NICE Privacy and Security Specification

Version 1.1

Scenera, Inc., Nikon Corporation, Sony Semiconductor Solutions Corporation, Wistron Corporation and Hon Hai Precision Industry Co., Ltd.(NICE Alliance Promoters) contributed to this document.

Edit History

Version	Date	Comments
1.1	26 Jan 2022	Initial Version 1.1 release

Table of Contents

1. Scope

This document provides an overview of security in the NICE System. It includes the overall threat model for the NICE System and the roles of the subsystems in the NICE System with regards to security.

It also describes how the privacy of data generated by Devices and Data Services is managed. This includes how data is encrypted, how the keys that are used to encrypt this data are managed and distributed and the rules that define how data may be analyzed and distributed. The scope of this specification includes Devices, Data Services and NICE Apps.

2. Overview

The NICE System provides for the security of the Data generated by Devices and Data Services. It also provides for the access control to Devices, Data Services and Apps. Each Entity in the NICE System has an EndPointID, a Private Key used for signing objects generated by the Device and used to decrypt objects sent to the Device. The NICE License Authority provides these credentials for Devices, while the NICE Account Service provides these credentials for Data Services and Apps.

The NICE System makes use of the Private Key and Public Key to distribute Content Keys to manage the encryption of Data in the NICE System. NICE also makes use of the Private Key and Public Key to protect access tokens that are used enable access to NICE Devices or Data Services. The NICE License Authority and NICE Account Service manages which Certificate Authorities shall be considered to be valid when a NICE Device communicates using TLS. This enables NICE Devices to communicate via TLS with other Entities which are not part of the NICE eco system.

The NICE System makes use of standard methods for the encryption of SceneData and SceneMarks. It makes use of TLS and DTLS to protect data links between devices. Access Tokens conform to the JSON Web Token format.

3. High Level Threat Model

To ensure the security of Data, there are several aspects relating to security that need to be handled by the NICE specification. The major security assets in the system are the user's account, the devices, data (SceneMarks and SceneData) and Applications. See the following data table.

Major security assets and threats with counter measures in the NICE system			
Asset	**Attack**	**Attacker Motivation**	**Counter Measure**
Account	Hijack of Account	Ransom	Managed Credentials
	Full access to devices and data	Invasion of Privacy	Revocable Access Tokens with limited access and time of access.
		Disable Physical Security	
		Fraud	
Device	Access Data	Ransom	Managed Credentials
	Disable/Reconfigure	Invasion of Privacy	Managed Keys
	Rogue Computing	Disable Physical Security	Implementation Requirements
	Denial of service, Cryptocurrency Mining	Fraud	
		Denial of Service attacks	
		Cryptojacking	
Data	Access Data	Ransom	Data Encryption
	Publish Data, use data to commit crimes	Invasion of Privacy	Privacy Management Service
	Modify Data	Fraud	*Limited access to data*
		Social Engineering	*Defines how data may be used*
			Implementation Requirements
App	Fake App	Implement attacks listed above	Manage Unique App Key
	Steal User Credentials		Managed Credentials
	Reverse Engineering of App		Privacy Management Service
	Steal App Credentials		Implementation Requirements

Table 1. Major Security Assets and Threats with Counter Measures in the NICE System

The following functionality is used to protect these assets:

- **Device:**
  - **Device Credential Management (Managed Credentials & Revocable Access Tokens with limited access and time of access)**
- **User Account:**
  - **Access Control (Managed Credentials & Revocable Access Tokens with limited access and time of access)**
- **Data:**
  - **Privacy Management Service (Data Encryption, Privacy Management Service, *Limited access to data, Defines how data may be used,* Implementation Requirements)**
- **App Provider:**
  - **Credential Management (Managed Credentials)**
- **App Instance:**
  - **Application Key (Manage Unique App Key, Implementation Requirements)**
  - **Credential Management (Managed Credentials)**
  - **Privacy Management Service**

▲ Top

4. Trust Management

Key to any security system are the roles of parties in the system and how trust is managed. This section describes the hierarchy of entities that manage trust and how end users and devices fit into the trust model.

The highest authority in the NICE System is the NICE Licensing Authority (NICELA). This authority issues certificates to participants in the NICE System. The certificates used to create this hierarchy conform to the X.509 standard.

The following roles are defined in the NICE ecosystem:

- **NICE Licensing Authority**
  - Certificate Authority
  - Credential Generation and Distribution
- **NICE Account Service**
  - Account Management Service
  - Access Control Service
  - Privacy Management Service
  - Network Security Service
- **Device Manufacturer**
- **Device Seller**
- **App Developer**
- **App Instance**

4.1. NICE Licensing Authority

The NICE Licensing Authority provides root certificates for the participants in the NICE Ecosystem. It effectively brokers trust between the Device manufacturers and NICE Apps.

