API Documentation

ICICI Bank

API Gateway- Security Spec Doc

Document Revision

Date	Version	Description
17-May-2019	0.1	Initial Draft
04-Jun-2019	0.2	Symmetric and Asymmetric Enc-Dec Addition
18-Jun-2019	0.3	Added basic security section

Contents

Introduction:

- This document describes the security implementation details.

Basic Security

- API Key needs to be passed in the header (apikey) and merchant IP will also be required for IP whitelisting.
- API Key to be shared by ICICI.
- API Key needs to be passed http request header (name:- apikey).

API Key Validation

The API key and it's corresponding secret can be used in many ways such as just passing the key in a parameter or header. The most common way of passing the key is as an HTTP parameter or header. So that we will share this key with user and validate the key in request header by comparing it with the value at our end. If the key is Valid we will continue processing the request further otherwise throw an error stating as Invalid API key.

Example:
Passing the Key in Header

Key : API Key
Value: IP84UTvzJKds1Jomx8gIbTXcEEJSUilGqpxCcmnx

IP Whitelisting

IP whitelisting allows us to create lists of trusted IP addresses or IP ranges from which our users
can access our domains. IP whitelist is a security feature often used for limiting and controlling access only to trusted users. So at the time of configuration, We would white list user public IP on our firewall, so our web services would be called by the authorized user server only. Similarly, user also would have to white list our public IP to call the web services.

HTTPS

The web services to be hosted must be HTTPS. The HTTPS type web service would filter out middle man from pipeline while data transportation between bank and user application.

Advanced Security

- API request and response to Merchant is secured using advanced and agreed upon encryption algorithm agreed to maintain data confidentiality and integrity.
- API Gateway uses the standard authenticating and authorizing process for the incoming request from merchant and for maintaining the integrity and confidentiality we apply state of art Encryption/ Decryption algorithm.
- Please refer below given steps and choose any one for Encryption & Decryption process.
    - Asymmetric
    - Symmetric
    - Hybrid (Both Symmetric and Asymmetric)

Asymmetric Encryption & Decryption Process

Payload content-type will be in text/plain. Base64 encoded encrypted Cipher will be passed as a payload.

oG5mU1JJNBuwQaSLKb3wfRZks/cT2Vo2yBNNuqjNHDWEC144WxC8iKqBpJAgq7reFKC4sHNUmNPRD
ya1AvmQ7x1L+3EAdEs9FEWNurZuWTvZpk4y7JrGhg0rz9KptBf+JfJUkSMo7NR3Saxel6EYtckkDr3AG
W7WJZmhcEoAMMXRws/hLVmaNHC/nOjCNqqBd4IOOAzdJh/HADRVI+YAJKT8dE4x9NTI+UX1zAoo
Whza+TsWEHfxzQIa7zai7WSa/wiJD3uD7mk5vT1WY/fKJBquCuzM7l35vigDhmb7dLVLuX8VMiNQrtErW
Nl0uVaag1jg+uZUtyDSxjPFi5yEpKVVc7+T503lDnCvkCFDygqasDsPL24qOjYk4XavTZvwGuPAdYNNkVn
LzVElEhg4zS2ye+fa/8fZiMt/3fwYeN9dgn9i5R6VOFbXSuZJYPSci9k0oqz73h1nzFtps60rUEDoGIkGvm9w
aJU3W78VH5mIdGfGvvJjiKIuVHmi/huzEX9v4w3mW7RDGgmOuKImkqki+XWgyB0JvVmsLdO+cBaym/
seZP3+zdfhO9AWSI2tDLD4Vf0jDjzoDSFN2mzUFgHK9mbtbXgvsnReoGqx/KsivzmZNLmDmtg8eR4Z9Ln
Lni4rI4OtkDv5y/mxMtL3MBUUUajkw6OS6NnhEG895yo

For encryption of request at ICICI:

Encrypted_Payload = Base64Encode(RSA/ECB/PKCS1Encryption(payload,ICICIPubKey.cer))

For encryption of response at ICICI:

Encrypted_Payload = Base64Encode(RSA/ECB/PKCS1Encryption(payload,ClientPubKey.cer))

For decryption of response at Client:

plainResponse = Base64Decode(RSA/ECB/PKCS1Decryption(encPayload,ClientPrivateKey.p12))

Symmetric Encryption & Decryption Process

Payload content-type will be in text/plain. Base64 encoded encrypted Cipher will be passed as a payload.

Api Gateway use below ciphers for symmetric encryption-decryption, Client may can use any of them.

- AES/GCM/NoPadding
- AES/CBC/PKCS5Padding
- DES/CBC/PKCS5Padding
- DESede/CBC/PKCS5Padding

IV= The initialization vector used when encrypting data using the one-time use AES/GCM, AES/CBC ciphers.

