2024

SBI EIS GEN 6 Architecture & Payload Encryption Specification

Dhruvendra Kumar Pandey

SBI: Version 1.2

8/5/2024

Contents

ENCRYPTION AND DIGITAL SIGNATURE 2
Pre-requisites: 2
Implementation and Process Flow: 2
Encryption Steps: 3
Failure Scenarios: 3
Algorithm Specification: 4
REFERENCE SCREENSHOTS: 5

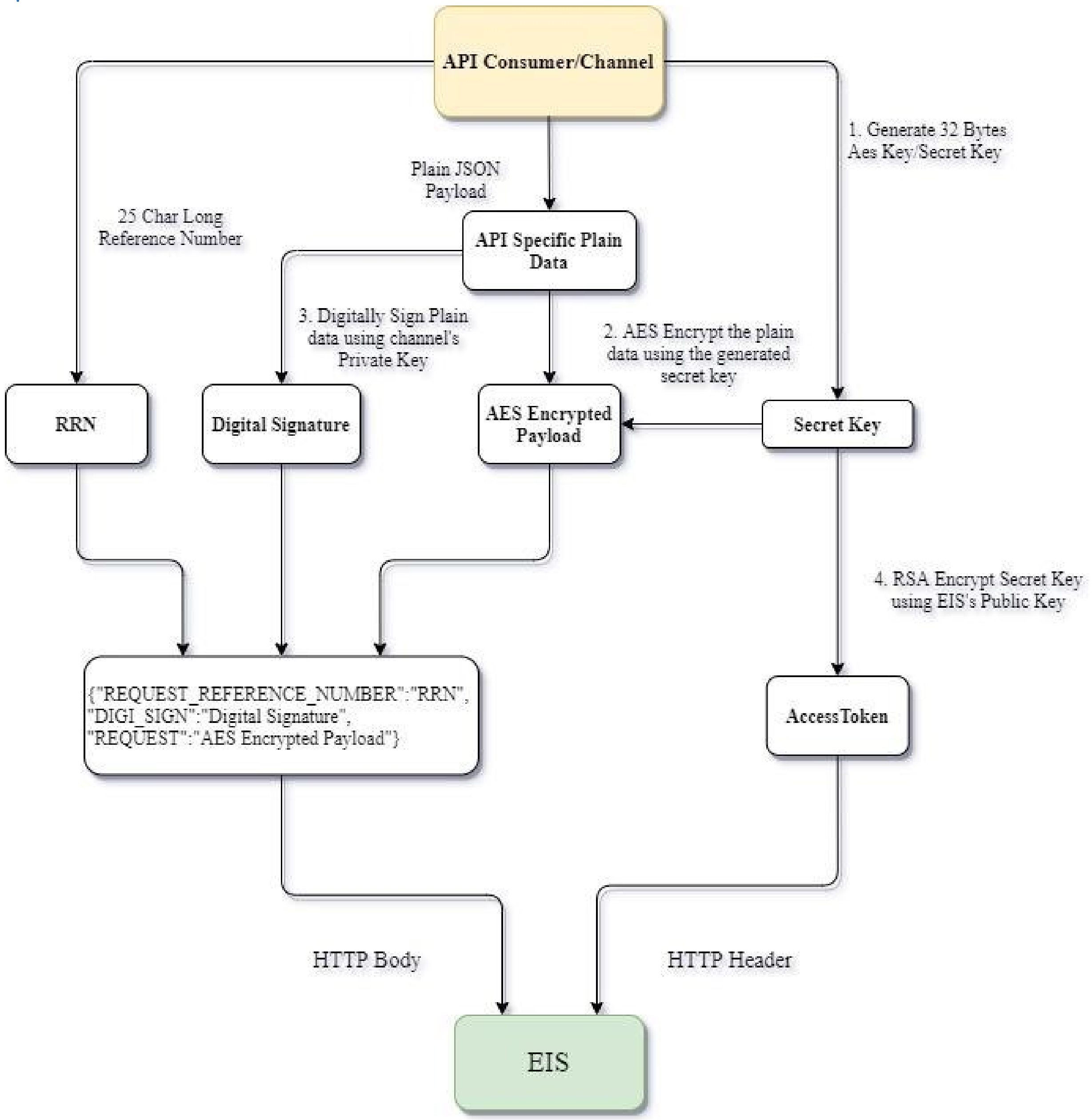
ENCRYPTION AND DIGITAL SIGNATURE

Payload Encryption is a fundamental security practices in SBI under digital communications, particularly crucial in the APIs connecting two heterogeneous systems. It involves encrypting the actual data – the payload – within a message, ensuring the information is secure with CIA standard (i.e. Confidentiality, Integrity & Availability) and protected from unauthorized access during its transmission. A Digital signature—is a mathematical algorithm routinely used to validate the authenticity and integrity of a message in non-repudiation manner.

Pre-requisites:

Two key pairs (Public and Private) will be generated, one at Destination end and other at Source end. Public key for the respective Private Key will be shared between Destination and Source.

Implementation and Process Flow:


API Consumer/Channel
25 Char Long Reference Number
Plain JSON Payload
1. Generate 32 Bytes Aes Key/Secret Key
API Specific Plain Data
3. Digitally Sign Plain data using channel's Private Key
2. AES Encrypt the plain data using the generated secret key
RRN
Digital Signature
AES Encrypted Payload
Secret Key
4. RSA Encrypt Secret Key using EIS's Public Key
{"REQUEST_REFERENCE_NUMBER":"RRN", "DIGI_SIGN":"Digital Signature", "REQUEST":"AES Encrypted Payload"}
AccessToken
HTTP Body
HTTP Header
EIS

Note:
EIS: - **Destination Channel** where API consumed for External/Internal SBI departments.
API Consumer/Channel: means the **requester** who is going to consume destination (i.e. EIS) API. EIS can also be a consumer channels in some cases where Destination will be External/Internal Departments.

Encryption Steps:

1) Source has to generate a 32 character plain text dynamic key (do not use Key generator function for generation of secret key), which will be used for encrypting the plain payload.
2) Encrypt the plain JSON request using AES algorithm with the help of secret key generated in step 1.

