verifone

3-D Secure JWT

The integration uses JWT's as a method of authentication between the merchant and the 3DS Server. In this section the generation and validation of a JWT are discussed.

Create JWT to initialize the JavaScript

In order to initialise the JavaScript a valid JWT is required. Any library supporting JSON Web Signature can be used. The JWT uses as signature a SHA-256 HMAC hash algorithm and must be created on the server side for security reasons. During the onboarding to the 3DS service, a merchant will receive 3 values used for authentication.

Claim	Description
API identifier	A non-secure value that should be passed within the JWT under the 'iss' (Issuer) claim.
Org Unit Id	A non-secure value that should be passed within the JWT under the OrgUnitId claim.
API Key	A secure value that should never be rendered or displayed anywhere your users could find it. The API Key should only be used to sign the JWT and to verify a JWT signature from the javascript. It should never be included within the JWT itself.

Required claims

Claim name	Description
jti	A unique identifier for this JWT. This field should change each time a JWT is generated.
iat	The epoch time in seconds of when the JWT was generated. This allows us to determine how long a JWT has been around and whether we consider it expired or not.
iss	An identifier of who is issuing the JWT. We use this value to contain the Api Key identifier or name.
OrgUnitId	The merchant SSO OrgUnitId
Payload	The JSON data object being sent to the JavaScript. This object is usually an Order object
ReferenceId	This is a merchant supplied identifier that can be used to match up data collected from the 3DS Server. 3DS Server can then use data collected to enable rules or enhance the authentication request. This value should be parsed as 'device_info_id' in the lookup request.

Optional claims

Claim name	Description
ObjectifyPayload	A boolean flag that indicates how 3DS Server Api should consume the Payload claim. When set to true, this tells 3DS Server the Payload claim is an object. When set to false, the Payload claim is a stringified object. Some JWT libraries do not support passing objects as claims, this allows those who only allow strings to use their libraries without customization
exp	**Expiration** - The numeric epoch time that the JWT should be consider expired. This value is ignored if its larger than 4 hrs. By default we will not consider any JWT older than 4 hrs.

Other claims

Claim name	Description
ConfirmUrl	The merchant endpoint that will receive the post back from the payment brand that contains the 3DS Server API response JWT describing the result of redirecting to the payment brand.

JWT example

```
{
"jti": "a5a59bfb-ac06-4c5f-be5c-351b64ae608e",
"iat": 1448997865,
"iss": "56560a358b946e0c8452365ds",
"OrgUnitId": "565607c18b946e058463ds8r",
"Payload": {
"OrderDetails": {
"OrderNumber": "0e5c5bf2-ea64-42e8-9ee1-71fff6522e15",
"Amount": "1500",
"CurrencyCode": "840"
}
},
"ObjectifyPayload": true,
"ReferenceId": "c88b20c0-5047-11e6-8c35-8789b865ff15",
"exp": 1449001465,
"ConfirmUrl": 'https://mywebsite.com/confirmHandler'
}
```

JWT Validation

When the JWT is received in the payments.validated event, the Response JWT shall be sent to the merchant’s **backend** to verify and obtain the results.

Claim	Description
aud	**Merchant jti Id** - This is the 'jti' field from your request JWT echoed back. This field allows you to match up your request JWT with Cardinals response JWT.
jti	**JWT Id** - A unique identifier for this response JWT. This value is generated by Cardinal.
iat	**Issued At Time** - This is a timestamp of when the JWT was created.
iss	**Issuer** - The request JWT's iss field echoed back.
ConsumerSessionId	The unique session Id for the current user.
Payload	The response object for your request. This field will contain any actual state information on the transaction. This is the decoded data object that is passed into the `payments.validated` event as the first argument.

Example of JWT Payload


```
{
"iss": "56560a358b946e0c8452365ds",
"iat": 1471014492,
"exp": 1471021692,
"jti": "8af34811-f97d-495a-ad19-ec2f68004f28",
"ConsumerSessionId": "0e1ae450-df2b-4872-94f7-f129a2ddab18",
"Payload": {
"Validated": true,
"Payment": {
"Type": "CCA",
"ExtendedData": {
"CAVV": "AAABAWFlmQAAAABjRWWZEEFgFz+=",
"ECIFlag": "05",
"PAResStatus": "Y",
"SignatureVerification": "Y",
"XID": "MHEyQjFRQkttemdpaFlRdHowWTA=",
"Enrolled": "Y"
}
},
"ActionCode": "SUCCESS",
"ErrorNumber": 0,
"ErrorDescription": "Success"
```

```
}
}
```

Stringified JWT Sample

```
{
"iss": "56560a358b946e0c8452365ds",
"iat": 1471015342,
"exp": 1471022542,
"jti": "55ebfa2a-665f-4d6b-81ea-37d1d4d12d9e",
"ConsumerSessionId": "fb3a97a3-0344-4d3d-93ea-6482d866ec97",
"Payload":
"{\"Validated\":true,\"Payment\":{\"Type\":\"CCA\",\"ExtendedData\":{\"CAVV\":\"AAABAWFlmQAAAABjRWWZEEFgFz+\\u003d\",\"ECIFlag\":\"05\",\"PAResStatus\":\"Y\",\"SignatureVerification\":\"Y\",\"XID\":\"MFpjUVpwb0FXcHdwMWJBdldwNDA\\u003d\",\"Enrolled\":\"Y\"}},\"ActionCode\":\"SUCCESS\",\"ErrorNumber\":0,\"ErrorDescription\":\"Success\"}
"
}
```