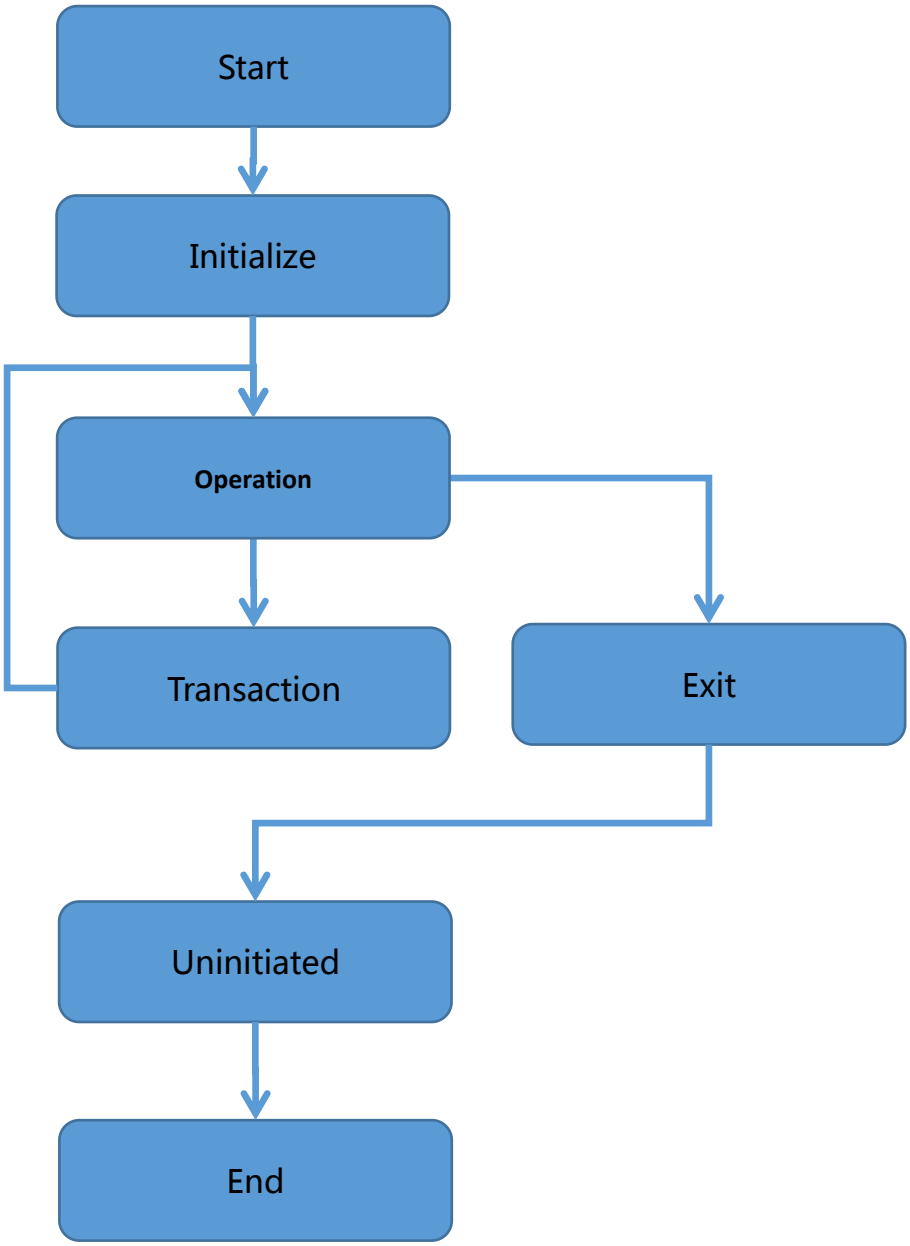
NEW POS Technology Limited
EMV Level2 Kernel-C SDK User Manual
V1.2

Contents

Document History

Version	Update Date	Author	Descriptions
V1.0	2019/4/11	Harrison Lee	Draft
V1.1	2019/6/5	Harrison Lee	Add callback function for the PAN check.
V1.2	2019/10/28	Harrison Lee	Add “param” other to Application selection callback function

1. Instructions



1.1. Initialize

Process	description
EMV_CALLBACKS_T callbacks EMV_FRAME_WORK fw; EMV_ENTRY_POINT ep ;	Define several variable callbacks, fw, ep.
Step1: cb_callbacks_get(exe, &callbacks);	Initiation callbacks, Get callback function pointer
Step2: fw = emv_fw_init(&callbacks);	Initiation fw pointer
Step3: if(fw != NULL) ep = emv_ep_init(fw);	Initiation ep pointer
Step4: load the kernel function by emv_ep_kernel_load().	Load the kernel function to ep->m_kernels list
Step5: load kernel parameter emv_ep_kernel_param_set(ep, param, param_len);	
Step5: if above all is success, start transaction	Start transaction process
Note: if have any failed on above, exit program at once.	

1.2. Transaction process

1.2.1. Contact Transaction process

Process	description

Step1: Initiation process
emv_error_indication_init();
emv_outcome_param_init();
emv_user_interface_reqeust_data_init();
	Initiation process

Step2: emv_ep_pre_transaction()	Previous process before start transaction
Step3: Polling card	
Step4: emv_ep_contact_build_candidate_list(ep, EMV_FALSE, EMV_TRUE);	Build application candidate list
Step5:emv_ep_contact_application_select(ep);	Select application
Step6:emv_ep_contact_initiate_application(ep)	Initiation application(GPO)
Step6:emv_ep_contact_read_application_data(ep)	Read application data(read record)
Step7:emv_ep_contact_data_authentcation(ep)	Offline data authentication(SDA or DDA or CDA)
Step8:emv_ep_contact_prcess_restriction(ep)	Restriction process(check application version,

	application effective&expired date check)
Step8:emv_ep_contact_cardholder_verification(ep)	CVM perform
Step9:emv_ep_contact_teminal_risk_management(ep)	Terminal risk management
Step10:emv_ep_contact_terminal_action_analysis(ep)	Terminal action analyze, according to previous process outcome to decide terminal decline or approve the transaction
Step10:emv_ep_contact_card_action_analysis(ep)	Card action analyze,according to terminal's process outcome, card to decide the transaction should be approved, declined or online process.
Step11: if above process outcome is online Emv_ep_contact_completion(ep) Else goto step12	if card decide online, terminal processing online and then complete transaction
Step12: end transaction	Close polling card Display transaction outcome

1.2.2. Contactless Transaction process

Process	**description**

Step1: Initiation process
emv_error_indication_init();
emv_outcome_param_init();
emv_user_interface_reqeust_data_init();
	Initiation process

Step2: emv_ep_pre_transaction()	Previous process before start transaction
Step3: Polling card	
Step4: emv_ep_contactless_build_combination()	Build application candidate list
Step5: emv_ep_contactless_transaction()	Contactless transaction process
Step6: if above process outcome is online auth. emv_ep_contactless_transaction_completion() Else goto step7	if card decide online, terminal processing online and then complete transaction
Step7: end transaction	Close polling card Display transaction outcome

1.3. Uninitiated

Process	description
Step1: emv_ep_free(exe->ep);	Free the ep point memory
Step2:emv_fw_free(exe->fw);	Free the fw point memory

2. Framework

2.1. Callback Functions

2.1.1. FN_CB_LOG_ENTER

Prototype	typedef EMV_VOID (*FN_CB_LOG_ENTER)(EMV_PROGRAM callback_program)	
Description	printf the log for enter(Each time the print message is called, it is indented backwards)	
Parameters	callback_program	handle
	fmt	
Return Value	None	

Example	EMV_VOID cb_log_enter(EMV_PROGRAM exe)
	{
	if(NULL != exe){
	exe = (EMV_PROGRAM)exe;
	exe->m_log_level++;
	}
	}	

2.1.2. FN_CB_LOG_LEAVE

Prototype	typedef EMV_VOID (*FN_CB_LOG_LEAVE)(EMV_PROGRAM callback_program)	
Description	printf the log for leave(Each time the print message is called, it is indented cancel)	
Parameters	callback_prog	handle

	ram	
	fmt	
Return Value	None	

Example	EMV_VOID cb_log_leave(EMV_PROGRAM exe)
	{
	if(NULL != exe){
	if (exe->m_log_level > 0)
	exe->m_log_level--;
	}
	}	

2.1.3. FN_CB_LOG_PRINTF

Prototype	typedef EMV_VOID (*FN_CB_LOG_PRINTF)(EMV_PROGRAM callback_program, EMV_CHAR_CPTR fmt,...)	
Description	printf the log	
Parameters	callback_prog ram	handle
	fmt	
Return Value	None	
Example		

2.1.4. FN_CB_LOG_PRINT_MEMORY

Prototype	Typedef EMV_VOID (*FN_CB_LOG_PRINT_MEMORY)(EMV_PROGRAM callback_program, EMV_CHAR_CPTR fmt, EMV_VOID_CPTR data, EMV_UINT data_len)	
Description	printf the log for memory	
Parameters	callback_prog fmt	handle
	data	
	data_len	
Return Value	None	
Example		

2.1.5. FN_CB_SYS_IFD_SERIAL_NUMBER_GET

Prototype	typedef EMV_BOOL (*FN_CB_SYS_IFD_SERIAL_NUMBER_GET)(EMV_PROGRAM callback_program, EMV_CHAR_PTR buff, EMV_UINT buff_size);	
Description	Get the terminal serial number	
Parameters	callback_program	handle
	buff	The function full the serial number to buff
	buff_size	
Return Value	EMV_TRUE:Succeed EMV_FAISE:false	
Example		

2.1.6. FN_CB_SYS_CURRENT_TIME_GET

Prototype	typedef EMV_BOOL (*FN_CB_SYS_CURRENT_TIME_GET)(EMV_PROGRAM callback_program, EMV_TIME_PTR current_time);	
Description	Get the terminal serial number	
Parameters	callback_program	handle
	current_time	The function full the current time to buff
Return Value	EMV_TRUE:Succeed EMV_FAISE:false	
Example		

2.1.7. FN_CB_SYS_MSLEEP

Prototype	typedef EMV_VOID (*FN_CB_SYS_MSLEEP)(EMV_PROGRAM callback_program, EMV_UINT mseconds)	
Description		
Parameters	callback_prog	handle

	ram	
	mseconds	
Return Value	None	
Example		

2.1.8. FN_CB_SYS_GET_TIMESTAMP

Prototype	EMV_VOID (*FN_CB_SYS_GET_TIMESTAMP)(EMV_PROGRAM exe, EMV_TIMESTAMP_PTR timeStamp);	
Description	Get the terminal tick count	
Parameters	callback_program	handle
	timeStamp	typedef struct{ EMV_TICKCOUNT m_seconds; EMV_TICKCOUNT m_useconds; }EMV_TIMESTAMP_T
Return Value		
Example		

2.1.9. FN_CB_SYS_TRANS_SEQ_NUMBER

Prototype	typedef EMV_VOID (*FN_CB_SYS_TRANS_SEQ_NUMBER)(EMV_PROGRAM callback_program, EMV_UINT_PTR trans_seq_number);	
Description	Get the transaction seq number	
Parameters	callback_program	handle
	trans_seq_number	The transaction seq number full to trans_seq_number
Return Value	None	
Example		

2.1.10. FN_CB_SYS_TRANS_AMOUNT_TOTAL_GET

Prototype	typedef EMV_VOID (*FN_CB_SYS_TRANS_AMOUNT_TOTAL_GET)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR pan, EMV_UINT pan_len, EMV_BYTE_CPTR seq, EMV_UINT seq_len, EMV_AMOUNT_PTR total_amount);	
Description	get the transaction data for total amount pan and seq.	
Parameters	callback_program	handle
	pan	
	pan_len	
	seq	
	seq_len	
	total_amount	
Return Value	None	
Example		

2.1.11. FN_CB_SYS_BREAK_DETECT

Prototype	typedef EMV_BREAK_SOURCE (*FN_CB_SYS_BREAK_DETECT)(EMV_PROGRAM callback_program)	
Description	Break the contactless transaction,check for contact and magnetic stripe	
Parameters	callback_program	handle
Return Value	Return the break source for CONTACT, SWIPE or CANCEL	
Example		

2.1.12. FN_CB_HSM_RANDOM_NUMBER_GET

Prototype	typedef EMV_BOOL (*FN_CB_HSM_RANDOM_NUMBER_GET)(EMV_PROGRAM callback_program, EMV_VOID_PTR buff, EMV_UINT len);

Description	Get the terminal random number	
Parameters	callback_program	handle
	buff	
	len	
Return Value	EMV_TRUE:Succeed EMV_FAISE:false	
Example		

2.1.13. FN_CB_HSM_SHA1_INIT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_SHA1_INIT)(EMV_PROGRAM callback_program, EMV_SHA1_CTX ctx)	
Description	Initialization the hash data for sha1	
Parameters	callback_program	handle
	ctx	
Return Value	EMV_TRUE:Succeed EMV_FAISE:false	
Example		

2.1.14. FN_CB_HSM_SHA1_UPDATE

Prototype	typedef EMV_VOID (*FN_CB_HSM_SHA1_UPDATE)(EMV_PROGRAM callback_program, EMV_SHA1_CTX ctx, EMV_VOID_CPTR data, EMV_UINT data_len)	
Description	update the hash data for sha1	
Parameters	callback_program	handle
	ctx	
	data	
	data_len	
Return Value	None	
Example		

2.1.15. FN_CB_HSM_SHA1_FINAL

Prototype	typedef EMV_VOID (*FN_CB_HSM_SHA1_FINAL)(EMV_PROGRAM callback_program, EMV_SHA1_CTX ctx, EMV_BYTE hash[20]);	
Description	Final the hash data for sha1	
Parameters	callback_program	handle
	ctx	
	hash[20]	Return the final hash data to hash[20]
Return Value		
Example		

2.1.16. FN_CB_HSM_SM3_INIT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_SM3_INIT)(EMV_PROGRAM callback_program, EMV_SM3_CTX ctx);	
Description	Init the hash data for SM3	
Parameters	callback_program	handle
	ctx	
Return Value	EMV_TRUE:succeed EMV_FALSE:false	
Example		

2.1.17. FN_CB_HSM_SM3_UPDATE

Prototype	typedef EMV_VOID (*FN_CB_HSM_SM3_UPDATE)(EMV_PROGRAM callback_program, EMV_SM3_CTX ctx, EMV_VOID_CPTR data, EMV_UINT data_len);	
Description	Updata the hash data for SM3	
Parameters	callback_program	handle

	ctx	
	data	
	data_len	
Return Value	None	
Example		

2.1.18. FN_CB_HSM_SM3_FINAL

Prototype	typedef EMV_VOID (*FN_CB_HSM_SM3_FINAL)(EMV_PROGRAM callback_program, EMV_SM3_CTX ctx, EMV_BYTE hash[32]);	
Description	Final the hash data for SM3	
Parameters	callback_program	handle
	ctx	
	hash[32]	Return the final hash data to hash[20]
Return Value	None	
Example		

