OneMoney Digital Payment System (DPS) API Integration Documentation Guide - V1.1

28 November 2024

Revision History

Date	Version	illustrate	Author/Upatedby	Audit
2024-11-0 7	V1.0	First draft writing	Zhang Jie	Hu Wenfeng
2024-11-28	V1.1	Update Final Copy	T i n a y e i s h e	Brighton Tsiga

Table of Contents

1. Interface Documentation

1.1. Objects of use

Developers, maintainers and managers who connect to the MFS platform should have the following basic knowledge:

- Understand HTML language and CGI (Common Gateway Interface) or ASP

(Active Server Pages)

- ➢ Development languages such as java, php, c#, go, python, etc.
- ➢ Understand mysql, oracle, sqlserver, db2 and other databases;
- ➢ Understand the basic concepts of information security.

2. Interface Calling Method

Except for the image upload interface which uses form request, other interfaces all use JSON data request.

2.1. Security Keys (Key Pair) Generation

The OneMoney DPS API integration process uses an a*symmetric encryption* and requires third-party systems to generate private and public keys. Asymmetric encryption is the process of using a public key from a public/private key pair to encrypt plaintext, and then using the corresponding private key to decrypt the ciphertext.

The following describes the flow for using an asymmetric key to encrypt and decrypt data. The two parties involved in the workflow consists of a third-party system and OneMoney DPS platform.

- The third-party generates the private and public keys to be used for the integration process.
- The third-party then shares the public key to OneMoney DPS platform and keeps the private key.
- The OneMoney DPS platform creates ciphertext using the third-party system's public key, and then the third-party system decrypts the ciphertext using the third-party system 's private key. Only someone with knowledge of the private key can decrypt the ciphertext.

2.1.1. Asymmetric Encryption Algorithm

The OneMoney DPS API integration uses **RSA algorithm** *with key length set to 2048 bits*.

Below is an example how Asymmetric keys are generated

RSA Key Generator

You may generate an RSA private key with the help of this tool. Additionally, it will display the public key of a generated or pasted private key.

Key Length

2048 Generate key pair

Private key

```
-----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQBv9e+y5+B/AaZ6biuc8ri4oAsw9AlgscWtRTAqEv22NJ49EgHM
A9jaBz5QpvOJ+A959vlx/CdLT8tpWQHbCc8SlMEYts47+doUTa8ATRmdJPviF+ic
VUgKSU3PX7kbR40BzTiT8wMvWwxK+ePAIwLZxgzC6DUpJGyb8Vw+783ZScLYa4gc
NyCs7IvnDK3QbVkLXWMAcboRQroYgqX5Bc/nL1uEaSCKSBdIZjW/Aps5cnXoL7gK
UJ1Ey65gqqbldAaXpTFdW0QFkOZ1vj5v/qSUJhcMyDvrmQmKxjeXBNkaNY3ebG+Z
M2cBUhCRsEH/426FTcv59LkxsMqaGJOtof5rAgMBAAECggEAEKOweBpWFwobv95X
xP+dzHKcty3Npv1Er7e0rMwdwNKZIeJnOfboZ57j+ACX+U1Qn4PBFtuuy1aUmJeN
2wveJpjNCetBQ1CixJm3BIhw4hX5CWUcQP+7sSur0MIdNJyxrC7f1b0sIAGcFhiF
bKzHY643f1Abdn3fVl3Mo93n7f8Bse9mUQ7s1jILrVDfiKQiS7+eAT3yJXvrm2vn
h7kb9OfFg3wA55dZGcwAsIidLGn6dkEqvnLTZlspitOXjDXO75J+Znj2BxPnX5tg
oZT+f7c17MqKpIRl6t5kueB8qhkNgvJT0hjxC+0ZU3TqJFpxyxl+Lo4Z9Ya/RDm7
CuaFwQKBgQC+/pFXzFDgU7VLE/YXIH+DMuVwgkMk2/ZjAtTaxuL4k4w93DI4aPaC
k1BNZJTl1fAhbeBWf2hEERuAvhaIGcZFs4s4e0CE9ZRd5buOHoUo8DF2ZrncYbnZ
igfGl3StZmQNLsE4inIazatYf4kxMYVp3aBhTN7DhiHZ8F0ABPGmKQKBgQCWER+s
4huBFM2mZniM+aLnMhjcSYLqfIY9cwzcqGT6flQY1ytU9DWzy/nFijriW9tNbX5Z
dRqNvNHtxEvvs3Ze3s6lEQLDRErLH/iW9LRAHcKlYS50K2HN7sIGzm8XjdHu+KK7
TeuVdPzlQ4GmD/SDTNxgr+q8fMFiIQhy1zjKcwKBgGO678/lqzGH5Dmg4bZvzMWI
VRArpL3hccV/m//lLG1aNjey/DJCdL34CE1xQOLGlI4QHr1r59jrCe5u07sbolGR
EBhdG3KR/XgOVqI+Q4/95AYMW+68xNvX3HmRamaE+ad8i35npG9kh509czjRv/Vo
```

