmp | Additonal test mode setup. |

### 4.4 WSSS Application

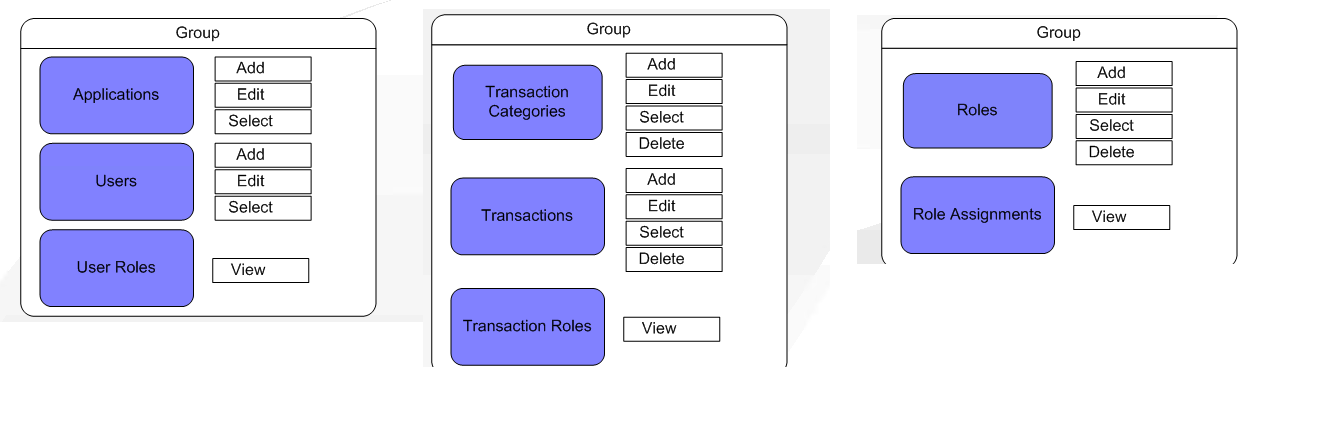
This WSSS application can be reached by link <http://localhost:8005/Default.aspx> on staging/production servers. Further research needs to be done to under the security set up. On one of the testing server, the link doesn’t work because the connection string is obsolete.

****

This is my assumption based on the video 5, starts at 16 minutes, for adding a new application and then user etc:

1. Use the Application tab to add a new Application like MZP\_X.
2. Add users to the application
3. Assign roles/create roles to the user
4. Add transaction to a role.
5. Merchant admin function is not used in the WSSS any more (video 5, 20 minutes) after the implementation of rapid connect.

The following is the functionalities provided by wsss as “site diagram”, I don’t see the links to do the role assignment to the user and adding transactions to a role. Need to take a look at the wsss application code to figure out later.