2.1.19. FN_CB_HSM_DES_ENCRYPT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_DES_ENCRYPT)(EMV_PROGRAM callback_program, EMV_DES_MODE mode, EMV_BYTE_CPTR key, EMV_UINT key_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out);	
Description	Encrypt data by the DES algorithm	
Parameters	callback_program	handle
	mode	The mode required for encryption
	key	The key required for encryption
	key_len	The key length
	data_in	

	data_in_len	
	data_out	
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.20. FN_CB_HSM_DES_DECRYPT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_DES_DECRYPT)(EMV_PROGRAM callback_program, EMV_DES_MODE mode, EMV_BYTE_CPTR key, EMV_UINT key_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out)	
Description	Decrypt data by the DES algorithm	
Parameters	callback_program	handle
	mode	The mode required for decrypt
	key	The key required for decrypt
	key_len	The key length
	data_in	
	data_in_len	
	data_out	
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.21. FN_CB_HSM_AES_ENCRYPT

Prototype	Typedef EMV_BOOL (*FN_CB_HSM_AES_ENCRYPT)(EMV_PROGRAM callback_program, EMV_AES_MODE mode, EMV_BYTE_CPTR key, EMV_UINT key_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out

	)	
Description	Encrypt data by the AES algorithm	
Parameters	callback_program	handle
	mode	The mode required for decrypt
	key	The key required for decrypt
	key_len	The key length
	data_in	
	data_in_len	
	data_out	
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.22. FN_CB_HSM_AES_DECRYPT

Prototype	Typedef EMV_BOOL (*FN_CB_HSM_AES_ENCRYPT)(EMV_PROGRAM callback_program, EMV_AES_MODE mode, EMV_BYTE_CPTR key, EMV_UINT key_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out)	
Description	Encrypt data by the AES algorithm	
Parameters	callback_program	handle
	mode	The mode required for decrypt
	key	The key required for decrypt
	key_len	The key length
	data_in	
	data_in_len	
	data_out	
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.23. FN_CB_HSM_RSA_PUBLIC_ENCRYPT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_RSA_PUBLIC_ENCRYPT)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR modules, EMV_UINT modules_len, EMV_BYTE_CPTR exponents, EMV_UINT exponents_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out, EMV_UINT_PTR data_out_len);	
Description	Encrypt data by the RSA algorithm	
Parameters	callback_program	handle
	mode	The mode required for decrypt
	modules_len	The modules length
	key	The key required for decrypt
	key_len	The key length
	data_in	Input data
	data_in_len	Input data length
	data_out	Output data
	data_out_len	Output data length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.24. FN_CB_HSM_RSA_PRIVATE_DECRYPT

Prototype	typedef EMV_BOOL (*FN_CB_HSM_RSA_PRIVATE_DECRYPT)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR modules, EMV_UINT modules_len, EMV_BYTE_CPTR private_exponents, EMV_UINT private_exponents_len, EMV_BYTE_CPTR public_exponents, EMV_UINT public_exponents_len, EMV_BYTE_CPTR data_in, EMV_UINT data_in_len, EMV_BYTE_PTR data_out, EMV_UINT_PTR data_out_len)

Description	Decrypt data by the RSA algorithm	
Parameters	callback_program	handle
	mode	The mode required for decrypt
	modules_len	The modules length
	key	The key required for decrypt
	key_len	The key length
	data_in	Input data
	data_in_len	Input data length
	data_out	Output data
	data_out_len	Output data length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

typedef EMV_VOID(*FN_CB_HSM_SM2_GET_ZA)(EMV_PROGRAM exe,
const EMV_BYTE pk_x[32],
const EMV_BYTE pk_y[32],
unsigned char za[32]);

2.1.25. FN_CB_HSM_SM2_GET_ZA

Prototype	typedef EMV_VOID(*FN_CB_HSM_SM2_GET_ZA)(EMV_PROGRAM exe, const EMV_BYTE pk_x[32], const EMV_BYTE pk_y[32], unsigned char za[32]);	
Description	SM2, get ZA	
Parameters	callback_program	handle
	pk_x[32]	
	pk_y[32]	
	za[32]	
Return Value		
Example		

2.1.26. FN_CB_HSM_SM2_VERIFY

Prototype	typedef EMV_BOOL (*FN_CB_HSM_SM2_VERIFY)(EMV_PROGRAM callback_program, const EMV_BYTE pk_x[32], const EMV_BYTE pk_y[32], const EMV_BYTE digit[32], const EMV_BYTE r[32], const EMV_BYTE s[32])	
Description	Verify the data to RSA algorithm	
Parameters	callback_program	handle
	pk_x[32]	
	pk_y[32]	
	digit[32]	
	r[32]	
	s[32]	
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.27. FN_CB_CT_APDU_EXCHANGE

Prototype	typedef EMV_BOOL (*FN_CB_CT_APDU_EXCHANGE)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR apdu, EMV_UINT apdu_len, EMV_BYTE_PTR resp, EMV_UINT_PTR resp_len)	
Description	The apdu exchange to contact	
Parameters	callback_program	handle
	apdu	Send the data to card
	apdu_len	The data length of send
	resp	The data return by the card
	resp_len	The data length of resp
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.28. FN_CB_CL_APDU_EXCHANGE

Prototype	typedef EMV_BOOL (*FN_CB_CL_APDU_EXCHANGE)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR apdu, EMV_UINT apdu_len, EMV_BYTE_PTR resp, EMV_UINT_PTR resp_len);	
Description	The apdu exchange to contactless.	
Parameters	callback_program	handle
	apdu	Send the data to card
	apdu_len	The data length of send
	resp	The data return by the card
	resp_len	The data length of resp
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.29. FN_CB_CL_APDU_SEND

Prototype	typedef EMV_BOOL (*FN_CB_CL_APDU_SEND)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR apdu, EMV_UINT apdu_len);	
Description	The contactless send apdu to card.	
Parameters	callback_program	handle
	apdu	Send the data to card
	apdu_len	Data length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.30. FN_CB_CL_APDU_RESP_GET

Prototype	typedef EMV_BOOL (*FN_CB_CL_APDU_RESP_GET)(EMV_PROGRAM callback_program, EMV_BOOL_PTR isdone,

	EMV_BYTE_PTR resp, EMV_UINT_PTR resp_len)	
Description	The contactless response card apdu reply data.	
Parameters	callback_program	handle
	isdone	Card send the apdu data to contactless
	resp	Card response data
	resp_len	length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.31. FN_CB_APP_PARAM_COUNT_GET

Prototype	Typedef EMV_BOOL (*FN_CB_APP_PARAM_COUNT_GET)(EMV_PROGRAM callback_program, EMV_UINT_PTR count)	
Description	Get the app parameter count.	
Parameters	callback_program	handle
	count	Application parameter count
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.32. FN_CB_APP_PARAM_GET

Prototype	typedef EMV_BOOL (*FN_CB_APP_PARAM_GET)(EMV_PROGRAM callback_program, EMV_UINT app_index, EMV_KERNEL_ID_PTR kernel_id, EMV_TRANS_TYPE_PTR trans_type, EMV_AID aid, EMV_UINT_PTR aid_len, EMV_BOOL_PTR asi, EMV_BYTE_PTR app_param, EMV_UINT_PTR app_param_len);	
Description	Get the app param data.	
Parameters	callback_program	handle

	app_index	Index of application parameter
	kernel_id	Kernel id
	trans_type	Transaction type
	aid	Get marched AID buff
	aid_len	AID length
	asi	Weather support partial march
	app_param	Application parameter
	app_param_len	Application parameter length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.33. FN_CB_CAPK_GET

Prototype	typedef EMV_BOOL (*FN_CB_CAPK_GET)(EMV_PROGRAM callback_program, const EMV_BYTE rid[5], EMV_BYTE data_index, EMV_PUBLIC_KEY_TYPE type, EMV_PUBLIC_KEY_PTR key)	
Description	Get the capk data.	
Parameters	callback_program	handle
	rid[5]	RID
	data_index	Capk index
	type	EMV_RSA, EMV_SM2

	key	typedef struct{
		EMV_PUBLIC_KEY_TYPE m_key_type;
		union{
		EMV_PUBLIC_KEY_RSA m_rsa;
		EMV_PUBLIC_KEY_SM2 m_sm2;
		};
		}EMV_PUBLIC_KEY;

Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.34. FN_CB_IPKC_REVOCK_CHECK

Prototype	typedef EMV_BOOL (*FN_CB_IPKC_REVOCK_CHECK)(

	EMV_PROGRAM callback_program, const EMV_BYTE rid[5], EMV_BYTE data_index, const EMV_BYTE serial_num[3]);	
Description	Check the data in the revock file.	
Parameters	callback_program	handle
	rid[5]	RID
	data_index	index
	serial_num	Serial number
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.35. FN_CB_EXCEPTION_FILE_CHECK

Prototype	typedef EMV_BOOL (*FN_CB_EXCEPTION_FILE_CHECK)(EMV_PROGRAM callback_program, EMV_BYTE_CPTR pan, EMV_UINT pan_len, EMV_BYTE_CPTR pan_seq, EMV_UINT pan_seq_len);	
Description	Check the exception data in the revock file.	
Parameters	callback_program	handle
	pan	
	pan_len	
	pan_seq	
	pan_seq_len	
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.36. FN_CB_UI_DISPLAY_PROCESSING

Prototype	typedef EMV_VOID (*FN_CB_UI_DISPLAY_PROCESSING)(EMV_PROGRAM callback_program);	
Description	Displayed the "PROCESSING..." on the scean.	
Parameters	callback_program	handle

Return Value	None
Example	

2.1.37. FN_CB_UI_APPLICATION_SELECT

Prototype	typedef EMV_BOOL (*FN_CB_UI_APPLICATION_SELECT)(EMV_PROGRAM callback_program, EMV_BOOL contactless, EMV_UINT app_count, EMV_KERNEL_ID_CPTR kernels, EMV_CHAR_CPPTR aids, EMV_CHAR_CPPTR lables, EMV_UINT_CPPTR other_tags_len, EMV_BYTE_CPPTR other_tags, EMV_UINT_PTR Selected);	
Description	Multi aoolication card,user selects the trading application.	
Parameters	callback_program	handle
	contactless	Weather is contactless application
	app_count	Application count
	kernels	
	aids	applications
	lables	Application label
	Ohter_tags_len	Length of ther tags, no defined in the kernel
	Other_tags	Other tags, no defined in the kernel
	Selected	Have selected application
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.38. FN_CB_UI_CARDHOLDER_CONFIRM

Prototype	typedef EMV_BOOL (*FN_CB_UI_CARDHOLDER_CONFIRM)(EMV_PROGRAM callback_program, EMV_CHAR_CPTR aid, EMV_CHAR_CPTR lable)
Description	Display the aid message and the cardholder confirm the

	transaction.	
Parameters	callback_program	handle
	aid	AID
	lable	Application label
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.39. FN_CB_UI_LANGUAGE_SELECT

Prototype	typedef EMV_BOOL (*FN_CB_UI_LANGUAGE_SELECT)(EMV_PROGRAM callback_program, EMV_CHAR_CPTR languages);	
Description	Cardholder select the transaction language	
Parameters	callback_program	handle
	languages	Language set
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.40. FN_CB_UI_CREDENTIALS_CHECK

Prototype	typedef EMV_BOOL (*FN_CB_UI_CREDENTIALS_CHECK)(EMV_PROGRAM callback_program, EMV_BYTE type, EMV_CHAR_CPTR number, EMV_BOOL_PTR Confirmed)	
Description	Check the credentials	
Parameters	callback_program	
	type	The credentials type
	number	credentials number
	Confirmed	
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.41. FN_CB_UI_PAN_CONFIRM

Prototype	typedef EMV_BOOL (*FN_CB_UI_PAN_CONFIRM)(EMV_PROGRAM callback_program, EMV_CHAR_CPTR pan);	
Description	Display the pan number and ask the operator to confirm the pan number.	
Parameters	callback_program	
	pan	Pan number
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.42. FN_CB_PED_PIN_VERIFY_STATUS_SHOW

Prototype	typedef EMV_BOOL (*FN_CB_PED_PIN_VERIFY_STATUS_SHOW)(EMV_PROGRAM callback_program, EMV_BOOL verify_success, EMV_BYTE pin_try_counter)	
Description	Sheow the ped pin verify status.	
Parameters	callback_program	handle
	verify_success	EMV_TRUE: pin Verify Succeed EMV_FAISE:pin verify false
	pin_try_counter	0:pin locked Other: pin counter
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.43. FN_CB_PED_PLAINTEXT_PIN_VERIFY

Prototype	typedef EMV_BOOL (*FN_CB_PED_PLAINTEXT_PIN_VERIFY)(EMV_PROGRAM callback_program, EMV_BOOL_PTR bypass, EMV_SW_PTR sw);
Description	

Parameters	callback_program	handle
	bypass	Function return to bypass EMV_TRUE: accroding to bypass EMV_FALSE: not bypass
	sw	
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.44. FN_CB_PED_ENCIPHER_PIN_VERIFY

Prototype	typedef EMV_BOOL (*FN_CB_PED_ENCIPHER_PIN_VERIFY)(EMV_PROGRAM callback_program, const EMV_BYTE random[8], EMV_BYTE_CPTR modules, EMV_UINT modules_len, EMV_BYTE_CPTR exponents, EMV_UINT exponents_len, EMV_BOOL_PTR bypass, EMV_SW_PTR sw)	
Description		
Parameters	callback_program	handle
	random[8]	Random for encipher
	modules	Modules for encipher
	modules_len	Modules length
	exponents	Exponents for encipher
	exponents_len	Exponents length
	bypass	EMV_TRUE:not entry Pin EMV_FALSE: have entry pin
	sw	
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.45. FN_CB_PED_ONLINE_PIN_ENTER

Prototype	typedef EMV_BOOL (*FN_CB_PED_ONLINE_PIN_ENTER)(EMV_PROGRAM callback_program,

	EMV_CHAR_CPTR pan, EMV_BOOL_PTR bypass)	
Description		
Parameters	callback_program	handle
	pan	
	bypass	EMV_TRUE:not entry Pin EMV_FALSE: have entry pin
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.1.46. FN_CB_TORN_RECORD_SEND

Prototype	typedef EMV_BOOL (*FN_CB_TORN_RECORD_SEND)(EMV_PROGRAM callback_program, EMV_KERNEL_ID kernel_id, EMV_UINT total, EMV_UINT record_index, EMV_BYTE_CPTR record, EMV_UINT record_len)	
Description	Send the torn record to host	
Parameters	callback_program	handle
	kernel_id	Kernel ID
	total	Record count of sum
	record_index	Record index
	record	Record buff
	record_len	Record length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.47. FN_CB_TORN_RECORD_SAVE

Prototype	typedef EMV_BOOL (*FN_CB_TORN_RECORD_SAVE)(EMV_PROGRAM callback_program, EMV_KERNEL_ID kernel_id, EMV_UINT record_index, EMV_VOID_CPTR record,

	EMV_UINT record_len)	
Description	Save the torn record to terminal	
Parameters	callback_program	handle
	kernel_id	Kernel ID
	record_index	Record index
	record	Record buff need to save
	record_len	Record length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.48. FN_CB_DEK_SEND

Prototype	typedef EMV_BOOL(*FN_CB_DEK_SEND)(EMV_PROGRAM callback_program, EMV_KERNEL_ID kernel_id, EMV_BYTE_CPTR data, EMV_UINT data_len)	
Description		
Parameters	callback_program	handle
	kernel_id	Kernel ID
	data	Data to send
	data_len	Data length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.49. FN_CB_DET_GET