Public key

```
-----BEGIN PUBLIC KEY-----
MIIBITANBgkqhkiG9w0BAQEFAAOCAQ4AMIIBCQKCAQBv9e+y5+B/AaZ6biuc8ri4
oAsw9AlgscWtRTAqEv22NJ49EgHMA9jaBz5QpvOJ+A959vlx/CdLT8tpWQHbCc8S
lMEYts47+doUTa8ATRmdJPviF+icVUgKSU3PX7kbR40BzTiT8wMvWwxK+ePAIwLZ
xgzC6DUpJGyb8Vw+783ZScLYa4gcNyCs7IvnDK3QbVkLXWMAcboRQroYgqX5Bc/n
L1uEaSCKSBdIZjW/Aps5cnXoL7gKUJ1Ey65gqqbldAaXpTFdW0QFkOZ1vj5v/qSU
JhcMyDvrmQmKxjeXBNkaNY3ebG+ZM2cBUhCRsEH/426FTcv59LkxsMqaGJOtof5r
AgMBAAE=
-----END PUBLIC KEY-----
```

```java
public class CryptoUtil {
    private static final byte[] IV = new byte[]{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15, 16};

    public CryptoUtil() {
    }

    public static String rsaEncrypt(String content, PrivateKey key) {
        RSA rsa = new RSA(AsymmetricAlgorithm.RSA_ECB_PKCS1.getValue(), key,
(PublicKey)null);
        return encrypt(content, rsa, KeyType.PrivateKey);
    }

    public static String rsaEncrypt(String content, PublicKey key) {
        RSA rsa = new RSA(AsymmetricAlgorithm.RSA_ECB_PKCS1.getValue(),
(PrivateKey)null, key);
        return encrypt(content, rsa, KeyType.PublicKey);
    }

    protected static String encrypt(String content, RSA rsa, KeyType keyType) {
        byte[] bytes = content.getBytes(StandardCharsets.UTF_8);
        bytes = rsa.encrypt(bytes, keyType);
        Encoder encoder = Base64.getEncoder();
        content = encoder.encodeToString(bytes);
        return encoder.encodeToString(content.getBytes(StandardCharsets.UTF_8));
    }
}
```

```java
public class KeyPairTest {
    public static void main(String[] args) throws NoSuchAlgorithmException {
        // Create an RSA key pair generator
        KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("RSA");
        // Set the key length, usually 2048 bits
        keyPairGenerator.initialize(2048);

        // Generate RSA key pair
        KeyPair keyPair = keyPairGenerator.generateKeyPair();
        PublicKey publicKey = keyPair.getPublic();
        PrivateKey privateKey = keyPair.getPrivate();

        System.out.println("Public key: " +
Base64.getEncoder().encodeToString(publicKey.getEncoded()));
        System.out.println("Private key: " +
Base64.getEncoder().encodeToString(privateKey.getEncoded()));
    }
}
```

2.4. OneMoney DPS Platform Public key

This is the OneMoney DPS platform public key

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzT00cO3c0GKpFSRA2JTfYKiPfwthrG
3Q1PRaOEm1rdBkGWEL3120Ukh/OBRPpzSJHgffyivWtdxUIREEFehdARG3Ru/
nhehmPbzODLInVUXib6VTmyc+o9NssQwzuqyXtHCpFOAcZUyIliI12MREz3pWRFdU9vutPE7egBdi
InzRdm5hC1z809Q/OA4HkosQqpvHF24TmjfvjI97gUY/
zwrX0dY5PRsIlJjuV1K5zhXu3TDYbbC8Nyclmbsk1AYGS9kQKtJsYWaN4zIM8svz5IGT8Mg/
FTARGKyhSXDR0lJ3ZvLYdvrVNu1XD5/OR6m+9Z1BbWeYPwXK5tGe9LEH2nQIDAQAB

2.5. Interface Payload Encryption and Decryption

Interface payload (requests and responses) encrypted and decrypted using a symmetric algorithm known as **AES**.