- Each **Device** in the NICE Ecosystem is issued the following by the NICELA:
    - **certificate and private signing key** used for signing objects generated by the Device and used for decrypting data sent to the Device.
- Each **NICE Account Service** in the NICE Ecosystem is provided **certificates** for validating the public key used to validate objects issued by the NICE Account Service.

4.2. NICE Account Service

Each NICE Account Service is issued certificates and private keys by the NICE licensing authority. One of these keys is used for encrypting objects sent to the NICEAS and the second is used for validating objects sent from the NICEAS. The NICE Account Service operates the Account Management Service, User Access Control Service, Privacy Management Service and Network Security.

4.2.1. Account Management Service

The Account Management Service manages the User's account settings and records. It maintains a list of Apps and Devices linked to a User's account.

4.2.2. Developer Portal

The Developer Portal enables the developer to register applications on one or more Account Management Services. Once the AppID has been registered on a NICE AS it is possible for users on the NICE AS to link the registered App to their Accounts.

4.2.3. DataPipelineController

The DataPipelineController configures the processing path from a Node through to an Application. The DataPipelineController will provide the DeviceNode with its SceneMode and configure the cloud component of the DataPipeline to perform further processing of SceneMarks in the cloud and define which apps will receive data from the created data pipelines.

4.2.4. User Access Control Service

This is responsible for managing access control the End User account credentials.

- It shall provide a high priority notification to the end user if and when the control of the device is transferred.

4.2.5. Privacy Management Service

This service is provided by the NICE Account Service.

- It manages the security of data generated by any party in the system.
- A Device that creates Scene Data uses SceneEncryptionKeys provided to it by the Privacy Management Service to encrypt the Scene Data that it generates.
- The Privacy Management Service manages who can access the Scene Data and what the entity may do with the Scene Data by issuing Privacy Objects which contain the SceneEncryptionKeys

that enable the Entity to decrypt the data and rules that determine what the Entity may do with the decrypted Scene Data.
- The Privacy Objects may also determine whether the Entity may export data and if so the SceneEncryptionKeys that shall be used to encrypt the exported data.

The Privacy Management Service controls which Entities may access this Scene Data by using Privacy Objects to control access to it.

4.3. Device Manufacturer

The Device Manufacturer provides a Device that is compliant to the NICE Specifications.

The Device Manufacturer:

- Performs conformance testing of the product to ensure that the product is functionally conforms to the specifications provided by the NICELA.
- Performs an audit on its implementation to ensure that it conforms to the Security Implementation Requirements defined by the NICELA.

The Device Manufacturer embeds a unique ID, Private Key into a Device.

- These are provided by the NICELA.
- These are used to set up secure communications with device and to enable the Device to validate itself when interacting with the NICE Account Service.

4.4. Device Seller

The Device Seller is the entity that brands and markets the device.

- The NICE device Seller notifies the NICELA of the devices that it is selling into the market.
- If this information is not provided the NICELA will not enable activation of the device onto a NICE Account Service.

The Device Seller is the brand or distributor who actually markets the product and manages any software updates in the field. The NICELA will only issue the X.509 certificate for a device and assign the device to a NICE Account Service if the device has been registered with the NICELA by the Device Seller. This registration is a backend process performed between the device seller and the NICELA.

▲ Top

4.5. App Developer

The App Developer develops and distributes Apps that make use of the NICE System to consume SceneData and SceneMarks. The App Provider may have backend servers that process Data that is generated by Devices and Data Services associated with an End Users Account. If these backend servers are to process Data that is protected using the Privacy Management Service, the App Developer is required to have its own AccountD, X.509 certificates and Private Key to enable it to participate in the Privacy Management Service. The App Developer's system shall abide by the rules defined for processing data in the Privacy Objects. The App Developer shall protect data that it redistributes to App

Instances in accordance to the rules defined in the Privacy Object. This may be managed under the Privacy Management Service or an equivalent protection of Data distributed to App Instances.

In the case of the App instance processing data in a secured manner the App Instance shall have an AppInstanceID and Private Keys and Certificates for a key pair that is used for encrypting data sent to the AppInstance validating objects sent from the AppInstance.

▲ Top

4.6. AppInstances

The NICE Account Service controls the access that Apps have to data and devices associated with the User's Account.