3) Source will sign the plain text request using "SHA256withRSA" algorithm along with the private key generated (the respective public key to be shared with Destination). This will provide the digital signature and need to be passed in **DIGI_SIGN** field in request body.
4) Encrypt the secret key generated in step 1 with the help of shared Destination Public key using RSA algorithm.

The generated value needs to be passed in request header parameter **AccessToken**.

5) All the three values (obtained from Steps 2, 3, 4) must be in **Base64 encoding** format and shared with Destination at the time of request in the below mentioned format.

```
{
        "REQUEST_REFERENCE_NUMBER": "SBISI25111900000000000006",
        "REQUEST": "[Outcome of Step (2)]",
        "DIGI_SIGN": "[Outcome of Step (3)]" }
```

Add the encrypted secret key in the Http Header request

```
        AccessToken - [Outcome of Step (4)]
```

6) Destination will validate the request received and decrypt the **AccessToken** with RSA algorithm to get the secret key (AES Key).
7) Decryption of **REQUEST** using the secret key obtained from decryption of AccessToken in step 7.
8) Destination will verify **DIGI_SIGN** received in the request body with the shared Source Public Key.
9) On Successful decryption and verification, Destination will move on for valid request processing. Depending on success/failure Destination will forward the response to the Source with 200 HTTP status code.

```
{
     "RESPONSE": "[Encrypted Response (Encrypted with the key obtained from the decryption of
 AccessToken)]",
"REQUEST_REFERENCE_NUMBER":"SBISI25111900000000000010",
     "RESPONSE_DATE": "26-11-2019 13:10:17",
     "DIGI_SIGN": "[Signed Data (Singed on Plain Response)]" }
```

Failure Scenarios:

Failures from Destination server may arises in two ways.

a **Failure at Destination Server end:**
Failure at destination server end includes server down, application not hosted, OR 404 – not found, 400 Bad request, timeout cases, network errors etc... such request received from Source will return ERROR_CODE like XX011 with 401 HTTP code

Ex.

```
{
   "REQUEST_REFERENCE_NUMBER":        "SBIXX25111900000000000011",
"ERROR_CODE": "XX051",
"ERROR_DESCRIPTION": "Unable to process due to technical error!!"
}
```

Response HTTP Header

X-Original-HTTP-Status-Code - 401

b **Failure at Destination Application end:**

Failure at destination application end includes application specific errors which should be followed proper naming convention guidelines.