2.5.1.AES Encryption and Decryption Method

Cipher Mode of Encryption	ECB
Padding	PKCS7Padding
Key Size in Bits	128 bits
Output Text Format	Base64

```
public static String aesEncrypt(String content, String key) {
        AES aes = new AES(Mode.ECB.name(), "PKCS7Padding",
key.getBytes(StandardCharsets.UTF_8));
        byte[] bytes = content.getBytes(StandardCharsets.UTF_8);
        Encoder encoder = Base64.getEncoder();
        content = encoder.encodeToString(aes.encrypt(bytes));
        return encoder.encodeToString(content.getBytes(StandardCharsets.UTF_8));
    }

    public static String aesDecrypt(String secret, String key) {
        AES aes = new AES(Mode.ECB.name(), "PKCS7Padding",
key.getBytes(StandardCharsets.UTF_8));
        Decoder decoder = Base64.getDecoder();
        secret = new String(decoder.decode(secret), StandardCharsets.UTF_8);
        secret = secret.replaceAll("\r\n", "");
        byte[] bytes = decoder.decode(secret);
        return new String(aes.decrypt(bytes), StandardCharsets.UTF_8);
    }
```

```java
public static String aesEncrypt(String content, String key) {
        AES aes = new AES(Mode.ECB.name(), "PKCS7Padding",
key.getBytes(StandardCharsets.UTF_8));
        byte[] bytes = content.getBytes(StandardCharsets.UTF_8);
        Encoder encoder = Base64.getEncoder();
        content = encoder.encodeToString(aes.encrypt(bytes));
        return encoder.encodeToString(content.getBytes(StandardCharsets.UTF_8));
    }

    public static String aesDecrypt(String secret, String key) {
        AES aes = new AES(Mode.ECB.name(), "PKCS7Padding",
key.getBytes(StandardCharsets.UTF_8));
        Decoder decoder = Base64.getDecoder();
        secret = new String(decoder.decode(secret), StandardCharsets.UTF_8);
        secret = secret.replaceAll("\r\n", "");
        byte[] bytes = decoder.decode(secret);
        return new String(aes.decrypt(bytes), StandardCharsets.UTF_8);
    }
```

Sample Request Payload


```
{
        "timestamp": 1730899573644,
        "random": "039cfed6-4cce-48b5-9erb-c4c8f44c0988",
        "encryptKeyId": "123456",
        "merNo": "1781526900800749576",
        "encryptData":"c2xnT29xM1hhK1o4d05FWTZzanZVUTgxV3JXMzZNZ293Ymh3NHp
        m R n V Y N l N
        3M2J1L2dKL3JPaDlxVGxvK2xyVS8ybFhIa1UzOWtaSGVFWlNlYUFVVXNGU0RuR
        HA5YXFOUUlQaUI5RWw5Rm96MlpVYnJxUHZxR1hpQ09mVHQ1WE8=",
        "encryptKey":"QmVDU3hyQzdTS1R6SjVhbkJWeFBKQUFoc0x3Y1llR3owMHNoRH
        B3VWwwOFhUMUhaZURoYjczdE94ZnY0MDlEREt3RFRrQXFZWVRkQU1aSmE5c
        kJFOWpEMFB2ZlVYNDkxN3VwR2VhRzcvMjV2QlRYa3ZOamp4eURnbHpYL2svM
        DJiSllCZVA1MmhyV2psT1VSYlVzYVVob2ZyenB4YU5YcHpQcUlYd3ZYb2dtSXFod
        ERzUXBtbWxJWUdXRlhlQjJPM2t6UXN0MURnMHF5eFkrODZjSWROSUlCWFIwT
        ms3bFg5YXI5VnN3N3J5UG81S2ZxUDJWRlJYK3JnaFl4eFNzL3U3QmQrSU9OU0VT
        SFdnYXF3bTlHVWZYNXhmYUxZZ2kxZVlCZXR0RXZIeHlvSXczZ2oyTUsyaGQvM
        ysyQ25SYjRVNEFBOWhFRnFMd3E4YXlFS1d4eWxRPT0=",
        "signData": :
        "228e3b4252bde75792cf7f5ad235a7aa0af8cccf7ba19ed859e6e1c2df240643"
}
```

2. 5. 1. 1. How to Encrypt

```
public static String encrypt(String content) {
        try {

            String aesKey = RandomUtil.randomString(16);

            PublicKey publicKey = KeyFactory.getInstance("RSA")
                    .generatePublic(new
X509EncodedKeySpec(Base64.getDecoder().decode(publicKeyStr)));

            String encryptKey = CryptoUtil.rsaEncrypt(aesKey, publicKey);
            String encryptData = CryptoUtil.aesEncrypt(content, aesKey);
            String signData = CryptoUtil.signWithSha256(content);

            BaseReqDto reqDto = new BaseReqDto();
            reqDto.setEncryptData(encryptData);
            reqDto.setEncryptKey(encryptKey);
            reqDto.setSignData(signData);
            return JSONUtil.toJsonStr(reqDto);
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }
```

encryptData

- Generate an AES encryption key randomly, encrypt the business parameters data JSON string in the request message with AES key, and use the encrypted data as the input parameter encryptData.

  encyptData: encypted (business_parameters, aesKey)

  Samples Business Parameter JSON Payload

```
{
                "transOrderNo":"1730901651412",
                "amt":100,
                "currency":"ZWG",
                "mobileNo":"712153410",
                "goodsName":"testing",
                "notifyUrl":http://10.245.10.1
```