Prototype	typedef EMV_BOOL(*FN_CB_DET_GET)(EMV_PROGRAM callback_program, EMV_KERNEL_ID kernel_id, EMV_BYTE_PTR data, EMV_UINT_PTR data_len)	
Description		
Parameters	callback_program	handle
	kernel_id	Kernel ID

	data	The data buff for DET
	data_len	Actual data length
Return Value	EMV_TRUE:succeed EMV_FALSE:failed	
Example		

2.1.50. FN_CB_USER_REQUEST_INTERFACE_SEND

Prototype	typedef EMV_VOID (*FN_CB_USER_REQUEST_INTERFACE_SEND) (EMV_PROGRAM callback_program, EMV_KERNEL_ID kernel_id, EMV_USER_INTERFACE_REQUEST_DATA_CPTR user_interface_request_data)	
Description		
Parameters	callback_program	handle
	kernel_id	Kernel ID
	user_interface_request_data	User interface request data
Return Value	None	
Example		

2.1.51. FN_CB_DRL_COUNT_GET

Prototype	typedef EMV_UINT (*FN_CB_DRL_COUNT_GET)(EMV_PROGRAM exe, EMV_KERNEL_ID kernel_id, EMV_UINT type)	
Description		
Parameters	exe	handle
	Kernel_id	Kernel ID
	type	
Return Value	Return the DRL count	
Example		

2.1.52. FN_CB_DRL_GET

Prototype	typedef EMV_BOOL (*FN_CB_DRL_GET)(EMV_PROGRAM exe, EMV_KERNEL_ID kernel_id, EMV_UINT type, EMV_UINT rec_no, EMV_BYTE_PTR param, EMV_UINT_PTR param_len)	
Description		
Parameters	exe	handle
	kernel_id	Kernel ID
	type	
	rec_no	DRL Record number
	param	DRL parameter
	param_len	Parameter length
Return Value	EMV_TRUE:0 EMV_FALSE:1	
Example		

2.2. Framework API

2.2.1. emv_fw_version_get

Prototype	EMV_CHAR_CPTR emv_fw_version_get(EMV_VOID);	
Description	Get version number of the EMV Framework.	
Parameters		
Return Value	String of version of Framework.	
Example		

2.2.2. emv_fw_version_time_get

Prototype	EMV_CHAR_CPTR emv_fw_version_time_get(EMV_VOID);	
Description	Get the date and time of the Framework.	
Parameters		
Return Value	String of version of Framework.	
Example		

2.2.3. emv_fw_init

Prototype	EMV_FRAME_WORK emv_fw_init(EMV_CALLBACKS_PTR Callbacks);	
Description	Initialize EMV Framework and get instantiate of the module.	
Parameters	Callbacks	Callback functions from the application program.
Return Value	NULL:Initialize failed Other:Initialize successful	
Example		

2.2.4. emv_fw_free

Prototype	EMV_VOID emv_fw_free(EMV_FRAME_WORK fw);	
Description	Free instance handle of the EMV Framework.	
Parameters	fw	Return value of the Framework.
Return Value	None	
Example		

2.3. Framework database API

2.3.1. emv_database_value_set

Prototype	EMV_ERROR emv_database_value_set(EMV_FRAME_WORK fw, EMV_DATA_SOURCE data_source, EMV_TAG templete, EMV_TAG tag, EMV_VOID_CPTR value, EMV_UINT value_len, EMV_BOOL OverWrite);	
Description	Set tag value to database	
Parameters	fw	handle
	data_source	enum{ EMV_DATA_SOURCE_ICC =0x01, EMV_DATA_SOURCE_TERMINAL=0x02, EMV_DATA_SOURCE_ISSUER =0x04,

		EMV_DATA_SOURCE_ISSUER_TERMINAL=0x06, EMV_DATA_SOURCE_KERNEL=0x08, EMV_DATA_SOURCE_ALL=0x0F }EMV_DATA_SOURCE;
	templete	template
	tag	tag
	value	Value of tag
	value_len	Value length
	OverWrite	Weather allow to over write
Return Value	EMV_OK: set success Other: set failed	
Example		

2.3.2. emv_database_value_get

Prototype	EMV_ERROR emv_database_value_get(EMV_FRAME_WORK fw, EMV_TAG tag, EMV_VOID_PTR value, EMV_UINT_PTR value_len);	
Description	get tag value to database, must malloc memory for value	
Parameters	fw	handle
	tag	tag
	value	Point tag value memory
	value_len	Point to actual value length
Return Value	EMV_OK: set success Other: set failed	
Example	EMV_BYTE value[2]; emv_database_value_get(paypass->m_fw, TAG_MC_CVC3_TRACK2, value, &len);	

2.3.3. emv_database_value_get_ex

Prototype	EMV_ERROR emv_database_value_get_ex(EMV_FRAME_WORK fw, EMV_TAG tag, EMV_BYTE_CPPTR value, EMV_UINT_PTR value_len);
Description	get tag value to database, no need malloc memory for *value

Parameters	fw	handle
	tag	tag
	value	Point the pointer that Point tag value memory
	value_len	Point to actual value length
Return Value	EMV_OK: set success Other: set failed	
Example	EMV_BYTE_CPTR value=NULL; emv_database_value_get_ex(jcb->m_fw, TAG_TRACK_2_EQUIVALENT_DATA, &value, &len);	

2.4. Framework other API

2.4.1. emv_outcome_param_init

Prototype	EMV_VOID emv_outcome_param_init(EMV_FRAME_WORK fw, EMV_OUTCOME_PARAM outcome);	
Description	get tag value to database, must malloc memory for value	
Parameters	fw	handle
	outcome	Store the contactless transaction outcome
Return Value	None	
Example		

2.4.2. emv_user_interface_reqeust_data_init

Prototype	EMV_VOID emv_user_interface_reqeust_data_init(EMV_FRAME_WORK fw, EMV_USER_INTERFACE_REQUEST_DATA user_interface_request_data);	
Description	get tag value to database, must malloc memory for value	
Parameters	fw	handle
	user_interface_request_dat	

	a	
Return Value	None	
Example		

2.4.3. emv_error_indication_init

Prototype	EMV_VOID emv_error_indication_init(EMV_FRAME_WORK fw, EMV_ERROR_INDICATION_PTR error_indication);	
Description	get tag value to database, must malloc memory for value	
Parameters	fw	handle
	error_indication	Error indication(L1, L2, L3)
Return Value	None	
Example		

2.5. Data Structure definition

2.5.1. Payment data structure

```
typedef struct{
    EMV_KERNEL_ID                   m_kernel_id;
    EMV_OUTCOME_PARAM_T             m_outcome;
    EMV_BOOL                        m_error_indication_present;
    EMV_ERROR_INDICATION_T          m_error_indication;
    EMV_USER_INTERFACE_REQUEST_DATA_T  m_user_interface_request_data;
    EMV_USER_INTERFACE_REQUEST_DATA_T  m_user_interface_request_data_on_restart;
}EMV_PAYMENT_T, *EMV_PAYMENT_PTR;
```

2.5.2. Outcome Parameter data structure

```
typedef struct EMV_OUTCOME_PARAM_S{
    EMV_OUTCOME_STATUS                  m_status;
    EMV_OUTCOME_START                   m_start;
    EMV_OUTCOME_ONLINE_RESPONSE_DATA    m_online_response_data;
    EMV_OUTCOME_CVM                     m_cvm;
    EMV_BOOL                            m_ui_request_on_outcome_present;
```

```
    EMV_BOOL                                                m_ui_request_on_restart_present;
    EMV_BOOL                                                m_data_record_present;
    EMV_BOOL                                                m_discretionary_data_present;
    EMV_BOOL                                                m_receipt_requested;
    EMV_OUTCOME_ALTERNATE_INTERFACE_PREFERENCE m_alternalte_interface_preference;
    EMV_OUTCOME_FIELD_OFF_REQUEST                    m_field_off_request;
    EMV_BYTE                                                m_removal_timeout;
}EMV_OUTCOME_PARAM_T;
typedef EMV_OUTCOME_PARAM_T * EMV_OUTCOME_PARAM_PTR;
typedef const EMV_OUTCOME_PARAM_T * EMV_OUTCOME_PARAM_CPTR;
```

2.5.3. Error Indication data structure

```
typedef struct{
    EMV_L1_ERROR_T    m_L1;
    EMV_L2_ERROR_T    m_L2;
    EMV_L3_ERROR_T    m_L3;
    EMV_SW                m_sw12;
    EMV_MESSAGE_INDENTIFIER    m_msg_on_error;
}EMV_ERROR_INDICATION_T;
typedef EMV_ERROR_INDICATION_T * EMV_ERROR_INDICATION_PTR;
typedef const EMV_ERROR_INDICATION_T * EMV_ERROR_INDICATION_CPTR;
```

2.6. Error number definition

Error number	description
EMV_OK = 0	Successful OK or APPROVED
EMV_OK_DELAYED_AUTHORIZATION	Required Delayed authorization
EMV_NO_APP_SUPPORTED	Terminal don't have application supported by the card
EMV_APP_LOCKED	Application Locked
EMV_FORCE_APPROVED=5	Transaction force approved
EMV_DECLIEND	Transaction was declined
EMV_REQUIRE_ONLINE	Transaction required online
EMV_REQUIRE_ONLINE_LONG_TAP	Transaction required online with card keep in the contactless field
EMV_REQUIRE_ONLINE_2END_TAP	Transaction required online with second card tap
EMV_REQUIRE_ONLINE_2END_TAP_IF_HAS_SCRIPT	Transaction required online, second card tap required if there has script from the host
EMV_SELECT_NEXT	Select Next Application
EMV_TRY_AGAIN	Reader return error, try again

EMV_PRESENT_CARD_AGAIN	Start the transaction again and present card
EMV_OTHER_CARD	Current card not supported, Use other card
EMV_OTHER_INTERFACE	Try to insert card or swipe card
EMV_CANCELED	Transaction was canceled
EMV_SERVICE_NOT_ALLOWED	Transaction type not allowed
EMV_SERVICE_NOT_ACCEPTED	Transaction type not accepted
EMV_TERMINATED	Transaction terminated
EMV_INTERRUPT_BY_INSERTED	Transaction was canceled because of card insert
EMV_INTERRUPT_BY_SWIPED	Transaction was canceled because of card swiped
EMV_ISSUER_UPDATE_FAILED	Issuer script execute failed
EMV_CARD_FILE_NOT_FOUND	Card application file not found
EMV_CARD_BLOCKED	Card was locked
EMV_CARD_CMD_WARNING	Card APDU command success with warning
EMV_CARD_CMD_FAILED	Card command failed
EMV_CARD_PIN_WRONG	PIN not correct
EMV_CARD_PIN_BLOCKED	PIN locked
EMV_CARD_INVALID_DATA	Data from the card are invalid
EMV_CARD_DATA_MISSING=30	Data missing from the card
EMV_CARD_DATA_DUPLICATE	Data duplicated from the card
EMV_CARD_IN_EXCEPTION_FILE	Card in the exception file
EMV_ODA_CAPK_NOT_FOUND	CA public key not found
EMV_ODA_IPK_REVOCKED	Issuer public key certificate was revocked
EMV_ODA_SDA_FAILED	Offline data authentication - SDA failed
EMV_ODA_DDA_FAILED	Offline data authentication - DDA failed
EMV_ODA_CDA_FAILED	Offline data authentication - CDA failed
EMV_ODA_FDDA_FAILED	Offline data authentication - FDDA failed
EMV_ODA_KEY_RECOVER_FAILED	Offline data authentication - Key recover failed
EMV_APPLICATION_EXPIRED	Application Expired
EMV_GAC_REQUEST_ADVICE	Card required Advice(For Contact only)
EMV_TAG_DUPLICATE	Data duplicated int the database
EMV_TAG_UNDEFIND	Unknown Tag
EMV_TAG_IS_CONSTRUCTED	Tag is a template
EMV_DATA_LENGTH_INVALID	Data length invalid
EMV_DATA_NOT_PRESENT	Data not present
EMV_DATA_EMPTY	Data length is zero
EMV_DATA_INVALID	Data length or format invalid
EMV_DATA_PARSE_ERROR	Data format invalid, parsing failed
EMV_DATA_TEMPLETE_NO_MATCHED	Data under wrong template
EMV_TLV_INVALID	TLV data invalid
EMV_ICCARD_ERROR	Contact or contactless reader return error
EMV_ICCARD_APDU_IN_PROCESS	APDU in processing
EMV_HSM_ERROR	Encrypt or decrypt failed

EMV_NO_SUPPORTED	Function no supported
EMV_NO_IMPLEMENTED	Function no implemented
EMV_NO_MORE_DATA	Reched the end of the data
EMV_MEMORY_OVERFLOW	Malloc memory failed
EMV_MEMORY_NO_ENOUGH	Buffer size too small
EMV_TIMEOUT	Timeout
EMV_CALLBACK_ERROR	Callback failed
EMV_PARAM_ERROR	Parameter invalid
EMV_OTHER_ERROR	Other error
EMV_TRY_AGAIN_SEEPHONE	Try again see phone
EMV_QUICS_ERROR_MUTE_RETRY	QUICS, mute transaction
EMV_QUICS_ERROR_MUTE_RECOVERED	QUICS, mute transaction recovered
EMV_QUICS_ERROR_CARD_NOT_SAME	QUICS, card presented not same as the last one
EMV_QUICS_ERROR_EXPIRED_ONLINE	Card or application expired, go online
EMV_ODA_ONLINE_DDA_APPROVED	Online ODA approved, transaction approved
EMV_ODA_ONLINE_DDA_DECLINED	Online ODA declined, transaction declined
EMV_ODA_ONLINE_SDA_APPROVED	Online SDA approved, transaction approved
EMV_ODA_ONLINE_SDA_DECLINED	Online SDA declined, transaction declined
EMV_ODA_ONLINE_FAILED_DECLINED	Online ODA failed, transaction declined

2.7. Data Type definition

2.7.1. Base data type

```
typedef   unsigned long long      EMV_SIZE_T;
typedef   unsigned int            EMV_UINT;
typedef   unsigned char           EMV_BYTE;
typedef   int                     EMV_INT;
typedef   char                    EMV_CHAR;
typedef   void                    EMV_VOID;
typedef   unsigned short          EMV_SHORT;
typedef   unsigned long long          EMV_AMOUNT;
typedef   unsigned long long          EMV_TICKCOUNT;
typedef   unsigned long long          EMV_ULONG;
typedef   long long               EMV_LONG;

typedef   enum{
    EMV_FALSE = 0,
    EMV_TRUE
}EMV_BOOL;
```

```
typedef     EMV_BYTE *          EMV_BYTE_PTR;
typedef     const EMV_BYTE *    EMV_BYTE_CPTR;
typedef     const EMV_BYTE **   EMV_BYTE_CPPTR;
typedef     EMV_CHAR    *       EMV_CHAR_PTR;
typedef     const EMV_CHAR *    EMV_CHAR_CPTR;
typedef     const EMV_CHAR **   EMV_CHAR_CPPTR;
typedef     EMV_VOID *          EMV_VOID_PTR;
typedef     const EMV_VOID *        EMV_VOID_CPTR;
typedef     const EMV_VOID **   EMV_VOID_CPPTR;
typedef     EMV_VOID *          EMV_HANDLE;
typedef     EMV_BOOL *              EMV_BOOL_PTR;

typedef     struct EMV_PROGRAM_S        *           EMV_PROGRAM;
typedef     struct EMV_ENTRY_POINT_S *              EMV_ENTRY_POINT;
typedef     struct EMV_KERNEL_S         *           EMV_KERNEL;
typedef     struct EMV_FRAME_WORK_S *               EMV_FRAME_WORK;
typedef     struct EMV_OUTCOME_PARAM_S    *         EMV_OUTCOME_PARAM;
```