- The configuration of Scene modes is controlled through the OAuth protocol, while the access to Scene Data is controlled through the Privacy Management Service.
- Where the App is a client to the Privacy Management Service, it is issued a certificate by the NICE Account Service. The key is generated by the App itself .

5. Security Objects Used in the NICE Ecosystem

The following objects are related to security in the NICE Ecosystem:

- The **DeviceSecurity Object** is issued by the NICELA to the device manufacture to enable the manufacturer to embed credentials into the device during manufacture.
- The **ManagenemtEndPoint Object** is issued by NICELA to the Device to indicate where it should get Management Object.
- The **Management Object** is issued by the NICELA and binds a device to a NICE Account Service.
- The **DeviceControl Object** configures the device to be managed by the NICE Account Service, the security settings of the device with respect to the security of the device on the network.
- The **SceneMode Object** configures the Node to be managed by the DataPipelineController.
- The **AppSecurity Object** enables the app developer or an app instance to access NICE API's and process NICE data.
- The **AppControl Object** configures the device to be managed by the NICE Account Service.
- The **AccessToken Object** contains Access Tokens as defined by JSON Web Token which enable an App or Service to access another Service or Device.
- The **Privacy Object** is provided to a device, app or service to enable the device, app or service to either generate scenedata or process scenedata.
- The **DeviceSeller Object** feeds back to the NICELA which devices the seller will be selling to consumers. This object is not encrypted or authenticated as it is used for administrative purposes only.
- The **TrustedTimeResponse Object** are used to synchronize the time stamp with the NICE LA.

These objects except for DeviceSeller Object are always encrypted using the Public Key of the Entity to which they are addressed.

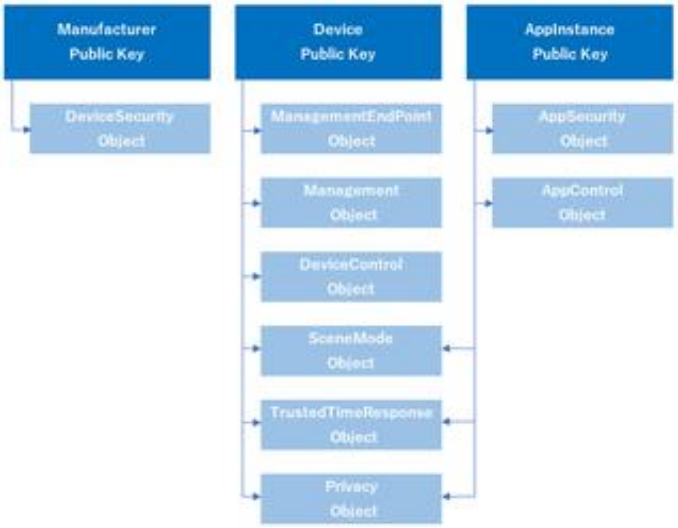

Figure 1. Keys Used to Encrypt Objects.

The **authentication** of the objects depends on the object.

- The **DeviceSecurity Object**, **ManagementEndPoint Object**, and **Management Object** are signed by the NICE License Authority.
- The **TrustedTimeResponse Object** and **AccessToken Object** are signed by either the NICE License Authority or NICE Account Service.
- The **SceneMode Object** is signed by DataPipelineController.
- The other Objects are signed by the NICE Account Service.