EncyptKey

- Use the OneMoney DPS platform public key to encrypt the AES key, and use it as the input parameter encryptKey.

  encyptKey: encypted (aesKey,platform_public_key)

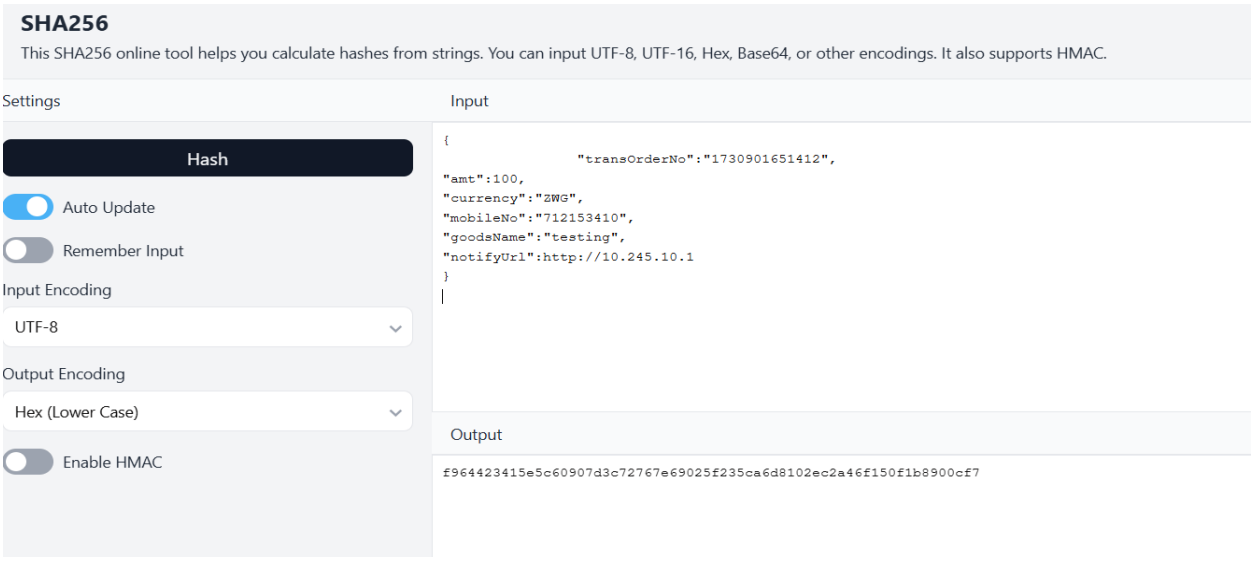
SignData

- Sign the business parameters data JSON string in the request message with **Sha256**, and use it as the input parameter signData. Request the wallet payment platform with a message consisting of the encrypted data and public parameters.

Sha256 Hashing Method

Input Encoding	UTF-8
Output Encoding	Hex(lowercase)

```
public static String signWithSha256(String content) {
    Digester sha256 = new Digester(DigestAlgorithm.SHA256);
    return sha256.digestHex(content);
}
```



SignData: signWithSha256(business_parameters)

2.5.1.2.How to Decrypt

```
public static boolean verifyWithSha256(String content, String sign) {
    if (!StrUtil.isEmpty(content) && !StrUtil.isEmpty(sign)) {
        Digester sha256 = new Digester(DigestAlgorithm.SHA256);
        String sha256Sign = sha256.digestHex(content);
        return sign.equalsIgnoreCase(sha256Sign);
    } else {
        return false;
    }
}
```

```
public static String decrypt(String sn, String ciphertext) {
    try {
        PrivateKey privateKey = KeyFactory.getInstance("RSA")
                .generatePrivate(new PKCS8EncodedKeySpec(Base64.getDecoder().decode(privateKeyStr)));

        BaseResDto vo = JSONUtil.toBean(ciphertext, BaseResDto.class);
        String key = CryptoUtil.rsaDecrypt(vo.getEncryptKey(), privateKey);
        String content = CryptoUtil.aesDecrypt(vo.getData(), key);
        boolean verifyRes = CryptoUtil.verifyWithSha256(content, vo.getSignData());
        if (!verifyRes){
            System.out.println("Signature verification failed");
            return null;
        }
        return content;
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}
```

secretKey: rsaDecrypt (getEncyptKey, privateKey)

decryptData: aesDecrypt (getData, Key)

Verification: verifyWithSha256 (decryptData,getSignData)