### 4.5 Database

#### 4.5.1 Tables Overview

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Description** | **Sample Data** |
| user |  | MZP\_USER |
| User\_apps | This is the primary tables, user is the child table. Which means one app can have more than 1 user account if need to,  but right now it is 1-1 relationship | MZP |
|  |  |  |
| ECHECK\_CONFIG | Echeck related , ignore |  |
| ECHECK\_RESPONSE | Echeck related , ignore |  |
| ECHECK\_trans | Echeck related , ignore |  |
| EFTACH\_RESPONSECodes | Echeck related , ignore |  |
|  |  |  |
| CYBER | Credit card related merchant account table.    Created to be used for the cybersource, which is replaced by Rapid Connect.    REFER to 4.5.2 for more information. |  |
| MERCHANT\_SETTINGS\_RC | This table was added for rapid connection integration, it provides additional information needed for rapid connect. If MERCHANT\_FC\_FLAG in Cyber table is "Y", then additional information will be retrieved from this table.    So this.MerchIDNorth column needs to be the Cyber table's CYBER\_MID value, this how they are joined.    MerchIDNorth, MerchID in production has the same value, but they have different values in testing environment. MerChID in testing has the same value becaue they belongs to the same merchant in sandbox .    MerchID value is passed to rapid connect.    Other columns' values will be passed to rapid connect, can refer to rapid connect documentation for the meaning of the columns |  |
|  |  |  |
| TRANS\_ACTIONS | Audit table    TRANS\_DATE\_TIME: request time comes in  COMPLETED\_DATE\_TIME, response time go out.      There is alert generated if the difference between the above 2 columns is more than 10 second. It indicates bottleneck or timeout of the system.    If people reports error, this is the table that we need to look into.    Some of rows are generated by the purge process.    C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\D5964E39.tmp |  |
| CC\_TRANS | This is table for the real transaction    Anything in the payload of request xml will be stored here and response xml field.s.      TranState: C means captured.    SequenceNumber\_orig: say we want to void a transaction, in the parent transaction the sequence number wlll be saved here.    PAN: is cc number 16 digits.  Encryption is happening inside epayment code , NCipher?    ContainerName: is where the encryption key was stored. We are suppose to rotate the key every year.    Say if PAN String is read with the encrypted value, it will check which container it is, then grab the key from that container,  and try to decrypt it. |  |
| CC\_TRANS\_RC | This is created when integrated with Rapid connect, it is a child table to CC\_TRANS    It has the data that got sent to and received from rapid connect. Ed said typically you don't need to check this table unless you want to check the transaction a little more when error occurred.  The primary key in this table is used as sequence number for sub transactions, for example, the CAPTURE transaction can use the AUTH related CC\_TRANS\_RC row’s primary key as sequence number. |  |
| CC\_TRANS\_RC\_XML | This is the actual xml got sent and received from rapid connect.    This table got purged every couple of months.    The request xml contains the last 4 numbers. |  |
|  |  |  |
| Aaa\_Merchants | This is not used by epayment, it is for report purpose. It is more of a lookup table.    Every time a new merchant is added to the system, a row is created here as well. So we know how many merchants are there etc. |  |
|  |  |  |
| Admin | User need to be an admin, also the correct role is set up in mzp , so they can see the credit card number |  |
| AllSequences | This table maintains sequences. It is no need to dig into right now.    Rapid connect requires some number to be reset every day, it starts from 1 and increase,    Some of the sequence can keep growing. |  |
|  |  |  |
| Vault | Main table here, it has all the credit card information . Token number of the credit card is the primary key Vault\_ID here.    Vault\_ID: token number (10)  TOKEN: it has more digits, the last 4 are credit card last 4. It is not really used,  VAULT1: credit card  VAULT2: expiration date  VAULT3: Card type.    VALUT\_TS: the last time the record was touched. Selected, created etc.    CONTAINER\_NAME:    DELETED: Y means it is deleted. Not really used. |  |
| VAULT\_ACCESS | Child table, to keep auditing record when Vault is accessed.    LOOKUP: Y means somebody did a full credit card request back. If it is N, it means either the user was not allowed to see it or it was froma  A system batch process. |  |
|  |  |  |
| Upd\_loc\_name | It is a temp table. Prod may have some other temp tables. |  |

#### 4.5.2 – Cyber table

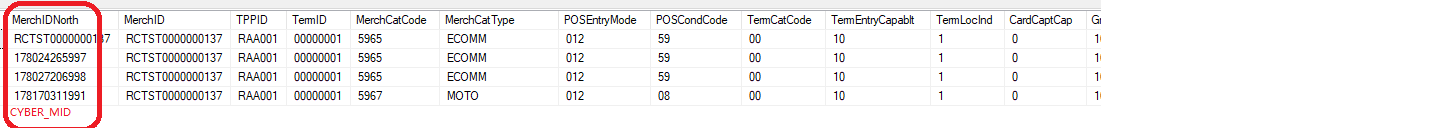
|  |  |
| --- | --- |
| CYBER | Credit card related merchant account table.    Created to be used for the cybersource, which is replaced by Rapid Connect.    It is used by the user id, mzp has user id 3 just remember    LOCATION\_CODE is the branch code    CYBER\_MID is the primary information    Since it was used for Cyber, the last 2 columns are important:    MERCHANT\_FC\_FLAG: Value Y means it was for Cyber but got migrated to Rapid connect, during the transition period, this flag value can determine where the payment should go to, either Cyber or Rapid connect.    MERCHANT\_RC\_ACTIVE\_TS (activated timestamp)    At this point, since the transition is done, All rows should have the MERCHANT\_RC\_FLAG as true, but you might still see the check of this value in the code.    MERCHANT\_DEFAULT if true, which means when query of combination (userid and location cd) can't find a match, then the default merchant (this column= Y) for a particular userid is used for this situation. If no merchant was set as Y, the default, then the 1st row of the user will be used.    CYBER\_IP: Obsolete    CYBER\_PORT: Obsolete.    MERCHANT\_CODE: not actively used, but the logic is still there. It was used for point of sale system. In the past one location can have different merchant accounts for different sub type. RS stands for POS, MOTO is for something else. By passing this subtype information, we can get the correct merchant for this subtype.  Mzp doesn't use this sub type in the xml node userID    MERCHANT\_NAME:    CYBER\_ECOMM: this flag is telling Rapid connect whether it is a ecommence transaction or not, If it is, the information got passed to Rapid connect is different. Ed kind of mentioned WM only do tokenization, but MZP is making the actual capture.    **CYBER\_CARD\_PRESENT\_FLAG/CYBER\_TERMINAL\_CAPABILITY/CYBER\_TERMINAL\_TYPE/CYBER\_POS\_ENTRY\_MODE/ CYBER\_CUSTOMER\_PRESENT\_FLAG**:  The above 5 fields act like default value used to pass to rapid connect. They can be passed in the request xml, and the XML value passed in request take priority.    C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\6232594.tmp    PAN\_READABLE:  Means:  if the user has the authoriztion right needed to see the full credit card back, or as an admin user passed the admin check in cc\_TRAN\_RC  If this location in cyber table has the Flag value set to "Y",  If the user in mzp has the correct role type.  then in the interface response xml, full 16 digits of cc number will be returned back.  save cc: both request and response xml will have the 16 digits cc number.  look up by token, request xml will have the token number and repsonse will have the credit card number.  So this control depends on location's flag value and the user's authorization level and mzp role.    TRAN\_DUPS\_INTERVAL:  if value equals to 0, there is no check of payment duplicates,  if value greater than 0, for example 10, then if duplicate payment was detected in the past 10 minutes, then the transaction will stop. |