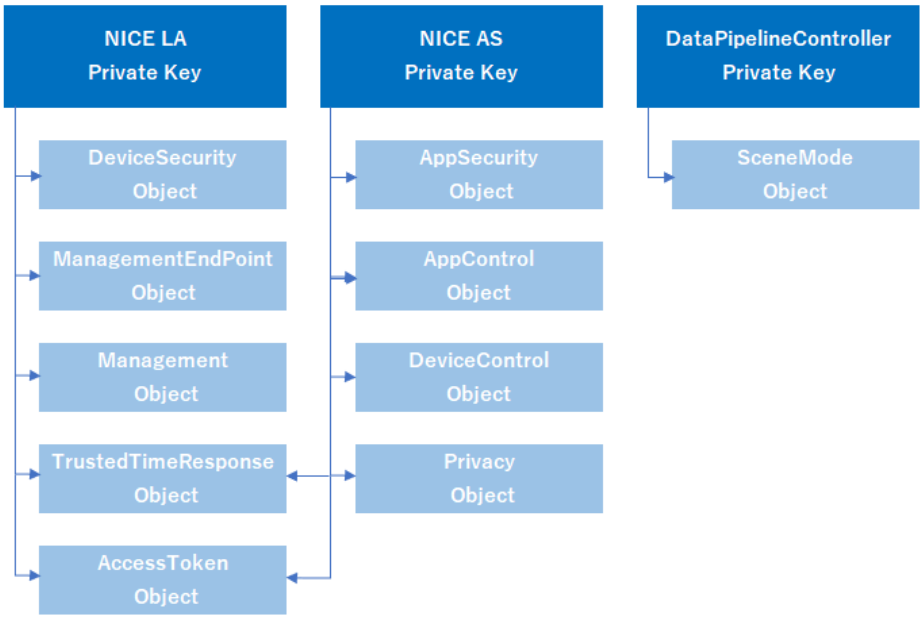

Figure 2. Keys used to validate Security Objects

▲ Top

6. Privacy Management

6.1. Overview

Devices generate Data in the form of SceneMarks and SceneData. Data Services and Apps may receive this Data, further process it and feed it onto other Data Services or Apps. The NICE specifications require that the data links between Entities shall be encrypted using TLS. The Privacy Management Service manages the security of Data. This security extends beyond the link and includes the security of Data when stored on the Device or further distributed to other Devices, Data Services and Apps.

Each Entity within the NICE System has a Unique ID, a Private Key used for decrypting data sent to the Entity and a Private Key for signing objects generated by the Entity. These Entities include Devices, Data Services and Apps. The NICE Account Service further configures these Entities to have a Private Key and Certificate to enable them to participate in the Privacy Management Service.

A Node within a Device may be configured to generate SceneData and SceneMarks. As part of the SceneMode configuration for the Node, an Output that is outputting either SceneMarks and SceneData shall be configured to either encrypt or not encrypt the SceneMarks or SceneData. If the Data is to be encrypted, the SceneMode configuration shall include a SceneEncryptionKeyID that references the SceneEncryptionKey that shall be used to encrypt the Data.

The SceneEncryptionKey shall be distributed in a Privacy Object that is provided to the Device. The Privacy Object is encrypted using the Public Key for the Device and signed by the Private Key for the NICE Account Service. The Privacy Object may also contain rules for how the data shall be handled. This may include the removal of sensitive data from the data. For example faces within an image may be blanked out.

Other Entities that may process data shall also be provided with a Privacy Object that contains the same SceneEncryptionKey as was used for the encryption of the Data. These Privacy Objects may also contain rules for handling the Data once it has been decrypted. The Privacy Object is always addressed to a single Entity and is always encrypted using the Public Key for the Entity. A Privacy Object can only be decrypted by the Entity for which it is intended.

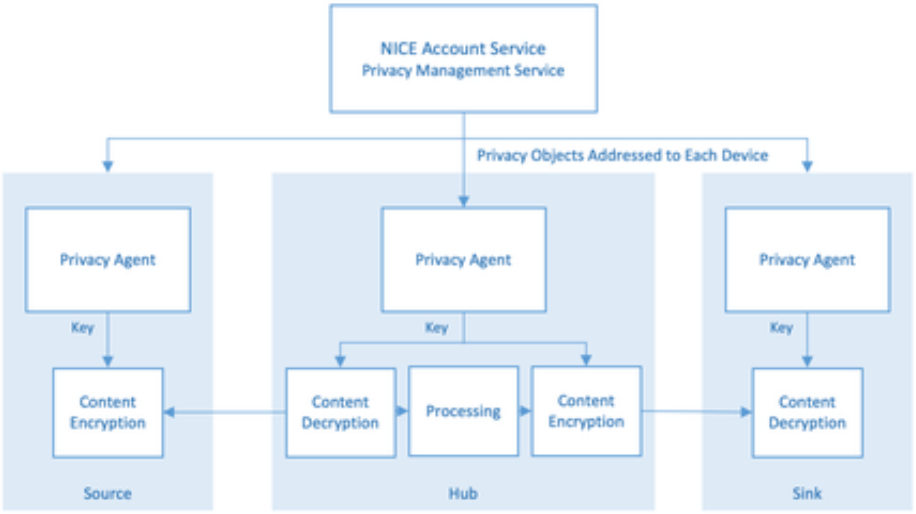

Figure 3. Processing of Protected Data Objects between Devices

▲ Top

6.2. Roles

6.2.1. Privacy Management Service

The Privacy Management Service enables the NICE Account Service to manage access to data. The End User controls which applications may access SceneData and SceneMarks. It provides fine grain control over the window of access and which data items may be accessed.

The Privacy Management Service manages the distribution of SceneEncryptionKeys used to either decrypt or encrypt SceneData. It is also responsible for defining the rules that are applied to access the SceneData. This service may be cloud based or for simple systems may be implemented in an appliance or in an individual device.