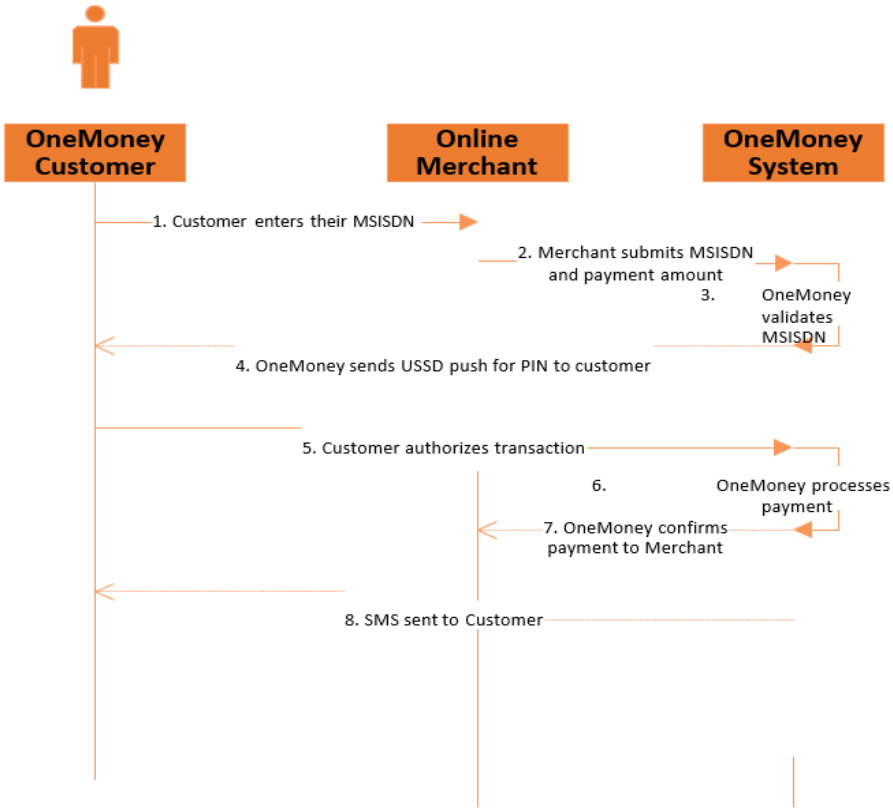
2. C2B Push Interface API

2.1. Interface Introduction

This interface is used by the business platform or third-party to initiate a USSD Push Payment request.

2.2. Interface Description

C2B Push enables customers to perform payment to merchants. During online purchase, customer selects the payment by OneMoney on the third-party channel e.g website. The customer is prompted to enter their account number (Phone Number), the merchant website sends the payment request to OneMoney, and OneMoney sends an USSD Push to customer to collect PIN and confirmation. After collect all the payment information, OneMoney process and send SMS notification to the customer.



2.3. Test environment interface address

http://IP_ADDRESS:PORT/api/pay/payment/push

2.4. Request Interface Parameters

Public parameters

name	type	Mandatory	Parameter Description	Additional
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	

language	String	yes	language English: en Shona: sn Ndebele: nr	
encryptKeyId	String	yes	EncryptedkeyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	**type**	**Mandatory**	**Parameter Description**	**Additional Information**
transOrderNo	string	yes	Payment order number of the business platform, up to 32 characters, supports numbers and letters	Maximum length: 32
amt	BigDecimal	yes	The transaction amount, not null	Greater than 0, maximum 16-digit integer and 2 decimal places
currency	string	yes	Transaction currency, not empty, USD, ZWG, ZAR, GBP, EUR	

mobileNo	String	yes	Personal customer mobile phone number 9-digit mobile phone number without area code	OneMoney registered individual customers
goodsName	String	no	Product name, maximum length 50 characters	
notifyUrl	String	no	Backend notification address, used to receive payment platform results. Can be empty, maximum length is 100 characters	

Sample C2B Push Request


```
{
        "timestamp": 1730899573644,
        "random": "039cfed6-4cce-48b5-9erb-c4c8f44c0988",
        "encryptKeyId": "123456",
        "merNo": "1781526900800749576",

        "encryptData":"c2xnT29xM1hhK1o4d05FWTZzanZVUTgxV3JXMzZNZ293Ymh3NHpmRnVYNl
        3M2J1L2dKL3JPaDlxVGxvK2xyVS8ybFhIa1UzOWtaSGVFWlNlYUFVVXNGU0RuRHA5YXFOUUlQaUI5RWw5Rm96MlpVYnJxUHZxR1hpQ09mVHQ1WE8=",

        "encryptKey":"QmVDU3hyQzdTS1R6SjVhbkJWeFBKQUFoc0x3Y1llR3owMHNoRHB3VWwwOFhUMUhaZURoYjczdE94ZnY0MDlEREt3RFRrQXFZWVRkQU1aSmE5ckJFOWpEMFB2ZlVYNDkxN3VwR2VhRzcvMjV2QlRYa3ZOamp4eURnbHpYL2svMDJiSllCZVA1MmhyV2psT1VSYlVzYVVob2ZyenB4YU5YcHpQcUlYd3ZYb2dtSXFodERzUXBtbWxJWUdXRlhlQjJPM2t6UXN0MURnMHF5eFkrODZjSWROSUlCWFIwTms3bFg5YXI5VnN3N3J5UG81S2ZxUDJWRlJYK3JnaFl4eFNzL3U3QmQrSU9OU0VTSFdnYXF3bTlHVWZYNXhmYUxZZ2kxZVlCZXR0RXZIeHlvSXczZ2oyTUsyaGQvMysyQ25SYjRVNEFBOWhFRnFMd3E4YXlFS1d4eWxRPT0=",

        "signData" : "228e3b4252bde75792cf7f5ad235a7aa0af8cccf7ba19ed859e6e1c2df240643"
}
```