10) Finally Source must decrypt the RESPONSE using the same AES 256 encryption key (secret key) which was used in for REQUEST and verify the DIGI_SIGN with SHA256 RSA algorithm with shared Destination Public Key.
11) Response will be sent back Destination (i.e. EIS as per flow) under encrypted format only .
12) Sample Request format

Header:- [

```
“content-type”: “application/json”,
“AccessToken”:
“bCYd64YS2ggAixhVFDd+afXWl8/wgGKOChPnntVShyAGUKOJFLLj9FqLB2Yv2N9OpGZvkuK8ZyoLtGrJGdscZAgL98
1ZQPTxRNmTZ6FLsURGPWMvxu7gGJHGTBoxBGIAbHoVRDtN9q4Cd8cW1+5T9p7P5VCjgXQhYcyxFZQJNRQ26eEjPhr/R
mf+U5fGpQcaZ5oMN1oJ42RcFVVh7uPF+us1i57sOS+2jC94G+F1RuHhsucze5ciFjPiUoMFah54JHP3eilu14LvHDTd
eKV2hq9zcRHXKn9dxZhiYtZWEpInZRDJl1QXwE5vfECst/skQ4qUFABcUkGYvfVyH7UalQ==”
]
```

Body :-

```
{
"REQUEST_REFERENCE_NUMBER":" SBIXX2 XXX183202200000001",

"REQUEST":"gbdyC1yDxuMjH8ImhWaAzyZ4O+6MJw4UMmS2mj+HhHzrx1+JWQrgdDSd59URYNyPBdR0qJeWCsvWpfOX
iWH32jOgAzTNAFbeiR3N7gNghdU+vObl5b7h9sFwQlI8PfAyyYjxhvRkjB8\/CLkftMywIIHlL249U7\/J2IThnXjum
331viawaX2NFOvTXAcOaZhstHOIlyhHtwbVh8ofFndHfw==",

"DIGI_SIGN":"bdm76ugSXugGziQlMd2Vse+wWy9Mis3D9eob5TnpAOAxJhXUq4BfKnl2ITGHQbjqhnitsv4xvzuodz
O0YxdClwdVNcamETXVw\/g9E1r2iwWooAULWgUycgYVDLfZ3b7bRfhckOSNTwSQYUDtoiE0iJwMo8PVhrMjrPPAIV58
+nNUmGeUIAT6hWyfhdlBrwhRw8ud\/Ak2sGZIBM8yEFkBR0+lxGXYyAbe2hPl3zpwXjrQIh1rkkW9B1HsvSzZzWSrvq
oCc0joEyr5p3X3fVuyCYusZcPsvJPYeHFfg8X\/zHac8QUXnN7sDaZoCH83cPUTjegB6b49WKUEEsDxbWEhUQ=="
}
```

Algorithm Specification:

Advanced Encryption Standard (AES) for Payload Encryption.

• Cipher Mode Operation	: Galois/Counter Mode (GCM) with No Padding
• Cryptographic Key	: 256 bits*
• IVector	: First 12 byte of cryptographic key (Secret Key)
• GCM Tag Length	: 16 Bytes

Rivest-Shamir-Adleman (RSA) for AES Key Encryption.