2.7.2. enum data type definition

2.7.2.1. EMV data format

```
typedef enum{
    EMV_FORMAT_A     = 0x00,
    EMV_FORMAT_AN    = 0x01,
    EMV_FORMAT_ANS = 0x02,
    EMV_FORMAT_B     = 0x03,
    EMV_FORMAT_CN    = 0x04,
    EMV_FORMAT_N     = 0x05
}EMV_DATA_FORMAT;
```

2.7.2.2. EMV Data Source

```
typedef enum{
    EMV_DATA_SOURCE_ICC                 =0x01,
    EMV_DATA_SOURCE_TERMINAL            =0x02,
    EMV_DATA_SOURCE_ISSUER              =0x04,
    EMV_DATA_SOURCE_ISSUER_TERMINAL     =0x06,
    EMV_DATA_SOURCE_KERNEL              =0x08,
    EMV_DATA_SOURCE_ALL                 =0x0F
```

```
}EMV_DATA_SOURCE;
```

2.7.2.3. EMV Data Source

```
typedef enum{
    EMV_KERNEL_TYPE_UNKNOW,
    EMV_CONTACT,
    EMV_CONTACTLESS
}EMV_KERNEL_TYPE;
typedef EMV_KERNEL_TYPE * EMV_KERNEL_TYPE_PTR;
```

2.7.2.4. EMV Kernel ID

```
typedef enum{
    KERNEL_EMV      =   0x00,
    KERNEL_PAYPASS =        0x02,
    KERNEL_PAYWAVE =    0x03,
    KERNEL_AMEX     =   0x04,
    KERNEL_JCB      =   0x05,
    KERNEL_DISCOVER=    0x06,
    KERNEL_QPBOC    =    0x07,
    KERNEL_RUPAY    =   0x0D,
    KERNEL_PURE     =   0x90,
    KERNEL_INTERAC =    0x91,
    KERNEL_UNKNOW   =   0xFF
}EMV_KERNEL_ID;
typedef  EMV_KERNEL_ID *            EMV_KERNEL_ID_PTR;
typedef  const EMV_KERNEL_ID *      EMV_KERNEL_ID_CPTR;
```

2.7.2.5. EMV Application Type

```
typedef enum
{
    EMV_APP_MSD,
    EMV_APP_EMV,
    EMV_APP_STANDARD
}EMV_APP_TYPE;
```

2.7.2.6. Outcome Status

```
typedef enum{
```

```
    EMV_OUTCOME_STATUS_APPROVED =                0x10,
    EMV_OUTCOME_STATUS_DECLIEND =                0x20,
    EMV_OUTCOME_STATUS_ONLINE_REQUEST =          0x30,
    EMV_OUTCOME_STATUS_END_APPLICATION =         0x40,
    EMV_OUTCOME_STATUS_SELECT_NEXT       =       0x50,
    EMV_OUTCOME_STATUS_TRY_ANOTHER_INTERFACE = 0x60,
    EMV_OUTCOME_STATUS_TRY_AGAIN         =       0x70,
    EMV_OUTCOME_STATUS_NA                =       0xF0
}EMV_OUTCOME_STATUS;
```

2.7.2.7. Outcome CVM

```
typedef enum{
    EMV_OUTCOME_CVM_NO_CVM =                 0x00,
    EMV_OUTCOME_CVM_OBTAIN_SIGNATURE =       0x10,
    EMV_OUTCOME_CVM_ONLINE_PIN           =   0x20,
    EMV_OUTCOME_CVM_CONFIRMTION_CODE_VERIFIED = 0x30,
    EMV_OUTCOME_CVM_NA                                = 0xF0
}EMV_OUTCOME_CVM;
```

2.7.2.8. L1 Error number

```
typedef enum{
    EMV_L1_OK = 0x00,
    EMV_L1_TIME_OUT_ERROR = 0x01,
    EMV_L1_TRANSMISSION_ERROR = 0x10,
    EMV_L1_TIME_PRTOCOL_ERROR = 0x11,
}EMV_L1_ERROR_T;
```

2.7.2.9. L2 Error number

```
typedef enum{
    EMV_L2_OK = 0x00,
    EMV_L2_CARD_DATA_MISSING = 0x01,
    EMV_L2_CAM_FAILED = 0x02,
    EMV_L2_STATUS_BYTES = 0x03,
    EMV_L2_PARSING_ERROR = 0x04,
    EMV_L2_MAX_LIMIT_EXCEEDED = 0x05,
    EMV_L2_CARD_DATA_ERROR = 0x06,
```

```
    EMV_L2_MAGSTRIPE_NOT_SUPPORTED = 0x07,
    EMV_L2_NO_PPSE = 0x08,
    EMV_L2_PPSE_FAULT = 0x09,
    EMV_L2_EMPTY_CANDIDATE_LIST = 0x0A,
    EMV_L2_IDS_READ_ERROR    = 0x0B,
    EMV_L2_IDS_WRITE_ERROR = 0x0C,
    EMV_L2_IDS_DATA_ERROR = 0x0D,
    EMV_L2_IDS_NO_MATCHING_AC = 0x0E,
    EMV_L2_TERMINAL_DATA_ERROR = 0x0F,
}EMV_L2_ERROR_T;
typedef EMV_L2_ERROR_T * EMV_L2_ERROR_PTR;
typedef const EMV_L2_ERROR_T * EMV_L2_ERROR_CPTR;
```

2.7.2.10. L3 Error number

```
typedef enum{
    EMV_L3_OK = 0x00,
    EMV_L3_TIME_OUT = 0x01,
    EMV_L3_STOP = 0x10,
    EMV_L3_AMOUNT_NOT_PRESENT = 0x11,
}EMV_L3_ERROR_T;
```

2.7.2.11. Online result

```
typedef enum{
    EMV_ONLINE_APPROVED,
    EMV_ONLINE_DECLIEND,
    //EMV_ONLINE_ISSUER_REFERRER,
    EMV_ONLINE_ERROR,
    EMV_ONLINE_NA = 0Xff,
}EMV_ONLINE_RESULT;
typedef EMV_ONLINE_RESULT * EMV_ONLINE_RESULT_PTR;
```

2.8. Micro definition

2.8.1. Transaction Type

symbol	value
EMV_TRANS_PURCHASE	0x00

EMV_TRANS_CASH	0x01
EMV_TRANS_PURCHASE_WITH_CASH_BACK	0x09
EMV_TRANS_MANUAL_CASH	0x12
EMV_TRANS_QUASI_CASH	0x11
EMV_TRANS_CASH_DISBURSEMENT	0x17
EMV_TRANS_REFUND	0x20
EMV_TRANS_CASE_DEPOSIT	0x21
EMV_TRANS_INQUIRE	0x31
EMV_TRANS_TRANSFER	0x60
EMV_TRANS_PAYMENT	0x50
EMV_TRANS_ADMIN	0x66
EMV_TRANS_RETRIEVE	0x78
EMV_TRANS_UPDATE	0x79
EMV_TRANS_AUTH_APPLICATION	0x90
EMV_TRANS_MONEY_ADD_LEGACY_RUPAY	0x33
EMV_TRANS_VOID_RUPAY	0x34
EMV_TRANS_MONEY_ADD_NO_LEGACY_RUPAY	0x28
EMV_TRANS_BALANCE_ENQUIRY_NO_LEGACY_RUPAY	0x31
EMV_TRANS_SERVICE_CREATION_NO_LEGACY_RUPAY	0x83

2.8.2. Kernel API Interface name

symbol	value
API_KERNEL_TYPE_GET	"emv_kernel_type_get"
API_KERNEL_NAME_GET	"emv_kernel_name_get"
API_KERENL_VERSION_GET	"emv_kernel_version_get"
API_KERNEL_ID_GET	"emv_kernel_id_get"
API_KERNEL_INIT	"emv_kernel_init"
API_KERNEL_FREE	"emv_kernel_free"
API_KERNEL_ACTIVE	"emv_kernel_active"
API_KERNEL_CONFIG_CHECKSUM_GET	"emv_kernel_config_checksum_get"
API_CONTACTLESS_FALLBACK_AID_SUPPORTED	"emv_cl_fallback_aid_supported"
API_CONTACTLESS_PRE_TRANSACTION	"emv_cl_pre_transaction"
API_CONTACTLESS_TRANSACTION	"emv_cl_transaction"
API_CONTACTLESS_TRANSACTION_COMPLETION	"emv_cl_transaction_completion"
API_CONTACTLESS_TORN_CLEAN	"emv_cl_torn_clean"
API_CONTACTLESS_TORN_ADD	"emv_cl_torn_add"
API_CONTACT_INITIATE_APPLICATION	"emv_ct_initiate_application"
API_CONTACT_READ_APPLICATION_DATA	"emv_ct_read_application_data"
API_CONTACT_DATA_AUTHENTICATION	"emv_ct_data_authentication"

API_CONTACT_PROCESS_RESTRICTION	"emv_ct_process_restriction"
API_CONTACT_CARDHOLDER_VERIFICATION	"emv_ct_cardholder_verification"
API_CONTACT_TERMINAL_RISK_MANAGEMENT	"emv_ct_terminal_risk_management"
API_CONTACT_TERMINAL_ACTION_ANALYZE	"emv_ct_terminal_action_analysis"
API_CONTACT_CARD_ACTION_ANALYSIS	"emv_ct_card_action_analysis"
API_CONTACT_COMPLETION	"emv_ct_completion"
API_CONTACT_READ_LOG	"emv_ct_read_log"

3. Entry Point

3.1. Entry Point API

3.1.1. ep_version_get

Prototype	EMV_CHAR_CPTR ep_version_get(EMV_VOID);	
Description	Get the version of Entrypoint Module.	
Parameters	None	
Return Value	Return Version String	
Example		

3.1.2. ep_version_time_get

Prototype	EMV_CHAR_CPTR ep_version_time_get(EMV_VOID);	
Description	Get compile time of the module	
Parameters	None	
Return Value	Return the compile time.	
Example		

3.1.3. emv_ep_init

Prototype	EMV_ENTRY_POINT emv_ep_init(EMV_FRAME_WORK fw);	
Description	Initialize the Entrypoint mode.	
Parameters	fw	Return value of the emv_fw_init().
Return	Return pointer that point to EMV_ENTRY_POINT_T structure	

Value	address
Example	

3.1.4. emv_ep_free

Prototype	EMV_VOID emv_ep_free(EMV_ENTRY_POINT ep);	
Description	Free instance handle of the Entrypoint.	
Parameters	ep	Handle of the Entrypoint.
Return Value		
Example		

3.1.5. emv_ep_kernel_count_get

Prototype	EMV_UINT emv_ep_kernel_count_get(EMV_ENTRY_POINT ep);	
Description	Get the kernel count number.	
Parameters	ep	Handle of the Entrypoint.
Return Value		
Example		

3.1.6. emv_ep_kernel_id_get

Prototype	EMV_KERNEL_ID emv_ep_kernel_id_get(EMV_ENTRY_POINT ep, EMV_UINT data_index);	
Description	Get kernel Id.	
Parameters	ep	Handle of the Entrypoint.
	Data_index	Index of the kernel.
Return Value		
Example		

3.1.7. emv_ep_kernel_type_get

Prototype	EMV_KERNEL_TYPE emv_ep_kernel_type_get(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id);	
Description	Get kernel Id.	
Parameters	ep	Handle of the Entrypoint.
	kernel_id	Id of the kernel.
Return Value	EMV_KERNEL_TYPE_UNKNOW: do not know the kernel EMV_CONTACT: contact kernel EMV_CONTACTLESS: contactless kernel	
Example		

3.1.8. emv_ep_kernel_name_get

Prototype	EMV_CHAR_CPTR emv_ep_kernel_name_get(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id);	
Description	Get kernel name	
Parameters	ep	Handle of the Entrypoint.
	kernel_id	Id of the kernel.
Return Value	Return the string kernel name according to the kernel id	
Example		

3.1.9. emv_ep_kernel_version_get

Prototype	EMV_CHAR_CPTR emv_ep_kernel_version_get(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id);	
Description	Get kernel version number	
Parameters	ep	Handle of the Entrypoint.
	kernel_id	Id of the kernel.
Return Value	Return the string version number according to the kernel id indicate	
Example		

3.1.10. emv_ep_kernel_version_get

Prototype	EMV_CHAR_CPTR emv_ep_kernel_checksum_get(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id);	
Description	Get kernel checksum	
Parameters	ep	Handle of the Entrypoint.
	kernel_id	Id of the kernel.
Return Value	Return the checksum of kernel library that the kernel id indicated	
Example		

3.1.11. emv_ep_kernel_load

Prototype	EMV_ERROR emv_ep_kernel_load(EMV_ENTRY_POINT ep, EMV_CHAR_CPTR kernel_checksum, FN_KERNEL_TYPE_GET func_kernel_type_get, FN_KERNEL_NAME_GET func_kernel_name_get, FN_KERNEL_VERSION_GET func_kernel_version_get, FN_KERNEL_ID_GET func_kernel_id_get, FN_KERNEL_FALLBACK_AID_SUPPORTED func_contactless_fallback_aid_supported, FN_KERNEL_INIT func_kernel_init, FN_KERNEL_FREE func_kernel_free, FN_KERNEL_CONFIG_CHECKSUM_GET func_kernel_config_checksum_get, FN_KERNEL_ACTIVE func_kernel_active, FN_CONTACTLESS_PRE_TRANSACTION func_contactless_pre_transaction, FN_CONTACTLESS_TRANSACTION func_contactless_transaction, FN_CONTACTLESS_TRANSACTION_COMPLETION func_contactless_transaction_completion, FN_CONTACTLESS_TORN_CLEAN func_contactless_torn_clean, FN_CONTACTLESS_TORN_ADD func_contactless_torn_add, FN_CONTACT_INITIATE_APPLICATION func_contact_initiate_application, FN_CONTACT_READ_APPLICATION_DATA

	func_contact_read_application_data, FN_CONTACT_DATA_AUTHENTICATION func_contact_data_authentication, FN_CONTACT_PROCESS_RESTRICTION func_contact_process_restriction, FN_CONTACT_CARDHOLDER_VERIFICATION func_contact_cardholder_verification, FN_CONTACT_TERMINAL_RISK_MANAGEMENT func_contact_terminal_risk_management, FN_CONTACT_TERMINAL_ACTION_ANALYZE func_contact_terminal_action_analysis, FN_CONTACT_CARD_ACTION_ANALYSIS func_contact_card_action_analysis, FN_CONTACT_COMPLETION func_contact_completion, FN_CONTACT_READ_LOG func_contact_read_log);	
Description	Load the kernel library to the entrypoint	
Parameters	ep	Handle of the entrypoint.
	kernel_checksum	
	func_kernel_type_get	
	func_kernel_name_get	
	func_kernel_version_get	
	func_kernel_id_get	
	func_contactless_fallback_aid_supported	
	func_kernel_init	
	func_kernel_free	
	func_kernel_config_checksum_get	
	func_kernel_active	
	func_contactl	

	ess_pre_transaction	
	func_contactless_transaction	
	func_contactless_transaction_completion	
	func_contactless_torn_clean	
	func_contactless_torn_add	
	func_contact_initiate_application	
	func_contact_read_application_data	
	func_contact_data_authentication	
	func_contact_process_restriction	
	func_contact_cardholder_verification	
	func_contact_terminal_risk_management	
	func_contact_terminal_action_analysis	
	func_contact_card_action_analysis	
	func_contact_completion	
	func_contact_read_log	
Return	EMV_OK: Success	