How to Encrypt

encryptData

- Generate an AES encryption key randomly, encrypt the business parameters data JSON string in the request message with AES key, and use the encrypted data as the input parameter encryptData.

  encyptData: {encypted(business_parameters, aesKey)}

  Samples Business Parameter JSON Payload

  ```
  {
  "transOrderNo":"1730901651412",
  "amt":100,
  "currency":"ZWG",
  "mobileNo":"712153410",
  "goodsName":"testing",
  "notifyUrl":http://10.245.10.1
  }
  ```

EncyptKey

- Use the platform public key to encrypt the AES key in RAS mode, and use it as the input parameter encryptKey.

  encyptKey: {encypted(aesKey,plaform_public_key)}

SignData

- Sign the business parameters data JSON string in the request message with Sha256, and use it as the input parameter signData. Request the wallet payment platform with a message consisting of the encrypted data and public parameters.

  SignData: signWithSha256(business_parameters)

2.5. Response Interface Parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the

encryptKey	String	yes	
message	string	no	Cause of failure
status	string	yes	0 for success, non-0 for failure
success	boolean	no	

data

name	**type**	**Mandatory**	**Parameter Description**
transOrderNo	String	yes	Business platform automatic deduction order
orderNo	String	yes	Payment platform automatically deducts order
orderAmt	BigDecima	yes	Transaction amount
currency	String	yes	Easy currency, USD, ZWG, ZAR, GBP, EUR
actAmt	BigDecima	yes	Actual deduction amount
feeAmt	BigDecima	yes	Amount of handling fee paid by the customer
taxAmt	BigDecima	yes	The actual amount of tax paid by the customer
orderStatus	String	yes	Status 10: Initialization, 20: Payment in progress, 30: Success, 40: Failure, 50: Order closed

Sample C2B Push Response


```
{
        "status": "0",
        "messageVariable": [],
        "message": "",
        "code": 200,
        "timestamp": 1732790069672,
        "success": false,
        "data":"WGlhdTFuYUxiOFBrZkhhTGxwUGNGdUxHWkNMdWN5bVh0ZHJPMmJiS
        HhuQzRGdVQrUGZsWWFMdi9kVCtEejNRbC9XaFZWZ29SNFYvb21IRDQ3YnovbE
        1JZlB4K0Fwbk95cVl6OU1QaFR5c3orQXk4VXEwMVRqcXh0Tytnd3R6Z0I=",

        "encryptKey":"aGZiczZLRDlybEs1bEUyV2hhSXd2LzY1MkhqdGVISkdVSXpUWTRw
        UXFtTExNV2wzRDZwUDVZNnE2UE9TRGxmQlpYT2VQU0pCcG9iWTZyY0pnQW
        5GWG5EdURLMzRkRDJnRDVHQVdNRS85bGxZSmFBVnFrMFNZRVJXQlJRdGgzS
```

XdhUE1XQ0hXWDl5aFFSS2hucVNBZm5rUVY3RkRvek9KY3I3RWtEcUFBVGxhY2
pMY2g1amF5YnJ1MVFDUXh4YVFhaTlhcTFGWlU4K2FBZUVPeTFkT0FZUG1GU
HdueHRyTU1mb1BLSWV5ek5mMGhuY3Nwb2RGRlYrOFh5NksxTVBUS1VmRTNE
cXJHenY5UUFRMW4wanJ1SFFrZlNZbUFJWUxDUDJpVTdnbFY0VFNYc2I4YllySm
9HTmRQV2Rseitvd1E5OTJYMURSczU5MWkrRlZSWEF5MDdnPT0=",

"signData":"e58d4cb2b20f2ef492462c61fa5492971363a64a8df4a44826108f459eb27485
"

}

3. C2B Query Transaction API Interface

3.1. Interface Description

This interface is used by the business platform to query C2B Payment transaction results.