Sample Encrypted Payload

wBJSefFsnJVlobh1cJR553w6Ay6b8/2frCjxvdZ1Bsnxztsul7Ha8lFl4PoZD+IhdlRShWdKgz3yJYlisGV/KKpyMSY3DILOpbkqEa0Qq0g=

For encryption of request:

PublicKey (Confidential key) is nothing but randomly generated string of length 16 (OR 32). Will be shared between Client and Api gateway.
IV = Initialization Vector- Exactly 16 bytes actual value to match the block size (Optional)
EncPayload = Base64Encode(AES/CBC/PKCS5Padding(payload,publicKey, IV))

For decryption of response:

Response = Base64Decode (AES/CBC/PKCS5Padding Decryption(encPayload,publicKey, IV))

Hybrid Encryption & Decryption Process

Sample Encrypted Request to be sent to Client (JSON):


```
{
   "requestId": "",
   "service": "LOP",
   "encryptedKey":
"oG5mU1JJNBuwQaSLKb3wfRZks/cT2Vo2yBNNuqjNHDWEC144WxC8iKqBpJAgq7reFKC4sHNUmNPR
Dya1AvmQ7x1L+3EAdEs9FEWNurZuWTvZpk4y7JrGhg0rz9KptBf+JfJUkSMo7NR3SaxeI6EYtckkDr3AG
W7WJZmhcEoAMMXRws/hLVmaNHC/nOjCNqqBd4IOOAzdJh/HADRVI+YAJKT8dE4x9NTI+UX1zAoo
Whza+TsWEHfxzQIa7zai7WSa/wiJD3uD7mk5vT1WY/fKJBquCuzM7I35vigDhmb7dLVLuX8VMiNQrtErW
NI0uVaag1jg+uZUtyDSxjPFi5yEpKVVc7+T503IDnCvkCFDygqasDsPL24qOjYk4XavTZvwGuPAdYNNkVn
LzVEIEhg4zS2ye+fa/8fZiMt/3fwYeN9dgn9i5R6VOFbXSuZJYPSci9k0oqz73h1nzFtps60rUEDoGIkGvm9w
aJU3W78VH5mIdGfGvvJjiKIuVHmi/huzEX9v4w3mW7RDGgmOuKImkqki+XWgyB0JvVmsLdO+cBaym/
seZP3+zdfhO9AWSI2tDLD4Vf0jDjzoDSFN2mzUFgHK9mbtbXgvsnReoGqx/KsivzmZNLmDmtg8eR4Z9Ln
Lni4rl4OtkDv5y/mxMtL3MBUUUajkw6OS6NnhEG895yo=",
   "oaepHashingAlgorithm": "NONE",
   "iv": "",
   "encryptedData":
"wBJSefFsnJVIobh1cJR553w6Ay6b8/2frCjxvdZ1BsnxztsuI7Ha8lFI4PoZD+IhdIRShWdKgz3yJYlisGV/KKp
yMSY3DILOpbkqEa0Qq0g=",
   "clientInfo": "",
   "optionalParam": ""
}
```


Field Description:

Field	Description	Data Type	Max length	Mandatory	Permitted Values
requestId	Unique identifier for the request. Not being stored at any level.	String	64	No	
service type	Service Name ; Identifying the backend service name	String		Yes	
encryptedKey	One-time use AES key encrypted by the Client's public key. Requirement is for a 128-bit key (with 256-bit key	String. Base64-encoded data (case-insensitive)		Yes	

	supported as an option).				
oaepHashing Algorithm	Describes the algorithm used for Asymmetric Encryption of one time AES key.	String	6	Yes	Value : NONE, Meaning : RSA/ECB/PKCS1 is used for Asymmetric Encryption Value : SHA1, Meaning : RSA/NONE/ OAEPWithSHA1AndMGF1Padding OR RSA/ECB/ OAEPWithSHA1AndMGF1Padding

iv	The initialization vector used when encrypting data using the one-time use AES key.	String. Base64-encoded data (case-insensitive)	24	Yes, if IV is not part of encrypted data itself. Leave blank otherwise. For the response data encrypted at ICICI end, IV will always be part of encryptedData itself and it is randomly generated for each request. Can be retrieved by Client by retrieving the first 16 bytes of Base64 decoded encryptedData	Exactly 16 bytes actual value to match the block size
encryptedData	Contains the encrypted dataPayload object containing the business information. Encrypted by the ephemeral AES key using AES/CBC/PKCS5Padding. Sample unencrypted object: Please refer first section of document for a sample object.	String. Base64-encoded data (case-insensitive)		Yes	
clientInfo	ClientIP or other information	String		No	

optionalParam	Reserved for future use	String		No	

For encryption of request at ICICI:

SesionKey = Randomly generated string of length 16 (OR 32).
encryptedKey = Base64Encode(RSA/ECB/PKCS1Encryption(SesionKey,ICICIPubKey.cer))
IV = Initialization Vector- Exactly 16 bytes actual value to match the block size
encryptedData = Base64Encode(AES/CBC/PKCS5Padding(Request,SessionKey, IV))

For encryption of response at ICICI:

encryptedKey = Base64Encode(RSA/ECB/PKCS1Encryption(SesionKey,ClientPubKey.cer))
Session key is nothing but randomly generated string of length 16 (OR 32).
encryptedData = Base64Encode(AES/CBC/PKCS5Padding(Response,SessionKey))

For decryption of response at Client:

IV= getFirst16Bytes(Base64Decode(encryptedData)
SessionKey =
Base64Decode(RSA/ECB/PKCS1Decryption(encryptedKey,ClientPrivateKey.p12,))
Session key is nothing but randomly generated string of length 16 (OR 32) .
Response = Base64Decode (AES/CBC/PKCS5Padding Decryption(encryptedData,SessionKey, IV))

Or

Steps for Encryption

1) Generate 16 digit random number. Say RANDOMNO.
2) Encrypt RANDOMNO using RSA/ECB/PKCS1Padding and encode using Base64. Say encryptedKey.
3) Perform AES/CBC/PKCS5Padding encryption on request payload using RANDOMNO as key and iv- initialization vector. Say encryptedData.
4) Now client may choose to send IV in request from one of below two options.
   a. Send Base64 Encoded IV in "iv" tag. (Recommended Approach)
   b. Send IV as a part of encryptedData itself.

```
bytes[] iv = IV;
bytes[] cipherText = symmetrically encrypted Bytes (step3)
bytes[] concatB = iv + cipherText;
encryptedData = B64Encode(concatB);
```

5) Perform AES/CBC/PKCS5Padding encryption on DATA using RANDOMNO as key and Base64encoded RANDOMNO as IV. Say ENCR_DATA.

Steps for Decryption

1) Get the IV- Base64 decode the encryptedData and get first 16 bytes and rest is encrypted response.

bytes[] IV= getFirst16Bytes(Base64Decode(encryptedData)

2) Decrypt encryptedKey using algo (RSA/ECB/PKCS1Padding) and Client's private key.
3) Decrypt the response using algo (AES/CBC/PKCS5Padding).
4) Ignore first 16 bytes of response, as it contains IV.

cURL command

```
curl -X POST \
  'https://apigwuat.icicibank.com:8443/<relativeURI>' \
  -H 'apikey: <apikey> \
  -H 'content-type: application/json' \
  -d '{
    "requestId": "",
    "service": "",
    "encryptedKey":
"oG5mU1JJNBuwQaSLKb3wfRZks/cT2Vo2yBNNuqjNHDWEC144WxC8iKqBpJAgq7reFKC4sHNUmNPR
Dya1AvmQ7x1L+3EAdEs9FEWNurZuWTvZpk4y7JrGhg0rz9KptBf+JfJUkSMo7NR3SaxeI6EYtckkDr3AG
W7WJZmhcEoAMMXRws/hLVmaNHC/nOjCNqqBd4IOOAzdJh/HADRVI+YAJKT8dE4x9NTI+UX1zAoo
Whza+TsWEHfxzQIa7zai7WSa/wiJD3uD7mk5vT1WY/fKJBquCuzM7I35vigDhmb7dLVLuX8VMiNQrtErW
NI0uVaag1jg+uZUtyDSxjPFi5yEpKVVc7+T503IDnCvkCFDygqasDsPL24qOjYk4XavTZvwGuPAdYNNkVn
LzVEIEhg4zS2ye+fa/8fZiMt/3fwYeN9dgn9i5R6VOFbXSuZJYPSci9k0oqz73h1nzFtps60rUEDoGIkGvm9w
aJU3W78VH5mIdGfGvvJjiKIuVHmi/huzEX9v4w3mW7RDGgmOuKImkqki+XWgyB0JvVmsLdO+cBaym/
seZP3+zdfhO9AWSI2tDLD4Vf0jDjzoDSFN2mzUFgHK9mbtbXgvsnReoGqx/KsivzmZNLmDmtg8eR4Z9Ln
Lni4rl4OtkDv5y/mxMtL3MBUUUajkw6OS6NnhEG895yo=",
    "oaepHashingAlgorithm": "NONE",
    "iv": "",
    "encryptedData":
"wBJSefFsnJVIobh1cJR553w6Ay6b8/2frCjxvdZ1BsnxztsuI7Ha8IFI4PoZD+IhdIRShWdKgz3yJYlisGV/KKp
yMSY3DILOpbkqEa0Qq0g=",
    "clientInfo": "",
    "optionalParam": ""
}'
```