Value	Other: Failed
Example	

3.1.12. emv_ep_kernel_param_set

Prototype	EMV_ERROR emv_ep_kernel_param_set(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id, EMV_BYTE_CPTR param, EMV_UINT param_len);	
Description	Set kernel parameter	
Parameters	ep	Handle of the entrypoint.
	kernel_id	Id of the kernel.
	param	Kernel parameter must to be TVR Format
	param_len	Length of the parameter
Return Value	EMV_OK: Success Other: Failed	
Example		

3.1.13. emv_ep_kernel_param_get

Prototype	EMV_ERROR emv_ep_kernel_param_get(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id, EMV_BYTE_PTR param, EMV_UINT_PTR param_len, EMV_CHAR_PTR checksum, EMV_UINT checksum_len);	
Description	Set kernel param	
Parameters	ep	Handle of the framework.
	kernel_id	Id of the kernel.
	param	TVR Format
	param_len	Length of the parameter
	checksum	Kernel configuration checksum
	checksum_len	Size of the checksum buffer
Return Value	EMV_OK: Success Other: Failed	
Example		

3.1.14. emv_ep_pre_transaction

Prototype	EMV_ERROR emv_ep_pre_transaction(EMV_ENTRY_POINT ep, EMV_BOOL read_log, EMV_BYTE_CPTR trans_data, EMV_UINT trans_data_len);	
Description	Set kernel param	
Parameters	ep	Handle of the entrypoint.
	read_log	EMV_TRUE: prepare a transaction to read log from the card EMV_FALSE:normal transaction
	trans_data	Transaction data,TLV format
	trans_data_len	Length of the transaction data
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2. Contact API

3.2.1. emv_ep_contact_build_candidate_list

Prototype	EMV_ERROR emv_ep_contact_build_candidate_list(EMV_ENTRY_POINT ep, EMV_BOOL contactless, EMV_BOOL ignore_blocked);	
Description	Build candidate list	
Parameters	ep	Handle of the entrypoint.
	contactless	EMV_TRUE: contactless EMV_FALSE: contact
	ignore_blocked	EMV_TRUE:include the blocked EMV_FALSE:Don't include the blocked applications
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.2. emv_ep_contact_application_select

Prototype	EMV_ERROR emv_ep_contact_application_select(EMV_ENTRY_POINT ep)	
Description	Select application	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.3. emv_ep_contact_initiate_application

Prototype	EMV_ERROR emv_ep_contact_initiate_application(EMV_ENTRY_POINT ep);	
Description	Initialize application	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.4. emv_ep_contact_read_application_data

Prototype	EMV_ERROR emv_ep_contact_read_application_data(EMV_ENTRY_POINT ep);	
Description	Read application data	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.5. emv_ep_contact_data_authentcation

Prototype	EMV_ERROR emv_ep_contact_data_authentcation(EMV_ENTRY_POINT ep);	
Description	Offline data authentication	
Parameters	ep	Handle of the entrypoint.
Return	EMV_OK: Success	

Value	Other: Failed
Example	

3.2.6. emv_ep_contact_prcessing_restrictions

Prototype	EMV_ERROR emv_ep_contact_prcessing_restrictions(EMV_ENTRY_POINT ep);	
Description	Cardholder verification	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.7. emv_ep_contact_teminal_risk_management

Prototype	EMV_ERROR emv_ep_contact_teminal_risk_management(EMV_ENTRY_POINT ep);	
Description	Terminal risk management	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.8. emv_ep_contact_terminal_action_analysis

Prototype	EMV_ERROR emv_ep_contact_terminal_action_analysis(EMV_ENTRY_POINT ep);	
Description	Terminal action analysis	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.9. emv_ep_contact_completion

Prototype	EMV_ERROR emv_ep_contact_completion(EMV_ENTRY_POINT ep, EMV_ONLINE_RESULT online_result, EMV_BYTE_CPTR auth_resp_code, EMV_UINT auth_resp_code_len, EMV_BYTE_CPTR issuer_auth_data, EMV_UINT issuer_auth_data_len, EMV_BYTE_CPTR auth_code, EMV_UINT auth_code_len, EMV_BYTE_CPTR script, EMV_UINT script_len, EMV_BOOL_PTR advice_needed);	
Description	completion	
Parameters	ep	Handle of the entrypoint.
	online_result	Reference EMV_ONLINE_RESULT
	auth_resp_code	Authorization response code
	auth_resp_code_len	Length of the authorization resoinse code
	issuer_auth_data	Issuer authentication data
	issuer_auth_data_len	Length of the issuer authentication data
	auth_code	Authentication code
	auth_code_len	Length of authentication code
	script	Issuer script
	script_len	Length of the issuer script
	advice_needed	Indicate that the card required advice
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.10. emv_ep_contact_read_log_info

Prototype	EMV_ERROR emv_ep_contact_read_log_info(EMV_ENTRY_POINT ep, EMV_TAG tag_of_log_entry, EMV_TAG tag_of_log_format, EMV_BYTE_PTR sfi,

	EMV_BYTE_PTR max_num_of_record, EMV_BYTE_PTR log_format, EMV_UINT_PTR log_format_len);	
Description	Read card log information	
Parameters	ep	Handle of the entrypoint.
	tag_of_log_entry	Tag of the log entry
	tag_of_log_format	Tag of the log format
	sfi	Return SFI of the log
	max_num_of_record	Return max number of the log
	log_format	Return the format of the log
	log_format_len	Return the length of the format
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.11. emv_ep_contact_read_record_log

Prototype	EMV_ERROR emv_ep_contact_read_record_log(EMV_ENTRY_POINT ep, EMV_BYTE sfi, EMV_BYTE rec_no, EMV_BYTE_CPTR log_format, EMV_UINT log_format_len, EMV_BYTE_PTR record, EMV_UINT_PTR record_len);	
Description	Read record log	
Parameters	ep	Handle of the entrypoint.
	sfi	Sfi of the log,from the card by emv_ep_contact_read_log_info
	rec_no	1 to max number of_record, max number of the record get from the card by emv_ep_contact_read_log_info
	log_format	get from the card by emv_ep_contact_read_log_info
	log_format_len	get from the card by emv_ep_contact_read_log_info
	record	Buffer to receive the log record

	record_len	Size of the buffer
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.12. emv_ep_contact_read_log

Prototype	EMV_ERROR emv_ep_contact_read_log(EMV_ENTRY_POINT ep, EMV_LOG_ID log_id, EMV_LOG_DATA_PTR log);	
Description	Special log read	
Parameters	ep	Handle of the entrypoint.
	log_id	Defined by the kernel
	log	Defined by the kernel
Return Value	EMV_OK: Success Other: Failed	
Example		

3.2.13. emv_ep_contact_read_amount

Prototype	EMV_ERROR emv_ep_contact_read_amount(EMV_ENTRY_POINT ep, EMV_TAG tag, EMV_BOOL bcd, EMV_UINT value_len, EMV_AMOUNT_PTR amount);	
Description	Read number from the card	
Parameters	ep	Handle of the entrypoint.
	tag	Tag of the numeric data
	bcd	EMV_TRUE:BDA format EMV_FALSE:Binary format
	value_len	Length of the value in the card
	amount	Amount from the card
Return Value	EMV_OK: Success Other: Failed	
Example		

3.3. Contactless API

3.3.1. emv_ep_contactless_build_combination

Prototype	EMV_ERROR emv_ep_contact_build_candidate_list(EMV_ENTRY_POINT ep, EMV_BOOL contactless, EMV_BOOL ignore_blocked);	
Description	Build application candidate list	
Parameters	ep	Handle of the entrypoint.
	contactless	
	ignore_blocked	
Return Value	EMV_OK: Success Other: Failed	
Example		

3.3.2. emv_ep_contactless_transaction

Prototype	EMV_ERROR emv_ep_contactless_transaction(EMV_ENTRY_POINT ep, EMV_BOOL other_interface_supported, EMV_PAYMENT_PTR payment);	
Description	Build application candidate list	
Parameters	ep	Handle of the entrypoint.
	other_interface_supported	EMV_TRUE: support other interface EMV_TRUE: not support
	payment	
Return Value	EMV_OK: Success Other: Failed	
Example		

3.3.3. emv_ep_contactless_transaction_completion

Prototype	EMV_ERROR emv_ep_contactless_transaction_completion(EMV_ENTRY_POINT ep,

	EMV_ONLINE_RESULT online_result, EMV_BYTE_CPTR auth_resp_code, EMV_UINT auth_resp_code_len, EMV_BYTE_CPTR issuer_auth_data, EMV_UINT issuer_auth_data_len, EMV_BYTE_CPTR auth_code, EMV_UINT auth_code_len, EMV_BYTE_CPTR script, EMV_UINT script_len, EMV_PAYMENT_PTR payment);	
Description	Completion a contactless transaction,call this function if online requested.	
Parameters	ep	Handle of the entrypoint.
	online_result	EMV_TRUE: support other interface EMV_TRUE: not support
	auth_resp_code	Authorization response code
	auth_resp_code_len	Length of the authorization response code
	issuer_auth_data	Issuer authentication data
	issuer_auth_data_len	Length of the issuer authentication data
	auth_code	Authentication code
	auth_code_len	Length of authentication cde
	script	Issuer script
	script_len	Length of the issuer script
	payment	
Return Value	EMV_OK: Success Other: Failed	
Example		

3.3.4. emv_ep_contactless_torn_clean

Prototype	EMV_ERROR emv_ep_contactless_torn_clean(EMV_ENTRY_POINT ep);	
Description	Clean all torn records from the kernels.	
Parameters	ep	Handle of the entrypoint.
Return Value	EMV_OK: Success Other: Failed	

Example	

3.3.5. emv_ep_contactless_torn_add

Prototype	EMV_ERROR emv_ep_contactless_torn_add(EMV_ENTRY_POINT ep, EMV_KERNEL_ID kernel_id, EMV_BYTE_CPTR torn_record, EMV_UINT torn_record_len)	
Description	Load torn record to the kernel.	
Parameters	ep	Handle of the entrypoint.
	kernel_id	Id of the kernel.
	torn_record	Record data ,the format depends of the kernel
	torn_record_len	Length of the record data
Return Value	EMV_OK: Success Other: Failed	
Example		

4. Parameters

4.1. CA Public

Tag symbol	Tag value	length	Format	Description
TAG_PLATFORM_RID	0xDF808010	5	B	RID
TAG_PLATFORM_CAPK_MODULES	0xDF808011	Max 256	B	Modules of CA public key
TAG_PLATFORM_CAPK_EXPONENTS	0xDF808012	1-4	B	Exponents of CA public key
TAG_PLATFORM_TAG_PLATFORM_CAPK_KEY_TYPE	0xDF808013	1	B	Key type
TAG_PLATFORM_CAPK_CHECKSUM	0xDF808014	20	B	SHA1 of the public key

4.1.1. EMV & PBOC

RID	Index	Exponent	Modules
A000000003	0x57	0x010001	942B7F2BA5EA307312B63DF77C5243618ACC2002BD7EC B74D821FE7BDC78BF28F49F74190AD9B23B9713B140FF EC1FB429D93F56BDC7ADE4AC075D75532C1E590B21874 C7952F29B8C0F0C1CE3AEEDC8DA25343123E71DCF86C6 998E15F756E3
A000000003	0x53	0x03	BCD83721BE52CCCC4B6457321F22A7DC769F54EB80259 13BE804D9EABBFA19B3D7C5D3CA658D768CAF57067EE C83C7E6E9F81D0586703ED9DDDADD20675D63424980B 10EB364E81EB37DB40ED100344C928886FF4CCC37203E E6106D5B59D1AC102E2CD2D7AC17F4D96C398E5FD993 ECB4FFDF79B17547FF9FA2AA8EEFD6CBDA124CBB17A0F 8528146387135E226B005A474B9062FF264D2FF8EFA368 14AA2950065B1B04C0A1AE9B2F69D4A4AA979D6CE95F EE9485ED0A03AEE9BD953E81CFD1EF6E814DFD3C2CE37 AEFA38C1F9877371E91D6A5EB59FDEDF75D3325FA3CA6 6CDFBA0E57146CC789818FF06BE5FCC50ABD362AE4B80 996D
A000000003	0x96	0x03	B74586D19A207BE6627C5B0AAFBC44A2ECF5A2942D3A 26CE19C4FFAEEE920521868922E893E7838225A3947A26 14796FB2C0628CE8C11E3825A56D3B1BBAEF783A5C6A8 1F36F8625395126FA983C5216D3166D48ACDE8A431212 FF763A7F79D9EDB7FED76B485DE45BEB829A3D4730848 A366D3324C3027032FF8D16A1E44D8D
A000000003	0x51	0x03	DB5FA29D1FDA8C1634B04DCCFF148ABEE63C772035C79 851D3512107586E02A917F7C7E885E7C4A7D529710A14 5334CE67DC412CB1597B77AA2543B98D19CF2CB80C522 BDBEA0F1B113FA2C86216C8C610A2D58F29CF3355CEB1 BD3EF410D1EDD1F7AE0F16897979DE28C6EF293E0A192 82BD1D793F1331523FC71A228800468C01A3653D14C6B 4851A5C029478E757F
A000000003	0x50	0x010001	D11197590057B84196C2F4D11A8F3C05408F422A35D70 2F90106EA5B019BB28AE607AA9CDEBCD0D81A38D48C7 EBB0062D287369EC0C42124246AC30D80CD602AB7238 D51084DED4698162C59D25EAC1E66255B4DB2352526EF 0982C3B8AD3D1CCE85B01DB5788E75E09F44BE7361366 DEF9D1E1317B05E5D0FF5290F88A0DB47
A000000003	0x58	0x010001	99552C4A1ECD68A0260157FC4151B5992837445D3FC57 365CA5692C87BE358CDCDF2C92FB6837522842A48EB11 CDFFE2FD91770C7221E4AF6207C2DE4004C7DEE1B6276 DC62D52A87D2CD01FBF2DC4065DB52824D2A2167A06D