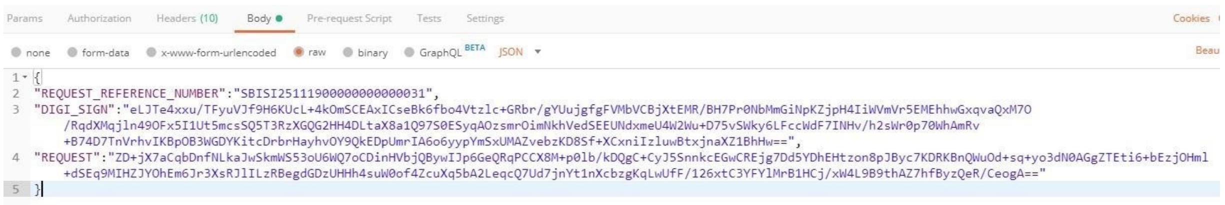
• Cipher Mode Operation	: Electronic Codebook (ECB) with OAEPPadding
• Cryptographic Key	: 2048 bit X509 Certificate

SHA256-Rivest-Shamir-Adleman (RSA) for Digital Signature.

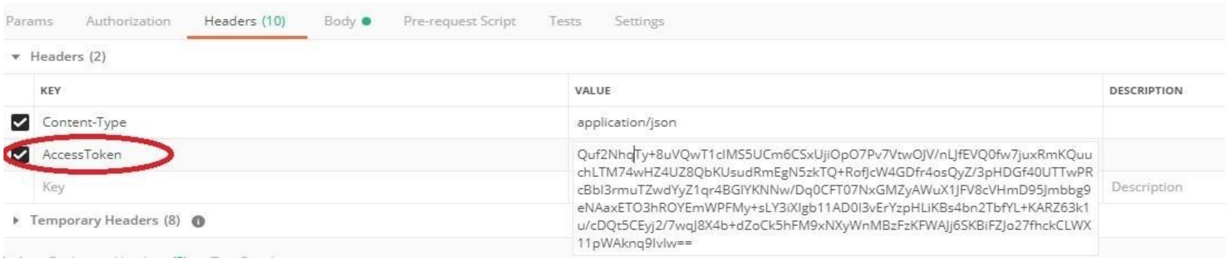
• Hashing Algorithm	: SHA 256
• Cryptographic Key	: 2048 bit X509 Certificate

REFERENCE SCREENSHOTS:

➢ Prepare JSON request using encrypted payload and encrypted key as shown below



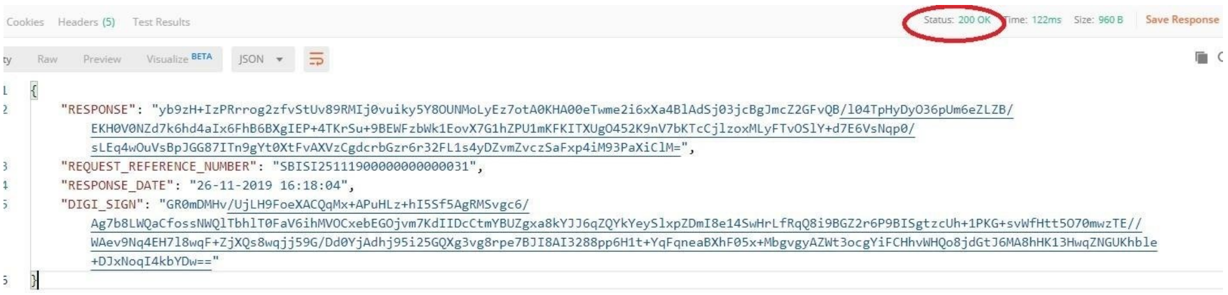
```
{
"REQUEST_REFERENCE_NUMBER":"SBISI25111900000000000031",
"DIGI_SIGN":"eLJTe4xxu/TFyuVJf9H6KUcL+4kOmSCEAxICseBk6fbo4Vtzlc+GRbr/gYUujgfgFVMbVCBjXtEMR/BH7Pr0NbMmGiNpKZjpH4IiWVmVr5EMEhhwGxqvaQxM7O
    /RqdXMqjln49OFx5I1Ut5mcsSQ5T3RzXGQG2HH4DLtaX8a1Q97S0ESyqAOzsmrOimNkhVedSEEUNdxmeU4W2Wu+D75vSWky6LFccWdF7INHv/h2sWr0p70WhAmRv
    +B74D7TnVrhvIKBpOB3WGDYKitcDrbrHayhvOY9QkEDpUmrIA6o6yypYmSxUMAZvebzKD8Sf+XCxniIzluwBtxjnaXZ1BhHw==",
"REQUEST":"ZD+jX7aCqbDnfNLkaJwSkmWS53oU6WQ7oCDinHVbjQBywIJp6GeQRqPCCX8M+p0lb/kDQgC+CyJ5SnnkcEGwCREjg7Dd5YDhEHtzon8pJByc7KDRKBnQWuOd+sq+yo3dN0AGgZTEti6+bEzjOHml
    +dSEq9MIHZJYOhEm6Jr3XsRJlILzRBegdGDzUHHh4suW0of4ZcuXq5bA2LeqcQ7Ud7jnYt1nXcbzgKqLwUfF/126xtC3YFYlMrB1HCj/xW4L9B9thAZ7hfByzQeR/CeogA=="
}
```