3.2. Test environment interface address

http://IP_ADDRESS:PORT/api/pay/payment/order/status/query

3.3. Request Interface Parameters

Public Parameters

name	type	Mandatory	Parameter Description	Additional
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	
language	String	yes	language English: en Shona: sn Ndebele: nr	
encryptKeyId	String	yes	EncryptedkeyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	type	Mandatory	Parameter Description	Additional Information

transOrderNo	string	yes	Payment order number of the business platform, up to 32 characters, supports numbers and letters	Maximum length: 32

Sample


```
{
   "transOrderNo":"1732618067884",

}
```


Sample Request

3.4. Response Interface Parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the following table.
encryptKey	String	yes	
message	string	no	Cause of failure
status	string	yes	0 for success, non-0 for failure
success	boolean	no	

data

name	type	Mandatory	Parameter Description
transOrderNo	String	yes	Business platform automatic deduction order number
orderNo	String	yes	Payment platform automatically deducts order number
orderAmt	BigDecimal	yes	Transaction amount
currency	String	yes	Easy currency, USD, ZWG, ZAR, GBP, EUR
actAmt	BigDecimal	yes	Actual deduction amount
feeAmt	BigDecimal	yes	Amount of handling fee paid by the customer
taxAmt	BigDecimal	yes	The actual amount of tax paid by the customer
orderStatus	String	yes	Status 10: Initialization, 20: Payment in progress, 30: Success, 40: Failure, 50: Order closed

Sample


```
{
        "transOrderNo":"1732618067884",
        "orderNo":"MP2411261248000011",
        "orderAmt":10,
        "currency":"ZWG",
```

```
"actAmt":19.9,
"feeAmt":5,
"taxAmt":4.9,
"orderStatus":"40"

}
```


Sample Response

3.5. Third-party payment interface (B2C)

3.5.1.Interface Description

This interface is used by the business platform for payment, i.e., institutions make payments to customers, international remittances and small loan services.

3.5.2.Test environment interface address

http://IP_ADDRESS:PORT/ api/thirdParty/paying

3.5.3.Interface request parameters

Public parameters

name	type	Is it necessary	Parameter Description	Additional Information
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	

language	String	yes	language English: en Shona : s n Ndebele : nr	
encryptKeyId	String	yes	Encrypted keyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

Samples Business Parameter JSON Payload


```
{
"transOrderNo":"1730901651412",
"orderAmt":100,
"currency":"ZWG",
"remark":" B2C Payment",
"notifyUrl":http://10.245.10.1,
" recCstMobile ":"712984123",
" recCstIdNumber ":"000000000Z00",
" businessType ":"1"

}
```

name	type	Is it necessary	Parameter Description	Additional Information
transOrderNo	string	yes	Payment order number of the business platform, up to 32 characters, supports numbers and letters	Maximum length: 32
orderAmt	BigDecimal	yes	Transaction amount, not empty, greater than 0, maximum 16-digit integer and 2 decimal places	
currency	string	yes	Transaction currency, not empty, USD, ZWG, ZAR, GBP, EUR	
remark	String	no	Order remarks, can be empty, maximum length is 200 characters	
notifyUrl	String	no	Background notification address, can be empty, maximum length is 100 characters	
recCstMobile	String	yes	Mobile phone number of the recipient 9-digit number starting with 7 and excluding area code	
recCstIdNumber	String	yes	Receiving customer ID number	
businessType	String	yes	Business Type 0-Loan Disbursement, 1-Remittance In	

3.5.4.Interface return parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the following table.
encryptKey	String	yes	
message	string	no	Cause of failure
status	string	yes	0 for success, non-0 for failure
success	boolean	no	

data

name	type	Is it necessary	Parameter Description	
transOrderNo	String	yes	Business platform payment order number	
merNo	String	yes	The institution number opened by the business merchant on the payment platform	
successAmt	String	no	Success amount has value when successful	
currency	String	yes	Easy currency, USD, ZWG, ZAR, GBP, EUR	
resultRemark	String	no	Results Notes	
orderStatus	String	yes	Status 10: Initialization, 20: Payment in progress, 30: Success, 40: Failure, 50: Order closed	

3. 6. Third-party payment status query (B2C Query)

3. 6. 1. Interface Description

This interface is used by the business platform to query the results of third-party payment orders.

3. 6. 2. Test environment interface address

http://10.245.10.49:8765/api/thirdParty/paying/order/check

3. 6. 3. Interface request parameters

Public parameters

name	type	Is it necessary	Parameter Description	Additional Information
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	
language	String	yes	language English: en Shona : s n Ndebele : nr	
encryptKeyId	String	yes	Encrypted keyId, assigned by the payment platform to the business platform	

merNo	String	yes	The Organization number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	type	Is it necessary	Parameter Description	Additional Information
transOrderNo	string	yes	Payment order number of the business platform, up to 32 characters, supports numbers and letters	Maximum length: 32

3. 6. 4. Interface return parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the following table.
encryptKey	String	yes	
message	string	no	Cause of failure
status	string	yes	0 for success, non-0 for failure
success	boolean	no	

data

name	type	Is it necessary	Parameter Description	
transOrderNo	String	yes	Business platform collects order number	
merNo	String	yes	The Organization number opened by the business merchant on the payment platform	
successAmt	String	no	Success amount has value when successful	
currency	String	yes	Trading currencies: USD, ZWG, ZAR, GBP, EUR	
resultRemark	String	no	Results Notes	
orderStatus	String	yes	Status 10: Initialization, 20: Payment in progress, 30: Success, 40: Failure, 50: Order closed	

3.7. Merchant Refund(Reversal API)

3.7.1.Interface Description

This interface is used for refunds of C2B payments initiated by the business platform. After the refund order is initiated, the refund is asynchronously refunded to the customer.