			19E6A0F781071CDB2DD314CB94441D8DC0E936317B77 BF06F5177F6C5ABA3A3BC6AA30209C97260B7A1AD3A1 92C9B8CD1D153570AFCC87C3CD681D13E997FE33B3963 A0A1C79772ACF991033E1B8397AD0341500E48A24770B C4CBE19D2CCF419504FDBF0389BC2F2FDCD4D44E61F
A000000003	0x54	0x010001	C6DDC0B7645F7F16286AB7E4116655F56DD0C94476604 0DC68664DD973BD3BFD4C525BCBB95272B6B3AD9BA88 60303AD08D9E8CC344A4070F4CFB9EEAF29C8A3460850 C264CDA39BBE3A7E7D08A69C31B5C8DD9F94DDBC9265 758C0E7399ADCF4362CAEE458D414C52B498274881B19 6DACCA7273F687F2A65FAEB809D4B2AC1D3D1EFB4F649 0322318BD296D153B307A3283AB4E5BE6EBD910359A85 65EB9C4360D24BAACA3DBFE393F3D6C830D603C6FC1E8 3409DFCD80D3A33BA243813BBB4CEAF9CBAB6B74B001 16F72AB278A88A011D70071E06CAB140646438D986D48 281624B85B3B2EBB9A6AB3BF2178FCC3011E7CAF24897 AE7D
A000009999	0xE1	0x03	99C5B70AA61B4F4C51B6F90B0E3BFB7A3EE0E7DB41BC4 66888B3EC8E9977C762407EF1D79E0AFB2823100A020C 3E8020593DB50E90DBEAC18B78D13F96BB2F57EEDDC30 F256592417CDF739CA6804A10A29D2806E774BFA751F2 2CF3B65B38F37F91B4DAF8AEC9B803F7610E06AC9E6B
A000009999	0xE2	0x03	BD232E348B118EB3F6446EF4DA6C3BAC9B2AE510C5AD1 07D38343255D21C4BDF4952A42E92C633B1CE4BFEC39A FB6DFE147ECBB91D681DAC15FB0E198E9A7E4636BDCA1 07BCDA3384FCB28B06AFEF90F099E7084511F3CC010D4 343503E1E5A67264B4367DAA9A3949499272E9B5022F
A000009999	0xE3	0x010001	BC01E12223E1A41E88BFFA801093C5F8CEC5CD05DBBDB B787CE87249E8808327C2D218991F97A1131E8A25B012 2ED11E709C533E8886A1259ADDFDCBB396604D24E505 A2D0B5DD0384FB0002A7A1EB39BC8A11339C7A9433A9 48337761BE73BC497B8E58736DA4636538AD282D3CD3 DB
A000009999	0xE4	0x03	CBF2E40F0836C9A5E390A37BE3B809BDF5D740CB1DA38 CFC05D5F8D6B7745B5E9A3FA6961E55FF20412108525E6 6B970F902F7FF4305DD832CD0763E3AA8B8173F847771 00B1047BD1D744509312A0932ED25FED52A959430768C CD902FD8C8AD9123E6ADDB3F34B92E7924D729CB6473 533AE2B2B55BF0E44964FDEA8440117
A000009999	0xE5	0x03	D4FDAE94DEDBECC6D20D38B01E91826DC69543383799 17B2BB8A6B36B5D3B0C5EDA60B337448BAFFEBCC3ABD BA869E8DADEC6C870110C42F5AAB90A18F4F867F72E33 86FFC7E67E7FF94EBA079E531B3CF329517E81C5DD9B3 DC65DB5F9043190BE0BE897E5FE48ADF5D3BFA0585E07

			6E554F26EC69814797F15669F4A255C13
A000009999	0xE6	0x010001	EBF9FAECC3E5C315709694664775D3FBDA5A504D89344DD920C55696E891D9AB622598A9D6AB8FBF35E4599CAB7EB22F956992F8AB2E6535DECB6B576FA0675F97C23DD4C374A66E6AF419C9D204D0B9F93C08D789D63805660FBB629DF1B488CFA1D7A13E9B729437EEAFE718EFA859348BA0D76812A99F31CD364F2A4FD42F
A000000004	0xFE	0x010001	E76317965175A08BEE510F58830E87B262C70D529803245FA8B88E0C753562DE7AEB5A9E3E6C1A98E94D8DB7C31407DAC5D071E06B80B09E146F22DB85F1D72D1EA18D22600032C6DD40E3714D5ADA7DE9D7D01E88391F893156D6F4BF13E9063559DA0786DE9BDE6B1C9B0BB968EDDE07145ABF877B931682CCB1FB800728724D04AF241E2827E0FA1F62591914FF25
A000000004	0xFC	0x010001	B3296C91F4795BD97112606903407B6EFF3AB39246E91095E51D17867DA4ADE59A48BE2FE9B52710283D3D32260E2C7D247214C57D46AA6465E47E0A4B3FFAAD8A7F6A190755BCCFE3F3FB3989A9F6B1C9E1845BCCCAD6F20B1DAC6033600234E81DAC4153212B0F760C23099192AA6C4C9083BEFFD9A79D2A27B08FECC8E5D437D6C68550A839B1294151DABA9D9CB2F160F60F749289F500C8C7F334BD20EBAC4AB109CF3C182F1B781C7C097A7903530746C449B99E39E4DB6493DD2A02E37C62AE8BC9A7470ECCCF8DC06A18C33CD24B30D56F25D2755CE82AA4DE4D2EAEC07750A03DB75EBD0D8EBC9F2A1D85A0D252EFF40329BE05
A000000004	0xFD	0x010001	C9485DBEB5E40415D1B397524F47685F306CFDC499D4E2E7D0CBAF222CFA8184BD111DAEEDC9CC6EC8540C3F7271EA9990119CC5C43180501D9F45252D6835053FAE35696AE8CD67A325647449CF5E594DA8F627209F7F03AE8D6DFC0DB3E79E28E415DF29A5B57D6814856CC30A96DA5B8890363E507FCB2E283DA1EBB5F18E8E24102B7D0192BB8E35A4F7CD05A435
A000000004	0xFB	0x010001	9B170603A489C7546C45DA57B8FFD1DB2061240F0E8C6D1F9ABDC6B265AA8911915C1A4EABD8D0ED4755D1B902BA06FE5A645B786CD241295517D44EF1A7C25D75AFE0EB28066E4D69FEE7ABAFDD5EEB230F14E402C9840825FA77EAD12B5F1C5494701DE1897F65FE6BF106D47545EBF70CE7C158068C61F0773534DB742AB83C28038C1494F15905D0AD17CF1BD38D
A000000004	0xFA	0x010001	A4203E0C7BEB27097B63C103C19FDCDA671AEA7F813065756F3B9B81810CBD4BC4DEC548FBF1F3CDAE51F847235CBF2C8BADD8ACA7C93BEA3D44E80ED6A7B70E29622619DB420ACCCE07E1DD4E6C354F359FBDC9C5B70813926

			F77D827E52B19DAF09BFAE5274438BB8F61D17753C9EC0A8EFA3B7E46F02692160D2653CDBCC71B7D48BD37968316EB444F6504B9421B7DD3035A2C117D8B1F76A8975440DA9563618102397B881CEF8ADA7689EDFACE32482A2DFFED656E7F951DB841DA78368C6293BFC1053A86A845BFA6578E4B69F100B42B558FDE1AECEC6D250741BC783AA8A68A4261E7BB9246B10587A498D68DD955CE8B2B2433
A000000004	0xFF	0x010001	F69DBB5E15983EAE3CCF31CF4E47098C2FC16F97A0C710F84777EFA99622D86502B138728AB12E3481A84D20E014AD2D634D2836F27F294924B895A87F91F81B8169D4DFDAD8D7CBD741804CD61B467C7A9ACFECEB71188CAA73A907547699D45C9C7D2098AC2966266417F665A46BDD012C097DBD33D1D11AFF6EC8A9C0AD814A65B48262CA011636079A328C1AAEB7
A000000025	0x60	0x010001	A8EE74EDEF3C0DCA5102FF9B5707975FF67B60D64B5E7322D48DE9D3BB6153F63512A091B606DD8FD5F6A14588324EF8827844C7FFC0BAB2334AE5207770078B69CDC3F2C666CF69E28E16E1816714C4DF313BEF539CC01DA9DD2D6F47DE4F247C500B561C099166AD4FC16DF12DFB684AC48D35CDD2C47A13A86A5A162306F64E33B092AB74EDA71A4091D96E3DAA47
A000000152	0xD0	0x010001	D05C2A09D09C9031366EC092BCAC67D4B1B4F88B10005E1FC45C1B483AE7EB86FF0E884A19C0595A6C34F06386D776A21D620FC9F9C498ADCA00E66D129BCDD4789837B96DCC7F09DA94CCAC5AC7CFC07F4600DF78E493DC1957DEBA3F4838A4B8BD4CEFE4E4C6119085E5BB21077341C568A21D65D049D666807C39C401CDFEE7F7F99B8F9CB34A8841EA62E83E8D63
A000000152	0xD0	0x010001	A71AF977C1079304D6DFF3F665AB6DB3FBDFA1B170287AC6D7BC0AFCB7A202A4C815E1FC2E34F75A052564EE2148A39CD6B0F39CFAEF95F0294A86C3198E349FF82EECE633D50E5860A15082B4B342A90928024057DD51A2401D781B67AE7598D5D1FF26A441970A19A3A58011CA19284279A85567D3119264806CAF761122A71FC0492AC8D8D42B036C394FC494E03B43600D7E02CB5267755ACE64437CFA7B475AD40DDC93B8C9BCAD63801FC492FD251640E41FD13F6E231F56F97283447AB44CBE11910DB3C75243784AA9BDF57539C31B51C9F35BF8BC24957628812554782764B792BBDCA6498777AE9120ED935BB3E8BEA3EAB13D9
A000000065	0x02	0x010001	BB7F51983FD8707FD6227C23DEF5D5377A5A737CEF3C5252E578EFE136DF87B50473F9341F1640C8D258034E14C16993FCE6C6B8C3CEEB65FC8FBCD8EB77B3B05AC7C4D0

			9E0FA1BA2EFE87D3184DB6718AE41A7CAD89B8DCE0FE80CEB523D5D647F9DB58A31D2E71AC677E67FA6E75820736C9893761EE4ACD11F31DBDC349EF

4.1.2. Paypass

RID	Index	Exponent	Modules
A000000004	0x00	0x03	9C6BE5ADB10B4BE3DCE2099B4B210672B89656EBA091204F613ECC623BEDC9C6D77B660E8BAEEA7F7CE30F1B153879A4E36459343D1FE47ACDBD41FCD710030C2BA1D9461597982C6E1BDD08554B726F5EFF7913CE59E79E357295C321E26D0B8BE270A9442345C753E2AA2ACFC9D30850602FE6CAC00C6DDF6B8D9D9B4879B2826B042A07F0E5AE526A3D3C4D22C72B9EAA52EED8893866F866387AC05A1399
A000000004	0x05	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83A356BD190B88AB8DF04284D02A4204A7B6CB7C5551977A9B36379CA3DE1A08E69F301C95CC1C20506959275F41723DD5D2925290579E5A95B0DF6323FC8E9273D6F849198C4996209166D9BFC973C361CC826E1
A000000004	0xEF	0x03	A191CB87473F29349B5D60A88B3EAEE0973AA6F1A082F358D849FDDFF9C091F899EDA9792CAF09EF28F5D22404B88A2293EEBBC1949C43BEA4D60CFD879A1539544E09E0F09F60F065B2BF2A13ECC705F3D468B9D33AE77AD9D3F19CA40F23DCF5EB7C04DC8F69EBA565B1EBCB4686CD274785530FF6F6E9EE43AA43FDB02CE00DAEC15C7B8FD6A9B394BABA419D3F6DC85E16569BE8E76989688EFEA2DF22FF7D35C043338DEAA982A02B866DE5328519EBBCD6F03CDD686673847F84DB651AB86C28CF1462562C577B853564A290C8556D818531268D25CC98A4CC6A0BDFFFDA2DCCA3A94C998559E307FDDF915006D9A987B07DDAEB3B
A000000004	0xF1	0x03	A0DCF4BDE19C3546B4B6F0414D174DDE294AABBB828C5A834D73AAE27C99B0B053A90278007239B6459FF0BBCD7B4B9C6C50AC02CE91368DA1BD21AAEADBC65347337D89B68F5C99A09D05BE02DD1F8C5BA20E2F13FB2A27C41D3F85CAD5CF6668E75851EC66EDBF98851FD4E42C44C1D59F5984703B27D5B9F21B8FA0D93279FBBF69E090642909C9EA27F898959541AA6757F5F624104F6E1D3A9532F2A6E51515AEAD1B43B3D7835088A2FAFA7BE7
A000000004	0xF3	0x03	98F0C770F23864C2E766DF02D1E833DFF4FFE92D696E16

			42F0A88C5694C6479D16DB1537BFE29E4FDC6E6E8AFD1 B0EB7EA0124723C333179BF19E93F10658B2F776E829E8 7DAEDA9C94A8B3382199A350C077977C97AFF08FD1131 0AC950A72C3CA5002EF513FCCC286E646E3C5387535D5 09514B3B326E1234F9CB48C36DDD44B416D23654034A6 6F403BA511C5EFA3
A000000004	0xF5	0x100001	A6E6FB72179506F860CCCA8C27F99CECD94C7D4F3191D 303BBEE37481C7AA15F233BA755E9E4376345A9A67E79 94BDC1C680BB3522D8C93EB0CCC91AD31AD450DA30D3 37662D19AC03E2B4EF5F6EC18282D491E19767D7B2454 2DFDEFF6F62185503532069BBB369E3BB9FB19AC6F1C3 0B97D249EEE764E0BAC97F25C873D973953E5153A4206 4BBFABFD06A4BB486860BF6637406C9FC36813A4A75F7 5C31CCA9F69F8DE59ADECEF6BDE7E07800FCBE035D317 6AF8473E23E9AA3DFEE221196D1148302677C720CFE25 44A03DB553E7F1B8427BA1CC72B0F29B12DFEF4C081D0 76D353E71880AADFF386352AF0AB7B28ED49E1E672D11 F9
A000000004	0xF6	0x03	A25A6BD783A5EF6B8FB6F83055C260F5F99EA16678F3B 9053E0F6498E82C3F5D1E8C38F13588017E2B12B3D8FF6 F50167F46442910729E9E4D1B3739E5067C0AC7A1F4487 E35F675BC16E233315165CB142BFDB25E301A632A54A3 371EBAB6572DEEBAF370F337F057EE73B4AE46D1A8BC4 DA853EC3CC12C8CBC2DA18322D68530C70B22BDAC351 DD36068AE321E11ABF264F4D3569BB71214545005558D E26083C735DB776368172FE8C2F5C85E8B5B890CC68291 1D2DE71FA626B8817FCCC08922B703869F3BAEAC1459D 77CD85376BC36182F4238314D6C4212FBDD7F23D3
A000000004	0xF7	0x100001	94EA62F6D58320E354C022ADDCF0559D8CF206CD92E86 9564905CE21D720F971B7AEA374830EBE1757115A85E0 88D41C6B77CF5EC821F30B1D890417BF2FA31E5908DED 5FA677F8C7B184AD09028FDDE96B6A6109850AA800175 EABCDBBB684A96C2EB6379DFEA08D32FE2331FE103233 AD58DCDB1E6E077CB9F24EAEC5C25AF
A000000004	0xF8	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F 9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83 A356BD190B88AB8DF04284D02A4204A7B6CB7C555197 7A9B36379CA3DE1A08E69F301C95CC1C20506959275F4 1723DD5D2925290579E5A95B0DF6323FC8E9273D6F849 198C4996209166D9BFC973C361CC826E1
A000000004	0xF9	0x03	A99A6D3E071889ED9E3A0C391C69B0B804FC160B2B4BD D570C92DD5A0F45F53E8621F7C96C40224266735E1EE1 B3C06238AE35046320FD8E81F8CEB3F8B4C97B940930A 3AC5E790086DAD41A6A4F5117BA1CE2438A51AC053EB

