Privacy and Security Specification

Version 0.9

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

Revision History

Version	Date	Comments
0.9rc1	13 Nov 2018	First draft
0.9rc2	25 Feb 2019	Second draft
0.9	25 Mar 2019	Final draft

Contributors

Name	**Company**
Andrew Wajs	Scenera
Aviram Cohen	Scenera
Munehiro Shimomura	Sony
Hironori Miyoshi	Sony
Wendy Tin	Wistron

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 a DeviceID, a Private Signing Key used for signing objects generated by the Device and the Private Encryption Key, the key 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.

NICE 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.

Asset	Attack	Attacker Motivation	Counter Measure
Account	Hijack of Account	Ransom	Managed Credentials
	Full access to devices and data	Invasion of Privacy	Revocable OAuth 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 with mutual TLS
	Rogue Computing	Disable Physical Security	Firmware Signing
	Denial of service, Cryptocurrency Mining	Fraud	Implementation Requirements
		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:

1. **Device:**
    1. Device Credential Management
    2. Device Management
        1. Network Security
        2. Trusted Time
        3. Code Signing
2. **User Account:**
    1. Access Control
3. **Data:**
    1. Privacy Management Service
4. **App Provider:**
    1. Application Key
    2. Credential Management
    3. Privacy Management Service
5. **App Instance:**
    1. Application Key
    2. Credential Management
    3. Privacy Management Service

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. Access Control, Privacy Management and Network Security all conform to this trust model.

The highest authority in the NICE System is the NICE Licensing Authority (NICE LA). 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 NICE LA:
    - **certificate and private signing key** used for signing objects generated by the Device.

    - **certificate and private encryption key** used for encrypting and decrypting data sent to the Device.
- Each **NICE Account Service** in the NICE Ecosystem is provided **certificates** for validating the private key used to validate objects issued by the NICE Account Service and the key used to encrypt data sent to the NICE Account Service. These are provided by the NICE Licensing Authority.

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 NICE AS and the second is used for validating objects sent from the NICE AS. The NICE Account Service creates its own trusted ecosystem of NICE Apps and Services. 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 as well as User information.

4.2.2. 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.
- It should also consider the usage of multi-factor authentication to enable recovery in case of the loss of AccountID and password.
- These may include messaging using mobile telephone number, email or may involve the use of other security tokens and forms of identification to validate the transfer of the device.

4.2.3. 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 extended Scene Data by using Privacy Objects to control access to it.

4.2.4. Network Security Service

It controls the configuration of the Network Security Function embedded in the Device, including the encryption keys used for any communication with the Device.

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 NICE LA.
- Performs an audit on its implementation to ensure that it conforms to the Security Implementation Requirements defined by the NICE LA.

The Device has a unique ID, Private Encryption Key and Private Signing Key embedded by the Device Manufacturer.

- These are provided by the NICE LA.
- 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 NICE LA of the devices that it is selling into the market.
- If this information is not provided the NICE LA will not enable activation of the device onto a NICE Account Service.
- The NICE device Seller may be responsible for issuing updates to the Device firmware signed under the firmware signing key.

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

The DeviceSeller object shall be provided by the Device Seller to the NICE LA to register the device with the NICE LA. When this has been received for the device, the NICE LA shall publish the device certificate on a NICE LA server. This certificate shall not be downloaded to the device. Applications and services that require this certificate to communicate with the device, obtain the certificate by querying the NICE LA server for the certificate corresponding to a particular DeviceID.

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 Keys to enable it to participate in the

Privacy Management Service (One Private Key and certificate is used to decrypt data sent to the App and the other used to sign objects sent from the App). 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 and a key pair for validating objects sent from the AppInstance.

4.6. Apps

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 and private key by the NICE Account Service.

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 NICE LA to the device manufacture to enable the manufacturer to embed credentials into the device during manufacture.
- The **Management Object** is issued by the NICE LA and binds a device to a NICE Account Service.
- The **Control 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 **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 the OAuth2 protocol 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 **FirmwareUpdate Object** enables the validation of a firmware update.
- The **DeviceSeller Object** feeds back to the NICE LA 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 Management Object is issued by the NICE License Authority, and binds a device to a NICE Account Service.