C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\332E64C2.tmp

C:\Users\yhu\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\EE5FA220.tmp

#### 4.5.3 - MERCHANT\_SETTINGS\_RC table

The CYBER\_MID in cyber table matches MerchIDNorth column in this RC related specific table. MerchID is a value used for RC request xml. In testing env, all MerchID has a value “RCTST0000000137” because it a sandbox set up. In production this column reflects the real value.



Testing env table values

|  |  |
| --- | --- |
| TPPID | Constant value: “RAA01” |
| TermID | “0000 0001” “5961930”(1) |
| MerchCatCode | 5965/5967/8675 |
| MerchCatType | MOTO /ECOMM (3)/ RETALE (1) |
| POSEntryMode | 012/902 |
| POSCondCode | 59 /08 (E)/ 00 (R) |
| TermCatCode | 00/01 (R1) |
| TermEntryCapablt | 10/03 (R1) |
| TermLocInd | 1/0 (R1) |
| CardCaptCap | 0/1 (R1) |
| GroupID | 10001 constant |
| PartAuthrztnApprvlCapablt | 1 |
| ACI | Y |
| VisaBID |  |
| VisaAUAR | 0000 0000 0000 |
| TaxAmtCapablt | 0 |
| EcommTxnInd | “ “ / 03 (E3) |
| CustSvcPhoneNumber |  |
| EcommURL |  |

## – Testing

### - Introduction

### - Ed’s certification soapui project

So far the research that I did to the soapui project found out this:

The soap ui projects contains test requests in 2 categories, Utility transactions/RC related transaction. For RC related transaction, requests are grouped into 3 different merchant types, ECOMM/MOTO/RETAIL. Then for each merchant type, the test cases are grouped into credit card type and then for different type of transaction. Finally for different scenarios, the transaction can have different payload. So the request name tells a lot of information about the case it is trying to certify.

|  |  |  |
| --- | --- | --- |
| RC  ECOMM (E)  MOTO (M)  RETAIL (R) | AUTH (AMEX/DS/MC/VISA) | E:  cc\_no\_token\_do\_not\_save\_cc\_no\_cvv  E:   cc\_no\_token\_do\_not\_save\_cc\_with\_cvv  R:   cc\_no\_token\_do\_not\_save\_cc\_no\_track  R:   cc\_no\_token\_do\_not\_save\_cc\_with\_track  M:   cc\_no\_token\_do\_not\_save\_cc  E/M/R: cc\_no\_token\_save\_cc  E/M/R: token\_no\_cc  E/M/R: token\_and\_cc  E/M/R: cc\_decline  E/M/R: token\_decline  E/M/R: validation\_errors |
|  | AUTH\_CAPTURE (AMEX/DS/MC/VISA | E/M:  cc\_no\_token\_do\_not\_save\_cc  R:   cc\_no\_token\_do\_not\_save\_cc\_no\_track  R:   cc\_no\_token\_do\_not\_save\_cc\_with\_track  E/M/R: cc\_no\_token\_save\_cc  E/M/R: token\_no\_cc  E/M/R: token\_and\_cc  E/M/R: cc\_decline  E/M/R: token\_decline  E/M/R: validation\_errors |
|  | CAPTURE | E/M/R:  sucess  E/M/R: validation\_errors |
|  | REF | E/M/R:  cc\_no\_token\_do\_not\_save\_cc  E/M/R:  cc\_no\_token\_save\_cc  E/M/R:  token\_no\_cc  E/M/R:  token\_cc  E/M/R:  validation\_errors |
|  | REF\_BY\_SEQ | E/M/R:  sucess  E/M/R: validation\_errors |
|  | REVERSAL | E/M/R:  sucess  E/M/R: validation\_errors |
|  | VOID | E/M/R:  sucess  E/M/R: validation\_errors |
| Utility | DELETE\_CC |  |
|  | LOOK\_UP\_BY\_SEQ |  |
|  | LOOK\_UP\_BY\_TOKEN |  |
|  | MATCH\_BY\_SEQ |  |
|  | SAVE\_CC |  |