			002AED866D2C458FD73359021A12029A0C043045C11664FE0219EC63C10BF2155BB2784609A106421D45163799738C1C30909BB6C6FE52BBB76397B9740CE064A613FF8411185F08842A423EAD20EDFFBFF1CD6C3FE0C9821479199C26D8572CC8AFFF087A9C3 CHECKSUM=C642D39603EEEC8FD7596E85A2858EE28531463E
A000000004	0xFA	0x03	A90FCD55AA2D5D9963E35ED0F440177699832F49C6BAB15CDAE5794BE93F934D4462D5D12762E48C38BA83D8445DEAA74195A301A102B2F114EADA0D180EE5E7A5C73E0C4E11F67A43DDAB5D55683B1474CC0627F44B8D3088A492FFAADAD4F42422D0E7013536C3C49AD3D0FAE96459B0F6B1B6056538A3D6D44640F94467B108867DEC40FAAECD740C00E2B7A8852D
B012345678	0x00	0x03	9C6BE5ADB10B4BE3DCE2099B4B210672B89656EBA091204F613ECC623BEDC9C6D77B660E8BAEEA7F7CE30F1B153879A4E36459343D1FE47ACDBD41FCD710030C2BA1D9461597982C6E1BDD08554B726F5EFF7913CE59E79E357295C321E26D0B8BE270A9442345C753E2AA2ACFC9D30850602FE6CAC00C6DDF6B8D9D9B4879B2826B042A07F0E5AE526A3D3C4D22C72B9EAA52EED8893866F866387AC05A1399
B012345678	0x02	0x03	A99A6D3E071889ED9E3A0C391C69B0B804FC160B2B4BDD570C92DD5A0F45F53E8621F7C96C40224266735E1EE1B3C06238AE35046320FD8E81F8CEB3F8B4C97B940930A3AC5E790086DAD41A6A4F5117BA1CE2438A51AC053EB002AED866D2C458FD73359021A12029A0C043045C11664FE0219EC63C10BF2155BB2784609A106421D45163799738C1C30909BB6C6FE52BBB76397B9740CE064A613FF8411185F08842A423EAD20EDFFBFF1CD6C3FE0C9821479199C26D8572CC8AFFF087A9C3
B012345678	0x05	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83A356BD190B88AB8DF04284D02A4204A7B6CB7C5551977A9B36379CA3DE1A08E69F301C95CC1C20506959275F41723DD5D2925290579E5A95B0DF6323FC8E9273D6F849198C4996209166D9BFC973C361CC826E1
B012345678	0xF3	0x100001	94EA62F6D58320E354C022ADDCF0559D8CF206CD92E869564905CE21D720F971B7AEA374830EBE1757115A85E088D41C6B77CF5EC821F30B1D890417BF2FA31E5908DED5FA677F8C7B184AD09028FDDE96B6A6109850AA800175EABCDBBB684A96C2EB6379DFEA08D32FE2331FE103233AD58DCDB1E6E077CB9F24EAEC5C25AF
B012345678	0xF7	0x03	98F0C770F23864C2E766DF02D1E833DFF4FFE92D696E16

			42F0A88C5694C6479D16DB1537BFE29E4FDC6E6E8AFD1 B0EB7EA0124723C333179BF19E93F10658B2F776E829E8 7DAEDA9C94A8B3382199A350C077977C97AFF08FD1131 0AC950A72C3CA5002EF513FCCC286E646E3C5387535D5 09514B3B326E1234F9CB48C36DDD44B416D23654034A6 6F403BA511C5EFA3
B012345678	0xF8	0x03	A99A6D3E071889ED9E3A0C391C69B0B804FC160B2B4BD D570C92DD5A0F45F53E8621F7C96C40224266735E1EE1 B3C06238AE35046320FD8E81F8CEB3F8B4C97B940930A 3AC5E790086DAD41A6A4F5117BA1CE2438A51AC053EB 002AED866D2C458FD73359021A12029A0C043045C1166 4FE0219EC63C10BF2155BB2784609A106421D451637997 38C1C30909BB6C6FE52BBB76397B9740CE064A613FF84 11185F08842A423EAD20EDFFBFF1CD6C3FE0C98214791 99C26D8572CC8AFFF087A9C3

4.1.3. Paywave

RID	Index	Exponent	Modules
A000000003	0x50	0x010001	D11197590057B84196C2F4D11A8F3C05408F422A35D70 2F90106EA5B019BB28AE607AA9CDEBCD0D81A38D48C7 EBB0062D287369EC0C42124246AC30D80CD602AB7238 D51084DED4698162C59D25EAC1E66255B4DB2352526EF 0982C3B8AD3D1CCE85B01DB5788E75E09F44BE7361366 DEF9D1E1317B05E5D0FF5290F88A0DB47
A000000003	0x51	0x03	DB5FA29D1FDA8C1634B04DCCFF148ABEE63C772035C79 851D3512107586E02A917F7C7E885E7C4A7D529710A14 5334CE67DC412CB1597B77AA2543B98D19CF2CB80C522 BDBEA0F1B113FA2C86216C8C610A2D58F29CF3355CEB1 BD3EF410D1EDD1F7AE0F16897979DE28C6EF293E0A192 82BD1D793F1331523FC71A228800468C01A3653D14C6B 4851A5C029478E757F
A000000003	0x52	0x03	AFF740F8DBE763F333A1013A43722055C8E22F41779E21 9B0E1C409D60AFD45C8789C57EECD71EA4A269A675916 CC1C5E1A05A35BD745A79F94555CE29612AC933876966 5B87C3CA8E1AC4957F9F61FA7BFFE4E17631E937837CAB F43DD6183D6360A228A3EBC73A1D1CDC72BF09953C81 203AB7E492148E4CB774CDDFAAC3544D0DD4F8C8A0E9 C70B877EA79F2C22E4CE52C69F3EF376F61B0F43A540FE 96C63F586310C3B6E39C78C4D647CADB5933

A000000003	0x53	0x03	BCD83721BE52CCCC4B6457321F22A7DC769F54EB80259 13BE804D9EABBFA19B3D7C5D3CA658D768CAF57067EE C83C7E6E9F81D0586703ED9DDDADD20675D63424980B 10EB364E81EB37DB40ED100344C928886FF4CCC37203E E6106D5B59D1AC102E2CD2D7AC17F4D96C398E5FD993 ECB4FFDF79B17547FF9FA2AA8EEFD6CBDA124CBB17A0F 8528146387135E226B005A474B9062FF264D2FF8EFA368 14AA2950065B1B04C0A1AE9B2F69D4A4AA979D6CE95F EE9485ED0A03AEE9BD953E81CFD1EF6E814DFD3C2CE37 AEFA38C1F9877371E91D6A5EB59FDEDF75D3325FA3CA6 6CDFBA0E57146CC789818FF06BE5FCC50ABD362AE4B80 996D
A000000003	0x58	0x010001	99552C4A1ECD68A0260157FC4151B5992837445D3FC57 365CA5692C87BE358CDCDF2C92FB6837522842A48EB11 CDFFE2FD91770C7221E4AF6207C2DE4004C7DEE1B6276 DC62D52A87D2CD01FBF2DC4065DB52824D2A2167A06D 19E6A0F781071CDB2DD314CB94441D8DC0E936317B77 BF06F5177F6C5ABA3A3BC6AA30209C97260B7A1AD3A1 92C9B8CD1D153570AFCC87C3CD681D13E997FE33B3963 A0A1C79772ACF991033E1B8397AD0341500E48A24770B C4CBE19D2CCF419504FDBF0389BC2F2FDCD4D44E61F
A000000003	0x96	0x03	B74586D19A207BE6627C5B0AAFBC44A2ECF5A2942D3A 26CE19C4FFAEEE920521868922E893E7838225A3947A26 14796FB2C0628CE8C11E3825A56D3B1BBAEF783A5C6A8 1F36F8625395126FA983C5216D3166D48ACDE8A431212 FF763A7F79D9EDB7FED76B485DE45BEB829A3D4730848 A366D3324C3027032FF8D16A1E44D8D
A000009999	0xE1	0x03	99C5B70AA61B4F4C51B6F90B0E3BFB7A3EE0E7DB41BC4 66888B3EC8E9977C762407EF1D79E0AFB2823100A020C 3E8020593DB50E90DBEAC18B78D13F96BB2F57EEDDC30 F256592417CDF739CA6804A10A29D2806E774BFA751F2 2CF3B65B38F37F91B4DAF8AEC9B803F7610E06AC9E6B
A000009999	0xE2	0x03	BD232E348B118EB3F6446EF4DA6C3BAC9B2AE510C5AD1 07D38343255D21C4BDF4952A42E92C633B1CE4BFEC39A FB6DFE147ECBB91D681DAC15FB0E198E9A7E4636BDCA1 07BCDA3384FCB28B06AFEF90F099E7084511F3CC010D4 343503E1E5A67264B4367DAA9A3949499272E9B5022F
A000009999	0xE3	0x010001	BC01E12223E1A41E88BFFA801093C5F8CEC5CD05DBBDB B787CE87249E8808327C2D218991F97A1131E8A25B012 2ED11E709C533E8886A1259ADDFDCBB396604D24E505 A2D0B5DD0384FB0002A7A1EB39BC8A11339C7A9433A9 48337761BE73BC497B8E58736DA4636538AD282D3CD3 DB

4.1.4. JCB

4.1.5. QPBOC

4.1.6. RuPay

RID	Index	Exponent	Modules
A000000524	0xFE	0x03	98F0C770F23864C2E766DF02D1E833DFF4FFE92D696E1642F0A88C5694C6479D16DB1537BFE29E4FDC6E6E8AFD1B0EB7EA0124723C333179BF19E93F10658B2F776E829E87DAEDA9C94A8B3382199A350C077977C97AFF08FD11310AC950A72C3CA5002EF513FCCC286E646E3C5387535D509514B3B326E1234F9CB48C36DDD44B416D23654034A66F403BA511C5EFA3
A000000524	0xFA	0x03	9C6BE5ADB10B4BE3DCE2099B4B210672B89656EBA091204F613ECC623BEDC9C6D77B660E8BAEEA7F7CE30F1B153879A4E36459343D1FE47ACDBD41FCD710030C2BA1D9461597982C6E1BDD08554B726F5EFF7913CE59E79E357295C321E26D0B8BE270A9442345C753E2AA2ACFC9D30850602FE6CAC00C6DDF6B8D9D9B4879B2826B042A07F0E5AE526A3D3C4D22C72B9EAA52EED8893866F866387AC05A1399
A000000524	0xF9	0x03	A99A6D3E071889ED9E3A0C391C69B0B804FC160B2B4BDD570C92DD5A0F45F53E8621F7C96C40224266735E1EE1B3C06238AE35046320FD8E81F8CEB3F8B4C97B940930A3AC5E790086DAD41A6A4F5117BA1CE2438A51AC053EB002AED866D2C458FD73359021A12029A0C043045C11664FE0219EC63C10BF2155BB2784609A106421D45163799738C1C30909BB6C6FE52BBB76397B9740CE064A613FF8411185F08842A423EAD20EDFFBFF1CD6C3FE0C9821479199C26D8572CC8AFFF087A9C3
A000000524	0xF8	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83A356BD190B88AB8DF04284D02A4204A7B6CB7C5551977A9B36379CA3DE1A08E69F301C95CC1C20506959275F41723DD5D2925290579E5A95B0DF6323FC8E9273D6F849198C4996209166D9BFC973C361CC826E1

A000000524	0xF7	0x010001	924D9576F8FB29F7E086265004EFB5897123F4FC6264E7AA61A53A352D83EFEC14B895101E8F9A00DF895FC780F13CFB5E43471E56BD51B7A6DC48044FA9BEE87032ACBBFB256E9B2559EF6A922F760AEDA1720818A954D6B0DA61F0E101371649898B8E18DCDEAA4BC7867D600A21D6CD462ACDE99F95672D52FECE228DE493
A000000524	0xF6	0x03	A25A6BD783A5EF6B8FB6F83055C260F5F99EA16678F3B9053E0F6498E82C3F5D1E8C38F13588017E2B12B3D8FF6F50167F46442910729E9E4D1B3739E5067C0AC7A1F4487E35F675BC16E233315165CB142BFDB25E301A632A54A3371EBAB6572DEEBAF370F337F057EE73B4AE46D1A8BC4DA853EC3CC12C8CBC2DA18322D68530C70B22BDAC351DD36068AE321E11ABF264F4D3569BB71214545005558DE26083C735DB776368172FE8C2F5C85E8B5B890CC682911D2DE71FA626B8817FCCC08922B703869F3BAEAC1459D77CD85376BC36182F4238314D6C4212FBDD7F23D3
A000000524	0xF5	0x010001	9C40C83BA1B9CA48296D1F7284553ED0BEB2D8D746034EEAD841FE0DB5F031D8EF70FB7E1A3FD479864551ADB333F59EBB9DFE200D813CF777133D51A402C29C282364A9FF4BD3FAD979DE18725BEDBD21B7175A06817BD21EEE4164E84B91F636D79D2BC66E6FFC45A3C75DB507AA086E993B88364C3AF6CBC2D0A34FDC91BF82DB9D750E44358E99D07406B06D7549EDCCD6164FB84C29258B655C2AEB98886BC4AF12AD151ED695B77434C2F857E981B332A9CF5959540CFBC7D1A197256BE75C200D94EF0B16FC34C1ED33D72CA9AABE06EC9019F299B5A322923E5A396C3A59D819BF2627DF82A10F29A1431492D1CCDD9FADA64CB7
A000000524	0xF3	0x03	98F0C770F23864C2E766DF02D1E833DFF4FFE92D696E1642F0A88C5694C6479D16DB1537BFE29E4FDC6E6E8AFD1B0EB7EA0124723C333179BF19E93F10658B2F776E829E87DAEDA9C94A8B3382199A350C077977C97AFF08FD11310AC950A72C3CA5002EF513FCCC286E646E3C5387535D509514B3B326E1234F9CB48C36DDD44B416D23654034A66F403BA511C5EFA3
A000000524	0x00	0x03	98F0C770F23864C2E766DF02D1E833DFF4FFE92D696E1642F0A88C5694C6479D16DB1537BFE29E4FDC6E6E8AFD1B0EB7EA0124723C333179BF19E93F10658B2F776E829E87DAEDA9C94A8B3382199A350C077977C97AFF08FD11310AC950A72C3CA5002EF513FCCC286E646E3C5387535D509514B3B326E1234F9CB48C36DDD44B416D23654034A66F403BA511C5EFA3
A000000524	0x01	0x010001	A6E6FB72179506F860CCCA8C27F99CECD94C7D4F3191D303BBEE37481C7AA15F233BA755E9E4376345A9A67E79