SceneEncryptionKeys are distributed by the Privacy Management Service through Privacy Objects.

▲ Top

6.2.2. Privacy Agent

The Privacy Agent is responsible for processing the Privacy Object, managing the decryption or encryption and enforcing rules that are contained in the Privacy Object.

6.2.3. Device

The Device shall implement the following functions:

1. Key Management and Storage.
2. Processing of Management, DeviceControl and Privacy Objects.
3. Processing of AccessTokens.

6.2.4. Node

A Device may contain one or more Nodes. A Node within a Device generates SceneData and SceneMark. The Device manages the SceneEncryption keys that are used by the Nodes in it.

▲ Top

6.2.5. App Developer

The App Developer develops the App and may operate servers that distribute data to Instances of the App executing on mobile or other devices.

If the App Developer shall access Data that is secured with by the Privacy Management Service, the App Developer shall have a AccountID and have a Private Encryption Key and Private Signing Key. The App Developer shall conform to the requirements defined in the Privacy Object and the conditions for the operation of a NICE App including the securing of data.

6.2.6. App Instance

An App Instance generates a Private Public Key pair. The NICE AS generates a X.509 certificate for the App. Each instance of application has its own public private key pair.

▲ Top

6.3. Privacy Object

The Privacy Object provides the Device with SceneEncryption Keys that are used to decrypt or encrypt data.

7. Trusted Time

Trusted time shall be used within the NICE System for the following purposes:

1. Trusted time stamping of SceneData and SceneMarks.
2. Prevention of replay attacks of sensitive messages.

The device shall maintain a Trusted Time clock. On power up the Device shall make a request for a Trusted Time Stamp from the NICE Account Service. In case the device has not been assigned to a NICE Account Service the request shall be made to the NICE License Authority. The NICE Account Service(s) and the NICELA shall have synchronized clocks.

The Trusted Time Protocol includes a challenge response system where the Device powers up, generates a random challenge, initiates a clock, transmits the challenge as a signed and encrypted JSON

object to the NICE Trusted Time Service which is part of the NICELA or NICEAS. The NICE Trusted Time Service returns an encrypted and signed response with original challenge and the NICE Trusted Time Service's time.

If this response passes verification, the challenge matches and has been received within 5 seconds of the original transmission, the time stamp shall be used to set the clock value for the Device.

- The Device may repeat this sequence if there is a requirement for greater certainty of the time on the device.
- The clock within the Device shall operate in a processor or zone which is not accessible by an Application on the Application processor.
- This protocol shall be performed at least on power up.
- The output of the clock shall be directly accessible to applications executing in the secure environment, without requiring processing by the Application Processor.

The following JSON objects shall be used in this protocol. The encryption and authentication of each object is done in the following way:

1. The TrustedTimeRequest Object is signed with the Private Key of the Device. The contents of the message are encrypted with the Public Key of the NICE Trusted Time Service.
2. The TrustedTimeResponse Object is signed with the Private Key of the NICE Secure Trusted Service and encrypted with the Public Key of the Device.

8. JSON Signing and Encryption

Any JSON object defined for Privacy Management or for Network Security shall be encrypted and signed in accordance with the JOSE specifications defined in RFC 7516 and RFC 7515. These objects are constructed in a prescribed format called Common Message Format (CMF). CMF Object used in NICE API call is called as CMFRequest Object. CMF Object used in the response of the API is called as CMFResponse Object. See the NICE Network Protocol Specification for details on CMF.

JWE Compact Serialization and **JWS Compact Serialization** respectively defined in RFC 7516 and RFC 7515 shall be used when encrypting and signing the Object.