## 6 - Maintenance

### 6.1 - Error Codes

|  |  |  |
| --- | --- | --- |
| Interface return code | Msg |  |
| -1000 | INVALID USER NAME OR PASSWORD |  |
| -1001 | USER NOT AUTHORIZED |  |
| -1002 | UNABLE TO RETRIEVE CYBER CONFIG SETTINGS |  |
| -1100 | CREDIT CARD NUMBER IS INVALID    This error code is also used in validateCCTran SP. The message can be differnt for different reason. |  |
| -1003 | UNABLE TO LOCATE CC INFO USING PROVIDED TOKEN  XXXXXXXXXXXXX | This is for AUTH\_CAPTURE/SAVE\_CC/AUTH/REF, when token was provoide  But can't find the token related vault row.    ATTN: token is the vault ID , if DELETED column value is 'Y', you can also see this error. |
| -1500 | UNABLE TO VOID TRANSACTION THAT WAS PREVIOUSLY VOIDED - ' + @SeqNumber | @pay\_type = 'P' or @pay\_type = 'RV' or @pay\_type = 'VD' or @pay\_type = 'Q'    Query the ccTran table by sequence number , if the original sequence related transaction is void already, then this error will be returned.      select \*  from CC\_TRANS  where SeqNumber = @SeqNumber and  IsNull(TranState,'') = 'V' |
| -1021 | UNABLE TO ISSUE SUBSEQUENT TRANSACTION. UNABLE TO LOCATE VALID ORIGINAL CC INFO BY PARENT SEQUENCE NUMBER - ' + @SeqNumber |  |
|  |  |  |
| TBC |  |  |
|  |  |  |

### 6.2 - Client Applications

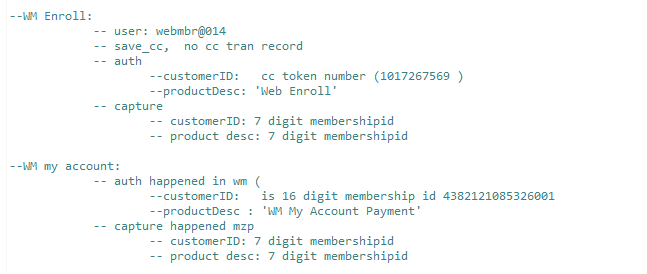
#### 6.2.1 - MZP

#### 6.2.2 - Web Member

Membership Enroll, the epayment interface service will be called directly for tokenization. The Membership MAV2 service will be called to AUTH/CAPTURE credit card in mzp enroll operation.

Membership my accounts, the epayment interface service will be called directly for authorization, the sequence number will be passed to MAV2 service to capture the fund. If there is any error returned to web member, then web member will call epayment service for VOID the AUTH with sequence number. (ATTN, reach out to Justin for a sample XML, somehow he gave me a non-interface xml. I checked the web member but it looks like wm is using the interface service)

**Note**: The following image shows cc tran record created by wm, in case there is a production issue and need to query the database really quick.



#### 6.2.3 - Data Scan:

This application is a mapforce generated .net app to process 2 flat files related to membership/member. Every row is a line of membership/member. The membership /member are linked together by a key value. Epayment interface application is called to store credit card information and return a token.

SAVE\_CC transaction is used and a token is generated.

<Datascan\_Consumer xsi:schemaLocation="http://aaamidatlantic.com/datascan/2009/06/04 Epay\_consumer.xsd" xmlns="http://aaamidatlantic.com/datascan/2009/06/04" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<Security>

<ConsumerID>MZP\_USER</ConsumerID>

<ConsumerToken>PWD</ConsumerToken>

<UserID>DATASCAN</UserID>

<TransactionID>SAVE\_CC</TransactionID>

</Security>

</Datascan\_Consumer>

The deployed exe file is on a pci scope server with a shared use name and password. User of this machine needs to sign a document for using this machine.