	KEY	VALUE	DESCRIPTION
✓	Content-Type	application/json	
✓	AccessToken	Quf2NhqTy+8uVQwT1cIMS5UCm6CSxUjiOpO7Pv7VtwOJV/nLJfEVQ0fw7juxRmKQuuchLTM74wHZ4UZ8QbKUsudRmEgN5zkTQ+RofJcW4GDfr4osQyZ/3pHDGf40UTTwPRcBbI3rmuTZwdYyZ1qr4BGIYKNNw/Dq0CFT07NxGMZyAWuX1JFV8cVHmD95Jmbbg9eNAaxETO3hROYEmWPFMy+sLY3iXIgb11AD0I3vErYzpHLiKBs4bn2TbfYL+KARZ63k1u/cDQt5CEyj2/7wqJ8X4b+dZoCk5hFM9xNXyWnMBzFzKFWAJj6SKBiFZJo27fhckCLWX11pWAknq9Ivlw==	
	Key		Description

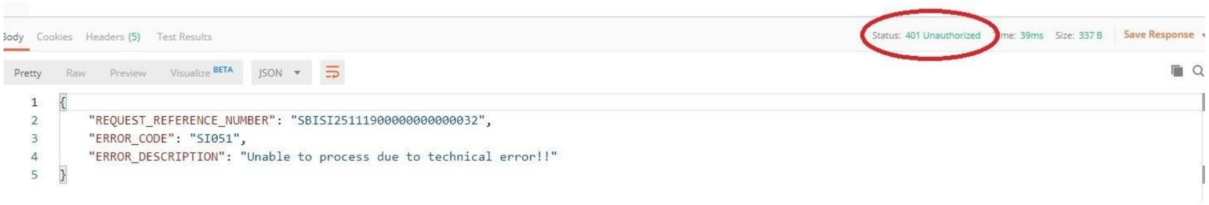
NOTE: - Payload is passed in body as *REQUEST* field, Signature is passed in body as *DIGI_SIGN* field and encrypted key in header as *AccessToken*.

➢ Encrypted response format after successful processing of the request.