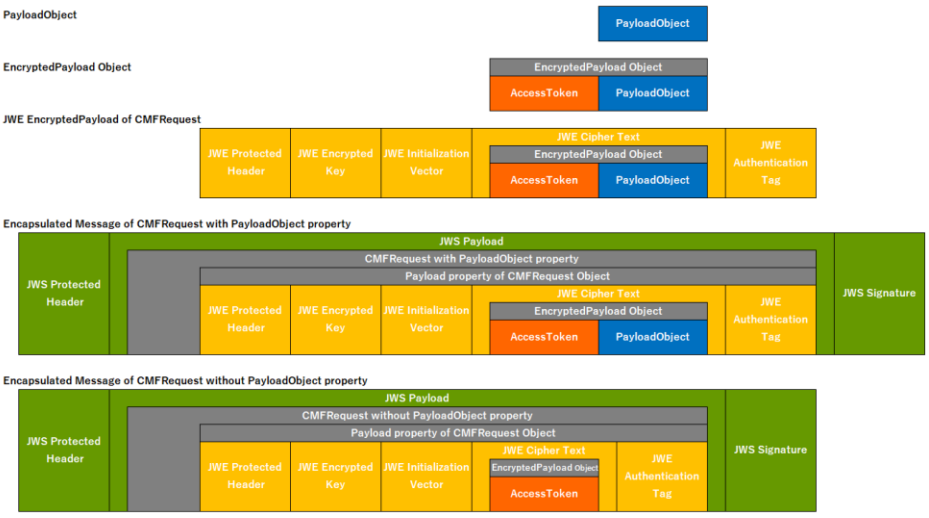

Figure 4. Encryption & Signing of JSON Objects in CMFRequest

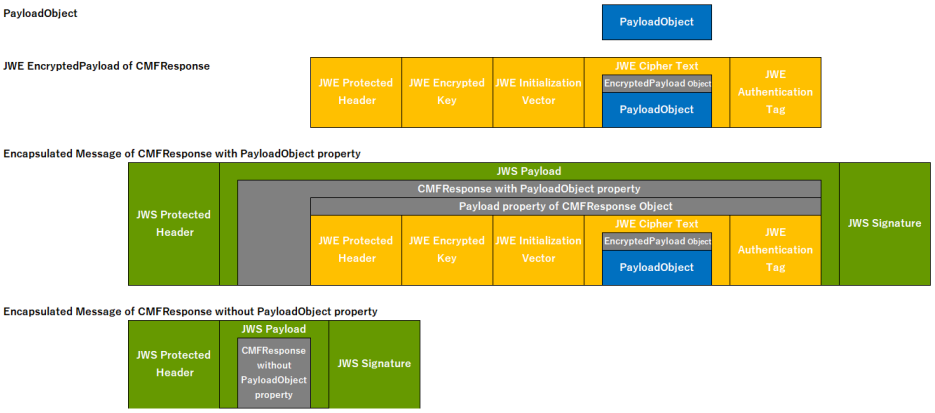

Figure 5. Encryption & Signing of JSON Objects in CMFResponse

The following describes the specific methods and algorithms that should be applied in the context of the NICE specifications.

- The handling of signing of JSON objects shall be implemented according to RFC 7515.
- RFC 7515 provides the linkage between the signed JSON object and the relevant X.509 certificates.
- The public keys for the actors in the NICE system shall comply with the X.509 structure for digital certificates.

There are two categories of JSON objects:

1. Individually addressed objects that can only be processed by a specific Entity (Device, App or DataService). All Privacy, Management and Control Objects are in this category. These are encrypted with the public key of the receiving Entity.
2. Objects that can be accessed by multiple entities. These are encrypted using SceneEncryptionKeys that are distributed via Privacy Objects in JSON Web Key format defined in RFC 7517.

Prescribed parameters are carried in the JOSE header.

The following parameters defined in RFC 7516 are compulsory for JWE EncryptedPayload:

- "alg" (Algorithm) Header Parameter
- "enc" (Encryption Algorithm) Header Parameter
- "kid" (Key ID) Header Parameter

The following parameter defined in RFC 7518 is additionally required for "EC" Key Type JWE EncryptedPayload:

- "epk" (Ephemeral Public Key) Header Parameter

The "kid" references the ID of the Entity to which the object is addressed.

The following is the high level structure of the JWE EncryptedPayload.

- Certain fields in the security objects must be accessible when the JSON object is encrypted.
- These are carried in the protected header section of the JSON object (according to the JWE specification, these are covered by the authentication tag of the object but are not encrypted).
- These fields are indicated in the JSON schemas for the JSON security objects

Figure 6. Structure of JWE EncryptedPayload

For the individually addressed objects the following settings shall be used:

- "alg":"ECDH-ES+A256KW" or “alg”:“RSA-OAEP-256”
- "enc":"A256GCM"

or alternatively the following setting shall be used (if required by regulation):

- "alg":"SM2+SM4"
- "enc":"SM4"