			94BDC1C680BB3522D8C93EB0CCC91AD31AD450DA30D3 37662D19AC03E2B4EF5F6EC18282D491E19767D7B2454 2DFDEFF6F62185503532069BBB369E3BB9FB19AC6F1C3 0B97D249EEE764E0BAC97F25C873D973953E5153A4206 4BBFABFD06A4BB486860BF6637406C9FC36813A4A75F7 5C31CCA9F69F8DE59ADECEF6BDE7E07800FCBE035D317 6AF8473E23E9AA3DFEE221196D1148302677C720CFE25 44A03DB553E7F1B8427BA1CC72B0F29B12DFEF4C081D0 76D353E71880AADFF386352AF0AB7B28ED49E1E672D11 F9
A000000524	0x02	0x03	A25A6BD783A5EF6B8FB6F83055C260F5F99EA16678F3B 9053E0F6498E82C3F5D1E8C38F13588017E2B12B3D8FF6 F50167F46442910729E9E4D1B3739E5067C0AC7A1F4487 E35F675BC16E233315165CB142BFDB25E301A632A54A3 371EBAB6572DEEBAF370F337F057EE73B4AE46D1A8BC4 DA853EC3CC12C8CBC2DA18322D68530C70B22BDAC351 DD36068AE321E11ABF264F4D3569BB71214545005558D E26083C735DB776368172FE8C2F5C85E8B5B890CC68291 1D2DE71FA626B8817FCCC08922B703869F3BAEAC1459D 77CD85376BC36182F4238314D6C4212FBDD7F23D3
A000000524	0x03	0x010001	94EA62F6D58320E354C022ADDCF0559D8CF206CD92E86 9564905CE21D720F971B7AEA374830EBE1757115A85E0 88D41C6B77CF5EC821F30B1D890417BF2FA31E5908DED 5FA677F8C7B184AD09028FDDE96B6A6109850AA800175 EABCDBBB684A96C2EB6379DFEA08D32FE2331FE103233 AD58DCDB1E6E077CB9F24EAEC5C25AF
A000000524	0x04	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F 9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83 A356BD190B88AB8DF04284D02A4204A7B6CB7C555197 7A9B36379CA3DE1A08E69F301C95CC1C20506959275F4 1723DD5D2925290579E5A95B0DF6323FC8E9273D6F849 198C4996209166D9BFC973C361CC826E1
A000000524	0x05	0x03	A99A6D3E071889ED9E3A0C391C69B0B804FC160B2B4BD D570C92DD5A0F45F53E8621F7C96C40224266735E1EE1 B3C06238AE35046320FD8E81F8CEB3F8B4C97B940930A 3AC5E790086DAD41A6A4F5117BA1CE2438A51AC053EB 002AED866D2C458FD73359021A12029A0C043045C1166 4FE0219EC63C10BF2155BB2784609A106421D451637997 38C1C30909BB6C6FE52BBB76397B9740CE064A613FF84 11185F08842A423EAD20EDFFBFF1CD6C3FE0C98214791 99C26D8572CC8AFFF087A9C3
A000000524	0xF1	0x03	A4DC71056B6607EFD116625AB0506D11DEEB4BAED647 5AEF11702C90604BA5D7F2F632236474F0C79E3FBE160A 6ABAC126730BD6853ECA412F38CD16DD48129CD53D91

			F1BB9196F2465C3014FCE2CA702C41472ED0609BD2380 52FE9C07F38DE7268DF1A0083E4DE20814B5BBFA9ADC3 3916A049155951648821A05C20CCFD7E8BC141EF3E29A 3F306325B13017EDC38D62E03B57A371DFC578274DC78 C3FBD6C5E60A0AF2901CAF3B0DD6975EFB5421
A000000524	0x06	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F 9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83 A356BD190B88AB8DF04284D02A4204A7B6CB7C555197 7A9B36379CA3DE1A08E69F301C95CC1C20506959275F4 1723DD5D2925290579E5A95B0DF6323FC8E9273D6F849 198C4996209166D9BFC973C361CC826E1

4.1.7. PURE

4.1.8. Discover

RID	Index	Exponent	Modules
A000000152	0x01	0x03	A1F5E1C9BD8650BD43AB6EE56B891EF7459C0A24FA84F 9127D1A6C79D4930F6DB1852E2510F18B61CD354DB83 A356BD190B88AB8DF04284D02A4204A7B6CB7C555197 7A9B36379CA3DE1A08E69F301C95CC1C20506959275F4 1723DD5D2925290579E5A95B0DF6323FC8E9273D6F849 198C4996209166D9BFC973C361CC826E1
A000000152	0x02	0x03	9C6BE5ADB10B4BE3DCE2099B4B210672B89656EBA0912 04F613ECC623BEDC9C6D77B660E8BAEEA7F7CE30F1B15 3879A4E36459343D1FE47ACDBD41FCD710030C2BA1D9 461597982C6E1BDD08554B726F5EFF7913CE59E79E3572 95C321E26D0B8BE270A9442345C753E2AA2ACFC9D3085 0602FE6CAC00C6DDF6B8D9D9B4879B2826B042A07F0E5 AE526A3D3C4D22C72B9EAA52EED8893866F866387AC05 A1399
A000000152	0x03	0x010001	94EA62F6D58320E354C022ADDCF0559D8CF206CD92E86 9564905CE21D720F971B7AEA374830EBE1757115A85E0 88D41C6B77CF5EC821F30B1D890417BF2FA31E5908DED 5FA677F8C7B184AD09028FDDE96B6A6109850AA800175 EABCDBBB684A96C2EB6379DFEA08D32FE2331FE103233 AD58DCDB1E6E077CB9F24EAEC5C25AF

4.2. Issuer Certificate Revoke List

Tag symbol	Tag value	length	Format	Description
TAG_PLATFORM_RID	0xDF808010	5	B	RID
TAG_CERTIFICATION_AUTHORITY_PUBLIC_KEY_INDEX_TERMINAL	0x9F22	1	B	Key Index
TAG_PLATFORM_CERT_SERIAL_NUMBER	0xDF808062	3	B	Serial number of the issuer public key

4.3. Exception File List

The parameter format can be defined by user.

4.4. Transaction Parameters

Tag symbol	Tag value	length	Format	Description
TAG_AMOUNT_AUTHORISED_NUMERIC	0x9F02	6	BCD	Authorized Number
TAG_AMOUNT_OTHER_NUMERIC	0x9F03	6	BCD	Authorized Number Other
TAG_TRANSACTION_CURRENCY_CODE	0x5F2A	2	B	Transaction Currency Code
TAG_TRANSACTION_CURRENCY_EXPONENT	0x5F36	1	B	Transaction Currency Exponent
TAG_TRANSACTION_REFERENCE_CURRENCY_CODE	0x9F3C	2	B	Transaction reference Currency Code
TAG_TRANSACTION_REFERENCE_CURRENCY_EXPONENT	0x9F3D	1	B	Transaction reference Currency Exponent
TAG_PLATFORM_FORCE_ONLINE	0xDF808069	1	B	Force Online
TAG_PLATFORM_FORCE_APPROVE	0xDF808068	1	B	Force Approve
TAG_PLATFORM_BATCH_DATA_CAPTURE	0xDF80806A	1	B	Batch Data capture indicator

4.5. Kernel Parameters

4.5.1. EMV&PBOC

Tag symbol	Tag value	length	Format	Description
TAG_PBOC_EC_LOG_ENTRY	0xDF4D	2	B	PBOC EC Log Entry
TAG_PBOC_EC_LOG_FORMAT	0xDF4F	Var.1-255	B	PBOC EC Log Format
TAG_PBOC_EC_BALANCE_EX	0x9F5D	6	N	PBOC EC Balance Ex
TAG_PBOC_EC_SINGLE_TRANS_LIMIT	0x9F78	6	N	PBOC EC Single Trans Limit
TAG_PBOC_EC_BALANCE	0x9F79	6	N	PBOC EC Balance
TAG_PBOC_EC_RESET_THRESHOLD	0x9F6D	6	N	PBOC EC Reset Threshold
TAG_PLATFORM_KERNEL_CONFIG	0xDF808061	Var.6-255	B	Kernel Configurations
TAG_PLATFORM_TAC_DENIAL	0xDF808020	5	B	Terminal Action Code Denial
TAG_PLATFORM_TAC_DEFAULT	0xDF808021	5	B	Terminal Action Code Default
TAG_PLATFORM_TAC_ONLINE	0xDF808022	5	B	Terminal Action Code Online
TAG_PLATFORM_TRM_TARGET_PERCENT	0xDF808023	1	B	Target Percent
TAG_PLATFORM_TRM_MAX_TARGET_PERCENT	0xDF808024	1	B	Max Target Percent
TAG_PLATFORM_TRM_THRESHOLD_VALUE	0xDF808025	6	N	Threshold Value
TAG_PLATFORM_DEFAULT_DDOL	0xDF808026	Var.0-256	B	Default DDOL
TAG_PLATFORM_DEFAULT_TDOL	0xDF808027	Var.0-256	B	Default TDOL
TAG_PLATFORM_SCRIPT_RESULT	0xDF808064	Var.1-256	B	Script Result

TAG_PBOC_CREDENTIALS_NUMBER	0x9F61	Var.1-256	ANS	Credentials number
TAG_PBOC_CREDENTIALS_TYPE	0x9F62	1	B	Credentials Type

4.5.2. QPBOC

4.5.3. Paypass

4.5.4. Paywave

4.5.4.1. Dynamic Reader Floor Limit Parameter

4.5.4.2. Other Parameter

4.5.5. JCB

4.5.6. American Express

4.5.7. Discover

4.5.8. RuPay

4.5.8.1. Service Parameter

Tag symbol	Tag value	length	Format	Description
TAG_RUPAY_SERVICE_AVAILABILITY_INFO	0xDF03	1	B	Service Availability Info
TAG_RUPAY_SERVICE_DIRECTORY	0xDF07	Var.0-256	B	Service Directory
TAG_RUPAY_SERVICE_MANAGEMENT_INFO	0xDF15	2	B	Service Management Information
TAG_RUPAY_SERVIC	0xDF16	2	B	Service ID

E_ID				
TAG_RUPAY_SERVICE_ATC	0xDF20	2	B	Service ATC
TAG_RUPAY_SERVICE_SUMMARY	0xDF22	8	B	Service Summary
TAG_RUPAY_SERVICE_SIGNATURE	0xDF23	8	B	Service Signature
TAG_RUPAY_SERVICE_CURRENCY_CODE	0xDF30	2	B	Service Currency Code
TAG_RUPAY_SERVICE_FOLDER	0xDF32	2	B	Service Folder
TAG_RUPAY_SERVICE_RELATED_DATA	0xDF33	Var.0-128	B	Service Related Data
TAG_RUPAY_SERVICE_DATA_FORMAT	0xDF44	Var.0-256	B	Service Data Format
TAG_RUPAY_SERVICE_TERMINAL_DATA	0xDF45	Var.0-96	B	Service Terminal Data
TAG_RUPAY_SERVICE_CONTROL	0xDF52	2	B	Service Control

4.5.8.2. Other Parameter

Tag symbol	Tag value	length	Format	Description
TAG_RUPAY_CARD_CVM_LIMIT	0xDF34	6	N	Card CVM Limit
TAG_RUPAY_ADDITIONAL_TERMINMAL_CAPABILITIES_EXTENSION	0xDF3A	5	B	Additional Terminal Cap. Extension
TAG_RUPAY_APPLICATION_USAGE_CONTROL_EXCTNSION	0xDF3B	2	N	Application Usage Control Extension
TAG_RUPAY_PRMiss	0xDF47	16	B	PRMiss
TAG_RUPAY_PRMacq	0xDF48	16	B	PRMacq
TAG_RUPAY_PRMicc	0xDF49	8	B	PRMicc
TAG_RUPAY_PREVIOUS_PRMICC	0xDF4B	8	B	Previous PRMicc
TAG_RUPAY_CONTACTLESS_TRANSACTION_LIMIT	0xDF4C	6	N	Contactless Transaction Limit

TAG_RUPAY_TERMINAL_CVM_LIMIT	0xDF4D	6	N	Terminal CVM Limit
TAG_RUPAY_PRMacq_INDEX	0xDF4E	1	B	PRMacq Index
TAG_RUPAY_PRMacq_KCV	0xDF54	3	B	PRMacq KCV
TAG_RUPAY_ICC_DYNAMIC_SIGNATURE_RECORD_IDENTIFIER	0xDF61	2	B	ICC Dynamic Signature Record Id
TAG_RUPAY_PRMacq_LEGACY	0xDF808900			
TAG_RUPAY_PRMacq_KCV_LEGACY	0xDF808901			
TAG_PLATFORM_KERNEL_CONFIG	0xDF808061	2	B	Kernel Configuration Bits description: Bits8:skip TAC/IAC default Bits7:online data capture Bits6:batch data capture Bits5:force approve Bits4:force online Other bit: RFU
TAG_PLATFORM_TAC_DENIAL	0xDF808020	5	B	Terminal Action Code Denial
TAG_PLATFORM_TAC_DEFAULT	0xDF808021	5	B	Terminal Action Code Default
TAG_PLATFORM_TAC_ONLINE	0xDF808022	5	B	Terminal Action Code Online
TAG_PLATFORM_TRM_TARGET_PERCENT	0xDF808023	1	B	Target Percent
TAG_PLATFORM_TRM_MAX_TARGET_PERCENT	0xDF808024	1	B	Max Target Percent
TAG_PLATFORM_TRM_THRESHOLD_VALUE	0xDF808025	6	N	Threshold Value
TAG_PLATFORM_FORCE_ONLINE	0xDF808069	1	B	Force Online
TAG_PLATFORM_FORCE_APPROVE	0xDF808068	1	B	Force Approve
TAG_PLATFORM_SCRIPT_RESULT	0xDF808064	Var.1-256	B	Script Result

4.5.9. Pure

Tag symbol	Tag value	length	Format	Description
TAG_PURE_TERMINAL_TRANSACTION_PROCESSING_INFORMATION_ENTRY	0xC7	5	B	Terminal Transaction Processing information
TAG_PURE_CONTACTLESS_CRYPTOGRAM_INFORMATION_DATA	0xC5	1	B	Contactless Cryptogram Information Data
TAG_PURE_CRM_CURRENCY_CODE	0xCD	2	N	CRM Currency Code
TAG_PURE_BALANCE	0x9F50	6	B	COTA offline Balance
TAG_PURE_MEMORY_SLOT_UPDATE_TEMPLATE	0xBF70	Var.1-800	B	Memory Slot Update Template
TAG_PURE_MEMORY_SLOT_READ_TEMPLATE	0xBF71	Var.1-800	B	Memory Slot Read Template
TAG_PURE_GDDOL	0x9F70	Var 1- 50	B	GDDOL
TAG_PURE_GDDOL_RESULTING_BUFFER	0x9F71	Var1-512	B	GDDOL Resulting Buffer
TAG_PURE_ISSUER_SCRIPT_RESULT	0x9F73	Var1-10	B	Issuer Script Results
TAG_PURE_DATA_ELEMENT_UPDATE_RESULT	0x9F74	2	B	Data Elements Update Result
TAG_PURE_ECHO_CARD_IDENTIFIER	0x9F75	Var1-20	B	Echo Card Identifier
TAG_PURE_TERMINAL_TRANSACTION_DATA	0x9F76	Var1-250	B	Terminal Transaction Data
TAG_PURE_TERMINAL_DEDICATED_DATA	0x9F77	Var1-250	B	Terminal Dedicated Data
TAG_PURE_SUPPORTED_APPLICATION	0xDF7F	Var1-16	B	Terminal AID value supported
TAG_PURE_PRE_PROCESSING_INDICAT	0xDF808700	1	B	Pre Processing Indicator

OR				
TAG_PURE_CONTACTLESS_APPLICATION_CAPABILITIES	0xDF8087 01	5	B	Contactless Applicationi Capabilities
TAG_PURE_ATOL	0xDF8087 02	Var1-50	B	Additional Tag Object List
TAG_PURE_MTOL	0xDF8087 03	Var1-50	B	Mandatory Tag Object List
TAG_PURE_ATDOL	0xDF8087 04	Var1-50	B	Authentication Transaction Data Object List
TAG_PURE_POS_TIMEOUT_LONG_TAP	0xDF8087 05	1	B	POS Timeout(Long Tap)
TAG_PURE_POS_TIMEOUT	0xDF8087 06	1	B	POS Timeout(Display)
TAG_PURE_TRANS_TYPE_OF_AAT	0xDF8087 07	1	B	POS Timeout(AAT)

4.5.10. Interact