**Note**: Ed was saying in video 5 that Data scan is using the implementation service directly which probably is not the case per the map. Wei has more information for this.

#### 6.2.4 - Data Line

Wei knows detail information about this Dataline integration. I didn’t talk with Wei yet, but sounds like It does the reverse thing of Datascan. It go through the membership and find the token and do Token look up with epayment to get credit card information and put the cc info with membership info into a flat file.

#### 6.2.5 – Virtual Terminal

POS

CardUpdate

Charge ERS

CCSearch:

this is the source code for credit card search , learned the name from Preethi

Reporting

Report – Priyank and Raj has some report built for epayment system by using the Epayment\_Repl.

### 6.3 - Production Issues

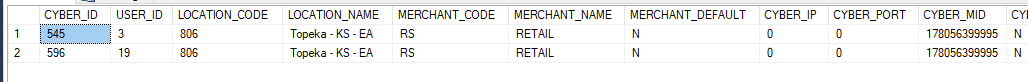
#### 6.3.1 - Add a new merchant

Any time there is an ACA owned new branch, or new contact center, with a new branch code and new location. In summary, any time there is a new branch got added to mzp.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| /\*  Add New ePay Merchant  • Company Name (MA, SJ, SP): MA  • Application Name (POS or MZP): MZP  • ePayment gateway login name: MZP\_USER  • \* Location Name: sample Millville  • \* Location Code: sample 001  • \* Merchant ID (12 digits numeric value) ---- sample 178000000000  • Merchant Type (MOTO, ECOMM, RETAIL, QUASI-CASH, or EMERGING MARKET) --- MOTO  • Merchant Code/MCC (8675, 6051, 6300, etc.) --- 8675  • Settlement Time (9pm, 10pm, or 1am) ---- 1am  • MAI Code (AAA MAT or AAA MEC) --- AAA MAT  • Nashville Merchant ID (MID) ---- optional value, provide if available  • \* Nashville Terminal ID (TID) ---- sample 5665527  \*/  exec usp\_ccTranRC\_AddMerchant  @COMPANY = 'MA', --MA/SJ/SP/IG  @APP\_NAME = 'MZP', --POS/MZP, etc.  @USER\_LOGIN = 'MZP\_USER', --POS\_USER/MZP\_USER, etc.  @LOCATION\_CODE = '103', --008,050, etc.  @LOCATION\_NAME = 'MARLTON (CCIT)', --WILMINGTON, CENTER CITY, etc.  @MERCHANT\_ID = '178056469996', --12 digits  @MERCHANT\_TYPE = 'MOTO', --RETAIL/MOTO/ECOMM/QUASICASH/EMERGINGMARKET  @MERCHANT\_CODE = '8675', --8675,6300,6051, etc  @SETTLE\_TIME = '10pm', --10pm, 1am  @MAI\_CODE = 'AAA MAT', --AAA MAT, AAA MEC  @NASHVILLE\_MID = '', --optional value  @NASHVILLE\_TID = '1735245' --7 digits | This stored procedure call is the entry place for a cluster of SP calls. It inserted a row to cyber and to aaa\_merchant table. Then it calls usp\_ccTranRC\_SwitchMerchantBatch to start the switch to rapid connect by preparing the hard coded values base on condition. Finally it inserts a role in MERCHANT\_SETTINGS\_RC table.  **IF FOR ANY REASON, NEED TO REFER TO EXISTING DATA FOR ADDING NEW MERCHANT, REFER TO PRODUCTION, THE TEST DATA MAY NOT BE EXACT.**   |  |  | | --- | --- | | **LOCATION\_CODE** | mzp branch code. | | **LOCATION\_NAME** | It is nice to match mzp branch name, but it’s ok that it doesn’t match. | | **MERCHANT\_ID** | Client need to obtain from pnc for the merchant id. It typically happens when store is opened.  This should be a brand new merchant id in the db, the stored procedure will check this merchant id. If exists, the execution will be stopped. | | **MERCHANT\_TYPE** | MOTO: stands for mail order telephone order.  It doesn’t need to be changed under most cases. MZP uses MOTO, WM uses ECOMM  This merchant type parameter will determine a lot of fields in cyber table: | | **MERCHANT\_CODE** | According to video 5, 32 minutes, this value 8675 should not be changed. | | **SETTLE\_TIME** | This is the settlement time that First data set for a certain merchant. 10 am is for MOTO, 1 pm is for ECOMM. | | **MAI\_CODE** | AAA MAT stands for mid atlantic, MOTO  AAA MEC stands for mid atlantic ecommerce, ECOMM | | **NASHVILLE\_MID** | Always blank, m stands for merchant id. | | **NASHVILLE\_TID** | T stands for terminal id. It goes with the merchant id as combination. Ed said merchant id is like a username, and Nashville\_tid is kind of like password. Both of them are needed for processing a transaction. | |