3.7.2.Test environment interface address

http://10.245.10.49:8765/api/trade/refund/order/create

3.7.3.Interface request parameters

Public parameters

name	type	Is it necessary	Parameter Description	Additional Information
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	
language	String	yes	language English: en Shona : s n Ndebele : nr	
encryptKeyId	String	yes	Encrypted keyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	

encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	type	Is it necessary	Parameter Description	Additional Information
transOrderNo	String	yes	The refund order number of the business platform, up to 32 characters, can be composed of numbers and letters	Maximum length: 32
originalTransOrderNo	String	yes	The original payment external order number, not empty	
refundAmt	BigDecimal	yes	The refund amount, not null The transaction amount of the original payment transaction	
remark	String	no	Order remarks, can be empty, maximum length is 200 characters	
notifyUrl	String	no	Backend notification address, used to receive refund results from the business platform. Can be empty, maximum length is 100 characters	

3.7.4.Interface return parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the following table.
encryptKey	String	yes	
message	string	no	Cause of failure
status	string	yes	0 for success, non-0 for failure
success	boolean	no	

data

name	type	Is it necessary	Parameter Description
refundOrderNo	String	yes	The payment platform generates a refund order number
transOrderNo	String	yes	Business platform refund order number
refundAmt	String	yes	Refund amount
remark	String	no	Remark

3.8. Merchant refund order verification

3.8.1.Interface Description

This interface provides refund result query for the business platform

3.8.2.Test environment interface address

http://10.245.10.49:8765/api/trade/refund/order/check

3.8.3.Interface request parameters

Public parameters

name	type	Is it necessary	Parameter Description	Additional Information
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	
language	String	yes	language English: en Shona : sn Ndebele : nr	
encryptKeyId	String	yes	Encrypted keyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	
signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	type	Is it necessary	Parameter Description	Additional Information

transOrderNo	string	yes	The refund order number of the business platform, up to 32 characters, can be composed of numbers and letters	Maximum length: 32

3.8.4.Interface return parameters

Public parameters

name	type	Is it necessary	Parameter Description	Additional Information
timestamp	Long	yes	Timestamp when the request was sent	
random	String	yes	Random number, UUID is recommended	
language	String	yes	language English: en Shona : s n Ndebele : nr	
encryptKeyId	String	yes	Encrypted keyId, assigned by the payment platform to the business platform	
merNo	String	yes	The institution number of the business platform on the payment platform	
encryptData	String	yes	Encrypted business parameters	
encryptKey	String	yes	The value of AES Key encrypted by RSA	

signData	String	yes	Signature data is signed by Sha256	

Business Parameters

name	type	Is it necessary	Parameter Description	Additional Information
refundOrderNo	String	yes	Payment platform refund transaction number	
transOrderNo	String	yes	Payment order number of the business platform , up to 32 characters, supports numbers and letters	
orderStatus	String	yes	Order status 10: Waiting for refund 20: Refund in progress 30: Refund successful 40: Refund failed	
currency	String	yes	Transaction currency, not empty, USD, ZWG, ZAR, GBP, EUR	
successAmt	String	no	The successful amount order status is 30: there is a value when the refund is successful	
resultRemark	String	no	Results Notes	

3.9. Transaction result notification

3.9.1. Interface Description

This interface is the address that the business platform actively provides. The payment platform sends a server notification to this address after the order is completed. The business platform updates the order status after receiving the notification. The business platform responds with "SUCCESS" after receiving the successful response. Before receiving the "SUCCESS" from the business platform, a maximum of 10 notifications are sent . The trigger time after the transaction is completed is as follows

0s, 30 s, 1m , 3m, 5m , 7m, 10m, 20m, 30m, 2h

3.9.2.Interface return parameters

name	type	Is it necessary	Parameter Description
data	String	yes	Encrypted business output parameters For detailed data, please see the following table.
encryptKey	String	yes	
message	string	no	Cause of failure

data

name	type	Is it necessary	Parameter Description
transOrderNo	String	yes	Business platform order number
notifyUrl	String	yes	Business platform notification address
merNo	String	yes	The institution number of the business platform after opening an account on the payment platform
orderStatus	String	yes	Order status 10: Initialization, 20: Payment in progress, 30: Success, 40: Failure, 50: Order closed
resultRemark	String	no	Results Notes
successAmt	String	no	Success has value
transType	String	yes	Transaction type 30-payment 50-refund 34-payment on behalf of