These objects are always encrypted using the Public Key of the device to which they are addressed.

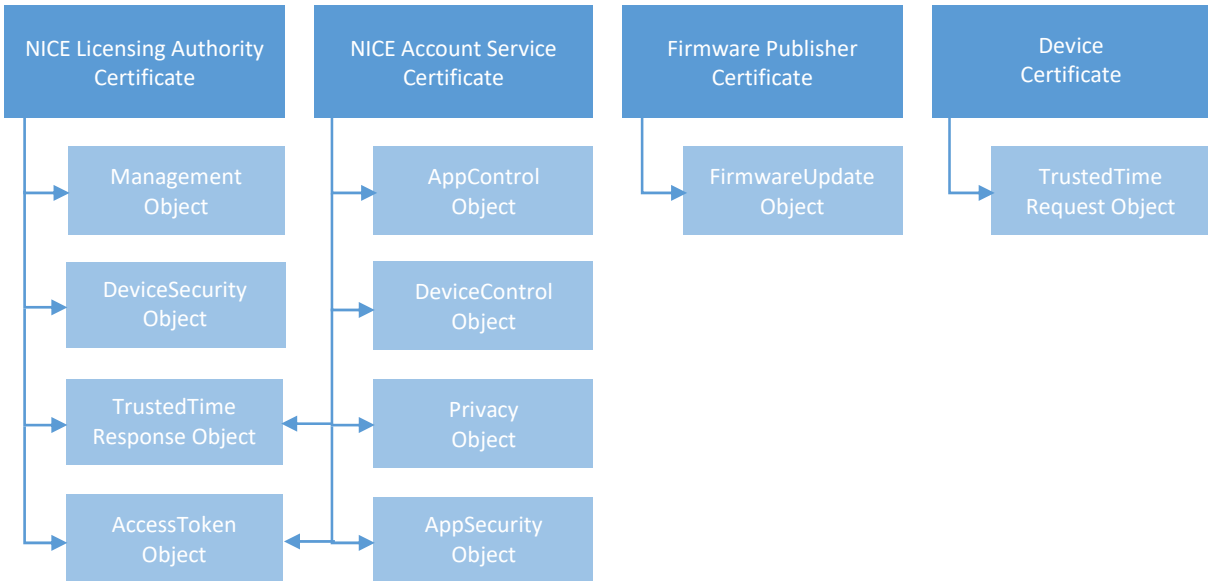

Figure 1. Certificates used to validate Security Objects

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

- The **Device Security Object** is always signed by the NICE Licence Authority.
- The other Objects are always signed by the NICE Account Service.