#### 6.3.2 – Add a new Entrepreneur Agent only.

Entrepreneur agents are not permanent employees, they are independent.



|  |  |
| --- | --- |
| --Add new EA merchant/branch to ePayment  --retrieve all EA merchant records  select \*  from cyber(nolock)  where [LOCATION\_NAME] like '%EA' -- all EA branches have an EA at the end of location name  --retrieve 2 merchant records belonging to specific EA branch, one assigned to MZP and second to WM  select \*  from cyber(nolock)  where cyber\_mid = '178056399995' and --main EA merchant id  location\_code = '806' --sample EA branch. Will return 2 records, 1st MZP and 2nd WM  --validate that each EA branch has 2 merchant records  select count(\*), location\_code, location\_name -- each EA branch should have 2 records (1 for mzp, 1 for wm)  from cyber(nolock)  where [LOCATION\_NAME] like '%EA'  group by location\_code, location\_name  --Insert 2 new merchant records for a new EA branch, use one of the exisiting EA branch records as a template  begin tran  INSERT INTO [dbo].[CYBER]  ([USER\_ID]  ,[LOCATION\_CODE]  ,[LOCATION\_NAME]  ,[MERCHANT\_CODE]  ,[MERCHANT\_NAME]  ,[MERCHANT\_DEFAULT]  ,[CYBER\_IP]  ,[CYBER\_PORT]  ,[CYBER\_MID]  ,[CYBER\_ECOMM]  ,[CYBER\_CARD\_PRESENT\_FLAG]  ,[CYBER\_TERMINAL\_CAPABILITY]  ,[CYBER\_TERMINAL\_TYPE]  ,[CYBER\_POS\_ENTRY\_MODE]  ,[CYBER\_CUSTOMER\_PRESENT\_FLAG]  ,[PAN\_READABLE]  ,[TRAN\_DUPS\_INTERVAL]  ,[MERCHANT\_RC\_FLAG]  ,[MERCHANT\_RC\_ACTIVE\_TS])  select  [USER\_ID]  ,'829' --replace with new branch code  ,'Mustang - OK - EA' --replace with new branch name, Format: "Branch Name - State - EA"  ,[MERCHANT\_CODE]  ,[MERCHANT\_NAME]  ,[MERCHANT\_DEFAULT]  ,[CYBER\_IP]  ,[CYBER\_PORT]  ,[CYBER\_MID]  ,[CYBER\_ECOMM]  ,[CYBER\_CARD\_PRESENT\_FLAG]  ,[CYBER\_TERMINAL\_CAPABILITY]  ,[CYBER\_TERMINAL\_TYPE]  ,[CYBER\_POS\_ENTRY\_MODE]  ,[CYBER\_CUSTOMER\_PRESENT\_FLAG]  ,[PAN\_READABLE]  ,[TRAN\_DUPS\_INTERVAL]  ,[MERCHANT\_RC\_FLAG]  ,getdate()  from cyber  where cyber\_mid = '178056399995' and --main EA merchant id  location\_code = '806' --sample EA branch to be used as a template. Will return 2 records, 1st MZP and 2nd WM  --rollback  commit | **NOTE: NEED TO VALIDATE THE CYBER TABLE DOESN’T HAVE RECORDS FOR THE LOCATION CODE THAT WE ARE GOING TO CREATE.**  Only need to replace the branch code and branch name in yellow highlighted area.  The main merchant with Rapid connect has been established already, we only need to clone the record in CYBER table for different location.  We have 50-100 ea branches set up in mzp, they are linked into one single merchant account.  Ed said the location name ends up with EA is a naming convention but not a requirement. He also mentioned if running the first select statement, you may find multiple merchant ids but it is ok.  2nd select query (combination of '806' with '178056399995' ) will return records like in above image. It has 2 records in return. One for mzp and one for WM.  The insert query will use the records from the 2nd query as template and create 2 more records with the new branch code |

#### 6.3.3 – Add a merchant in general.

If asked to add a new merchant, you need to ask back which type of merchant needs to be added.

In general, need to obtain the information from client and prepare the script and open the ticket for DBA to run the scripts in production.