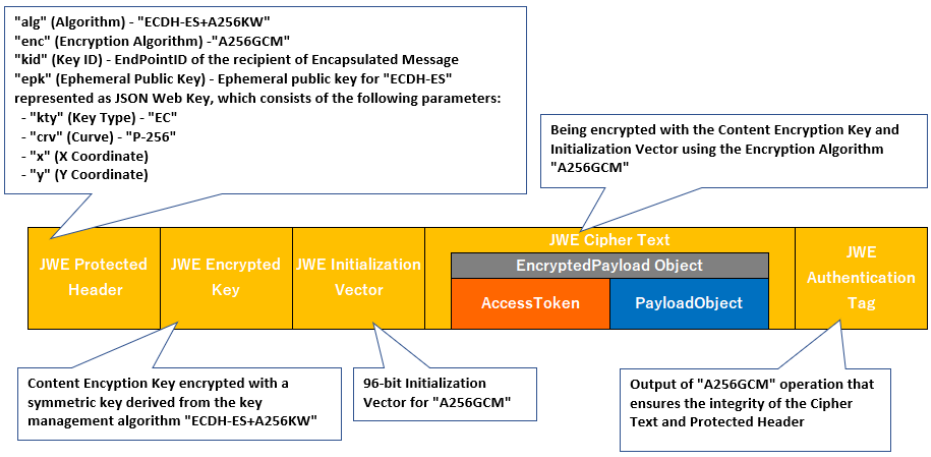

Figure 7. Encryption of individually addressed Objects using "EC" Key Type

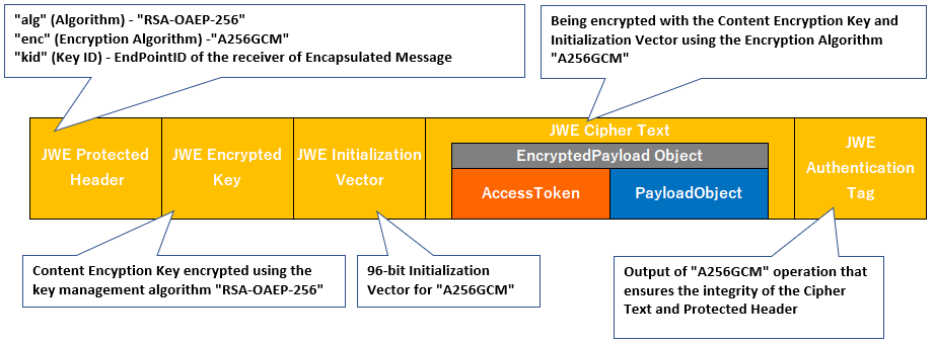

Figure 8. Encryption of individually addressed Objects using "RSA" Key Type

The choice of algorithm is determined by the NICE License Authority.

The specific definitions of the ECDH-ES, A256KW, RSA-OAEP-256 and A256GCM algorithms can be found in RFC 7518 . The definitions of the SM2 and SM4 algorithms are available from CNNIC.

Any message which indicates a different algorithm from the above shall be rejected as an invalid message.

All Management, Control and Privacy Objects shall be signed by the party issuing these objects.

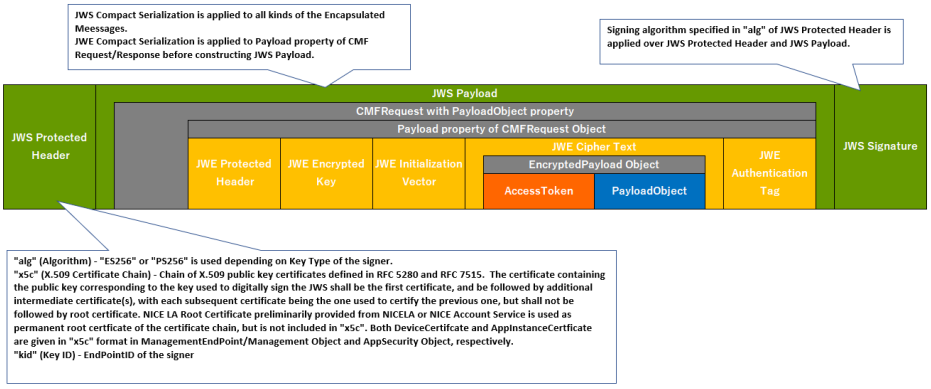

Figure 9. Signing of Privacy, Management and Control Objects

The following parameters defined in RFC 7515 are compulsory for JSON objects:

- "kid" (Key ID) Header Parameter
- "alg" (Algorithm) Header Parameter - "ES256" or "PS256" shall be used
- "x5c" (X.509 Certificate Chain) Header Parameter - given in certificate chain up to root certificate, but excluding NICE LA Root Certificate

▲ Top

9. Management API

9.1. GetPrivacyObject

Function

The Device or App shall be capable of requesting the Privacy Object configuration form the NICE Account Server. The Device or App may request the Privacy Object by referencing the SceneEncryptionKeyID carried in the Privacy Object or the PrivacyObjectID for the PrivacyObject.