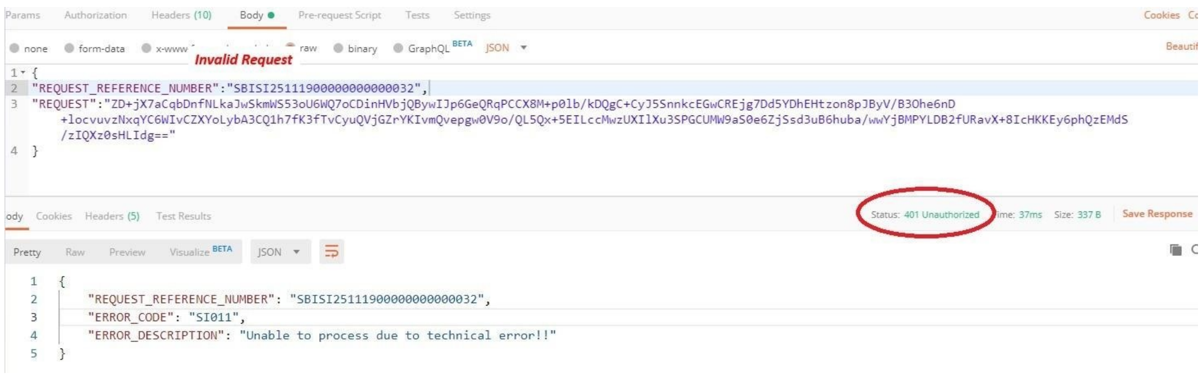
```
{
    "RESPONSE": "yb9zH+IzPRrrog2zfvStUv89RMIj0vuiky5Y8OUNMoLyEz7otA0KHA00eTwme2i6xXa4BlAdSj03jcBgJmcZ2GFvQB/l04TpHyDyO36pUm6eZLZB/
        EKH0V0NZd7k6hd4aIx6FhB6BXgIEP+4TKrSu+9BEWFzbWk1EovX7G1hZPU1mKFKITXUgO452K9nV7bKTcCjlzoxMLyFTvOSlY+d7E6VsNqp0/
        sLEq4wOuVsBpJGG87ITn9gYt0XtFvAXVzCgdcrbGzr6r32FL1s4yDZvmZvczSaFxp4iM93PaXiClM=",
    "REQUEST_REFERENCE_NUMBER": "SBISI25111900000000000031",
    "RESPONSE_DATE": "26-11-2019 16:18:04",
    "DIGI_SIGN": "GR0mDMHv/UjLH9FoeXACQqMx+APuHLz+hI5Sf5AgRMSvgc6/
        Ag7b8LWQaCfossNWQlTbhlT0FaV6ihMVOCxebEGOjvm7KdIIDcCtmYBUZgxa8kYJJ6qZQYkYeySlxpZDmI8e14SwHrLfRqQ8i9BGZ2r6P9BISgtzcUh+1PKG+svWfHtt5O70mwzTE//
        WAev9Nq4EH7l8wqF+ZjXQs8wqjj59G/Dd0YjAdhj95i25GQXg3vg8rpe7BJI8AI3288pp6H1t+YqFqneaBXhF05x+MbgvgyAZWt3ocgYiFCHhvWHQo8jdGtJ6MA8hHK13HwqZNGUKhble
        +DJxNoqI4kbYDw=="
}
```

Authorization/Authentication Failure response format before processing the request.



```
{
    "REQUEST_REFERENCE_NUMBER": "SBISI25111900000000000032",
    "ERROR_CODE": "SI051",
    "ERROR_DESCRIPTION": "Unable to process due to technical error!!"
}
```

➢ Validation Failure response format before processing the request.



```
{
"REQUEST_REFERENCE_NUMBER":"SBISI25111900000000000032",
"REQUEST":"ZD+jX7aCqbDnfNLkaJwSkmWS53oU6WQ7oCDinHVbjQBywIJp6GeQRqPCCX8M+p0lb/kDQgC+CyJ5SnnkcEGwCREjg7Dd5YDhEHtzon8pJByV/B3Ohe6nD
    +locvuvzNxqYC6WIvCZXYoLybA3CQ1h7fK3fTvCyuQVjGZrYKIvmQvepgw0V9o/QL5Qx+5EILccMwzUXIlXu3SPGCUMW9aS0e6ZjSsd3uB6huba/wwYjBMPYLDB2fURavX+8IcHKKEy6phQzEMdS
    /zIQXz0sHLIdg=="
}
```

Status: 401 Unauthorized Time: 37ms Size: 337 B

```
{
    "REQUEST_REFERENCE_NUMBER": "SBISI25111900000000000032",
    "ERROR_CODE": "SI011",
    "ERROR_DESCRIPTION": "Unable to process due to technical error!!"
}
```

* **Note**:

1. Kindly do not use Key generator function for generation of cryptographic key, only use keyboard characters of the appropriate length.
2. SBI EIS also recommends to develop standard error code some like 5 digit unique code followed by description ex. XX001 – Request not allowed.. and not to use EIS SI error like SI001, SI002 etc. avoid confusion and clearly differentiate between errors given by external channels vs SBI EIS.