#### 6.3.4 – production issue research in general

**cc\_tran table**

starts with this table. it has records for all credit card related transactions. Refer to 6.2.2 for additional information.

**Order id** is the payment key from mzp if it is a mzp payment.

To narrow down the records, can use customer id/created/amount/last4pan etc.

**TranResult**: 0 means end to end transaction with Rapid connect is successful, it doesn’t mean the payment is approved but means rapid connect returns ok. (Video 5 at 55 min) TranResult = 0 goes with ReturnMessage=”Success”.

If TransResult <>0, can refer to ReturnMessage for detail failure information.

**Trans\_Action** table,

it has information about the time epayment starts the transaction and the time to complete. If can’t find record in cc\_trans, try to find information in Trans\_Action.

Also it has a Tran\_Status column value to indicate the bank response if there is one.

Seq\_ID links the Trans\_Action table to the cc\_Trans

### 6.4 - Source Control Repository

Talk with Dan to gain SVN access: $SVN\_ROOT= <https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/>

If svn location get changed, please replace the $SVN\_ROOT value in the following links.

#### 6.4.1 - SVN: Implementation Service

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/services/implementations/ePayV2>

#### 6.4.2 - SVN: Interface Service Code:

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/services/interfaces/EPayment>

#### 6.4.3 - SVN: WSSS

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/services/implementations/WSSS>

#### 6.4.4 - SVN: wsdl and schema

Information needed for open the map, the original map for interface application need wsdl and schema to be at the correct location to open properly.

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/wsdl/electronicpayment>

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/schemas/electronicpayment>

#### 6.4.5 - SVN: Specification documentation for Interface Application

<https://mtlsubvrsn1.aaacorp.com/svn/aca-it/trunk/integrations/documents/specifications/interfaces/>

AAA Policy Administration System Integration - E-Payment Interface.doc

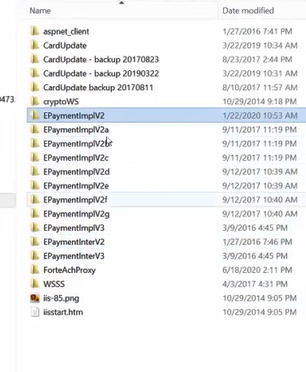
### 6.5 - Test Environment Change

#### 6.5.1 - RC test mode and rejection

## 7 - Other Information

### 7.1 - Other runtime in staging/production env.

From image below, you can see the impl has V2a –V2g. They are runtimes for V2. They are created to handle more requests from client application so the main runtime won’t have issue.



CardUpdate application has no direct relationship with Epayment, except it shares the same encryption mechanism/container etc.

### 7.2 – Test servers

Testing can be done to the implementation endpoint or interface endpoint. Please open incident ticket to Eric Burger for access to test windows server. Remote access to these servers are control by DUO, you need to access from laptop directly from home, can’t remote into a mini pc in company and then remote access to these servers. The user id that I used is “yahu”