In case only the SceneEncryptionKey is carried in the request the Privacy Management System shall determine whether the entity requesting the Privacy Object is entitled to receive the key and will either provide an existing Privacy Object or generate a new Privacy Object to provide the key to the entity.

Protocol(s) Used to Make Calls

WebAPI

Direction

Caller	DEVICE, APP
Callee	NICEAS

Request

PrivacyObjectID

Acknowledgement Parameters

Privacy Object

10. Data Objects

10.1. Privacy Object

The following code block is the **JSON schema** for the **Privacy Object**. The SceneEncryptionKey that is carried in the object is the actual SceneEncryptionKey value that is used to encrypt or decrypt the encrypted SceneData or SceneMarks. The SceneEncryptionKeyID provided in the Privacy Object shall correspond to the SceneEncryptionKeyID that is defined in the SceneMode for the encryption of SceneMarks or SceneData. The Device shall use this SceneEncryptionKeyID when requesting the Privacy Object corresponding to encrypted SceneData or SceneMarks.

The Privacy Object contains restrictions on processing that shall be enforced when either the SceneData or SceneMarks are decrypted.

Where a Device is generating SceneMarks or SceneData, the SceneMode shall define whether encryption is required and the SceneEncryptionKeyID that shall be used for encryption. The Device shall ensure that it either has a valid Privacy Object stored on the device which corresponds to the SceneEncryptionKeyID or shall request the Privacy Object from the NICEAS Privacy Server using the SceneEncryptionKeyID as a reference. The Device shall use the SceneEncryptionKey that corresponds to the SceneEncryptionKeyID that is delivered in the SceneEncryptionKey field in the Privacy Object.

The entities which consume SceneData and SceneMarks shall use the SceneEncryptionKey in the Privacy Object to decrypt encrypted SceneData and SceneMarks.

A Privacy Object is a **JSON object** that contains the following information:

1. EndPointID: Identity of the Entity that is being enabled to access the data.
2. PrivacyObjectID: Unique ID of the Privacy Object issued by Privacy Management Service.
3. StartDateTime: Date and Time from when object may be processed.
4. EndDateTime: Time from which the Privacy Object is no longer valid and may longer be processed.
5. SceneEncryptionKey. Object containing SceneEncryptionKey and SceneEncryptionKeyID that is used to decrypt or decrypt Data.

The following cryptographic operations shall be performed on all Privacy Objects.

1. A signature that is generated using the Private Key of NICE Account Service. The Privacy Object shall be rejected if this signature check fails in any way.
2. The payload of the Privacy Object is encrypted using the public key of the device, application or service that is consuming the Privacy Object. This encryption is described in the section covering JSON encryption and signing.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "Privacy",
    "description": "Used to distribute SceneEncryptionKeys and rules for the usage of
SceneEncryptionKeys.",
    "properties": {
        "Version": {
            "type": "string",
```

```
            "enum": [
                "1.0"
            ]
        },
        "EndPointID": {
            "type": "string",
            "description": "Identity of client that is being allowed access to the
data"
        },
        "PrivacyObjectID": {
            "type": "string",
            "description": "Unique ID of Privacy Object issued by Privacy Management
Service"
        },
        "StartDateTime": {
            "type": "string",
            "description": "Date and Time from when object may be processed."
        },
        "EndDateTime": {
            "type": "string",
            "description": "Time from which the object is no longer valid and may
longer be processed."
        },
        "SceneEncryptionKey": {
            "$ref": "Definitions.json#/definitions/SceneEncryptionKey"
        }
    },
    "required": [
        "PrivacyObjectID",
        "StartDateTime",
        "EndDateTime",
        "EndPointID",
        "Version"
    ]
}
```

10.2. PrivacyObjectID

This object is used to indicate which Privacy Object is being requested. The Privacy Object may be identified by the specific PrivacyObjectID for that Privacy Object or the SceneEncryptionKey which is contained in the Privacy Object.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "PrivacyObjectID",
    "description": "Privacy Object ID that is being requested. The PrivacyObject may
be referenced by the PrivacyObjectID or the SceneEncryptionKey.",
    "properties": {
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "PrivacyObjectID": {
            "type": "string",
```

```
            "description": "The Privacy Object may be requested by referencing the
PrivacyObjectID or by the SceneEncryptionKeyID"
        },
        "SceneEncryptionKeyID": {
            "type": "string",
            "description": "The Privacy Object may be requested by referencing the
PrivacyObjectID or by the SceneEncryptionKeyID"
        }
    },
    "required": [
        "Version"
    ]
}
```