#### 7.2.1 – test servers, .net

epaytoctstapp2 & epayhoctstapp2

#### 7.2.2 - Test Servers Database

Server: SQLDBTSTYELLOW\EPAY

Database: epayment

Window’s authentication

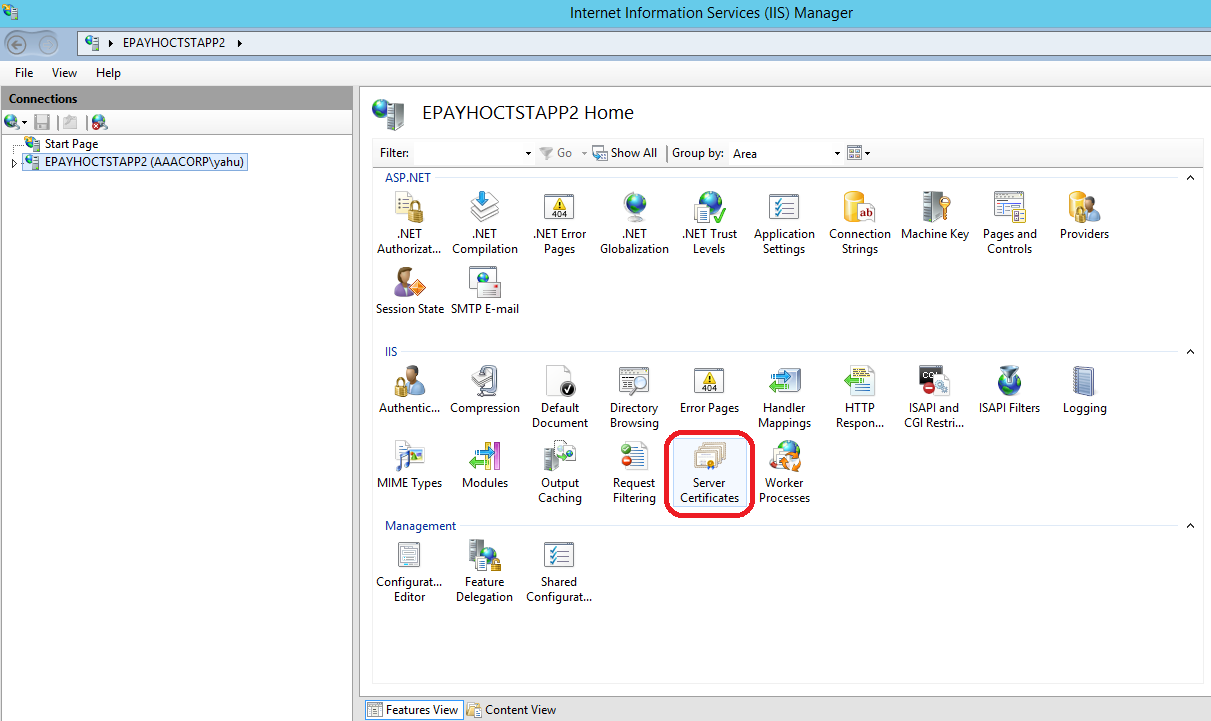
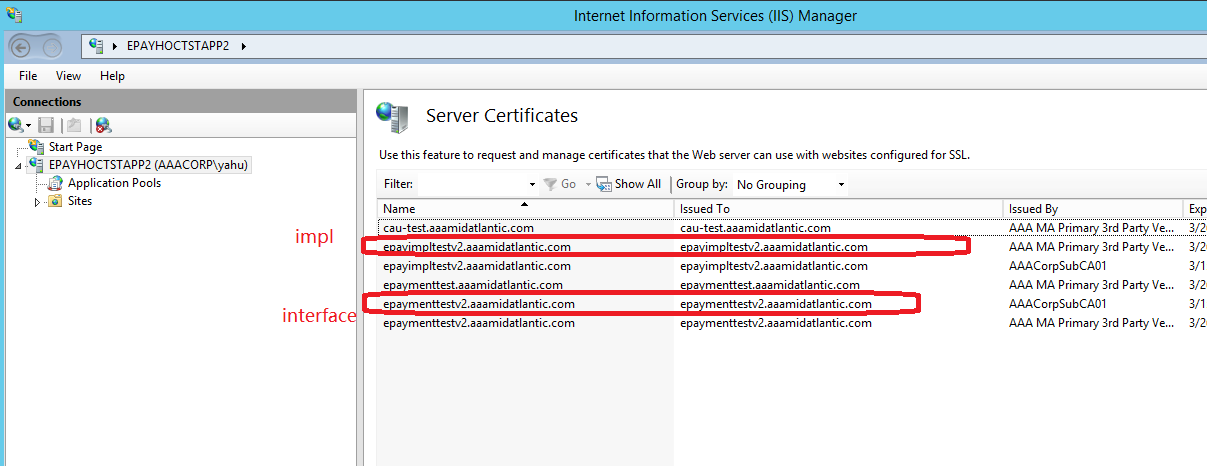
#### 7.2.3 - Interface endpoint:

<https://epaymenttestv2.aaamidatlantic.com/ElectronicPaymentService.asmx>

#### 7.2.4 - Implementation endpoint:

<https://epayimpltestv2.aaamidatlantic.com/epay.asmx>

Both http and https ports on staging server were blocked to client application, client application needs to use the KEMP advertised link to access to either interface service or implementation service endpoint above. Follow the screen shots below to see the impl/interface links on staging epayment server.

### 7.3 – production server

Please open incident ticket to Eric Burger for access to test windows server.

#### 7.3.1 – production servers, .net

Epaytocprodapp2 & Epayhocprodapp2

#### 7.3.2 - Database server:

Server: SQLDBPRODBLUE\EPAY

Database: epayment

Windows authentication

#### 7.3.3 – production interface endpoint

#### 7.3.4 – production implementation endpoint.

### 7.4 – Testing CC account

#### Accepts

VISA

4012000033330026 (04/2016)

4111111111111111

AMEX

373953192351004

DISC

6011000990099818

MC

5424180279791732

#### Reject

VISA 4012000033330026 - $29.95

MC 5424180279791732 - $29.95

AMX 373953192351004 - $31.00