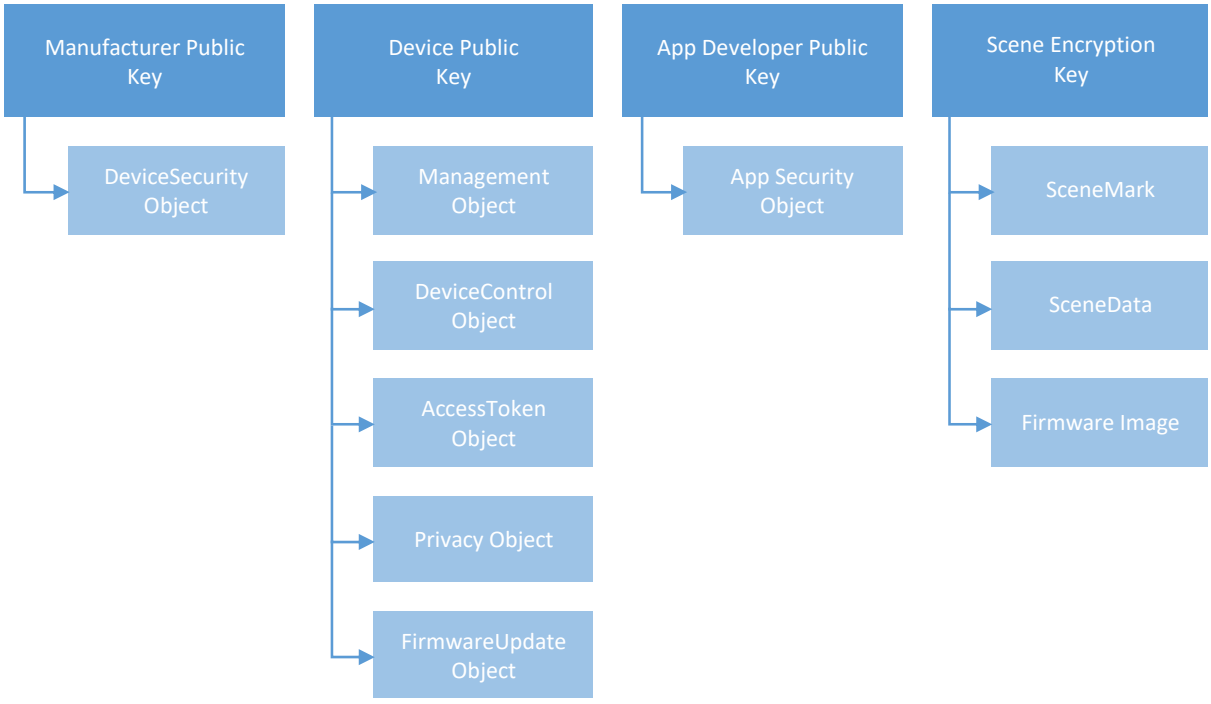

Figure 2. Keys Used to Encrypt Objects.

6. Privacy Management

6.1. Overview

Devices generate Data in the form of SceneMarks, SceneData and Live Streams. 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 or DTLS. 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 SceneModes 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.

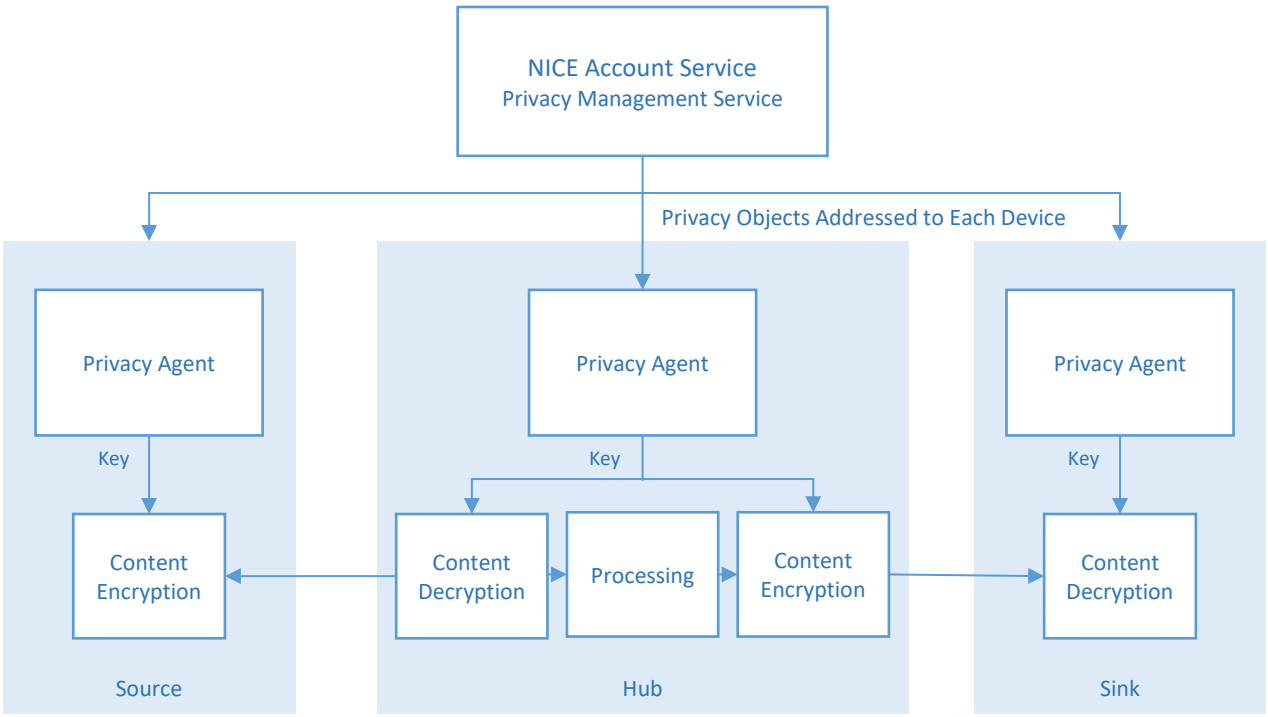

Figure 3. Processing of Protected Data Objects between Devices

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 Private Management Service performs the following functions:

- Control over who may or may not access data.
- Further control over the SceneEncryptionKeys is applied by attaching usage rules to the SceneEncryptionKeys that control how the data may be accessed, how it may be processed and how the result of the processing may be distributed.
- These rules include a window of time during which data may be accessed, types of processing that may be applied to data, ability to forward the results of analysis of the data etc.

The Privacy Management Service may define standard policies for how SceneData is managed and encrypted. It may also define standard policies for how different NICE service providers may access and process SceneData.

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 which include the following fields:

1. EndPointID of the receiving Device, App or Service.
2. SceneEncryptionKeys for encrypting or decrypting SceneData and SceneMarks.
3. Usage rules for SceneEncryptionKeys.
4. Authentication field.

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. The Privacy Agent shall be implemented in a Trusted Execution Environment.

6.2.3. Device

A Device may contain one or more Nodes. The Device shall have a Security Application which implements the following functions:

1. Key Management and Storage.
2. Processing of Management, Control and Privacy Objects.
3. Processing of Firmware Object.
4. Processing of AccessTokens.

The Security Application shall be isolated from other applications in the device. The requirements for this isolation are defined in the Device Implementation Guide.

6.2.4. Node

A Node within a Device generates either SceneData or SceneMark. The Node is always associated with a Node which is responsible for managing the SceneEncryptionKeys that the Node shall use to process the data that the Node is generating or consuming.

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 NICE Account Service shall issue a Private Public Key pairs and X.509 certificate to the App Developer. The Privacy Management Service utilizes this public key to encrypt Privacy Object to enable the App Developer to access SceneData and SceneMarks. 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 may be issued a Private Public Key pair and X.509 certificate by the NICE Account Service. The Privacy Management Service utilizes this public key to encrypt Privacy Object to enable the NICE Service Provider to access SceneData. Each instance of application has its own public private key pairs.

6.3. Data Encryption

A Node acts as a source or a sink for SceneData and SceneMarks. The SceneMode for the Node defines the Input and Output settings for that Node. These settings include how the Data shall be encrypted or decrypted. If the SceneMode specifies that an Input or Output shall be encrypted or decrypted, the SceneMode shall also specify a SceneEncryptionKeyID that is used to encrypt or decrypt the data. The SceneEncryptionKey that is used to encrypt the data is delivered to the Device in a Privacy Object. The SceneEncryptionKey is extracted from the Privacy Object and is used to encrypt the data. The format of the encryption is determined by the encoding format for the data that is being sent over the input or output. Where the data is a SceneMark the JSON Object Signing and Encryption format is to be used. In the case of SceneData the encryption format is dependent on the container format for the SceneData.

The Privacy Object containing the SceneEncryptionKey may be fetched by the Device when the SceneMode is set or pushed to the Device ahead of the SceneMode configuration. Privacy Objects may be cached in the device for future use.

Data may be encapsulated in either the ISO Base Media Format or as JSON documents.

6.3.1. JSON Data Encryption

SceneData in JSON format is encrypted according to the section below which describes JSON encryption within NICE.

Each JWE encrypted object shall have the SceneEncryptionKeyID for the SceneEncryptionKey that enables the Privacy Agent in the Entity to reference a Privacy Object addressed to the Device/Service/App that enables the decryption of the SceneData or SceneMarks.

6.3.2. Video/Audio Encryption

SceneData shall be encapsulated in the **ISO Base Media File Format**.

The encryption of data is performed according to the mechanism defined in the ISO Base Media File Format specification.

The SceneEncryptionKey carried in the Privacy Object shall be used to encrypt and decrypt the Video/Audio.

6.3.3. Interoperability with Existing DRM Systems

The ISO Base Media File Format is compatible with the DRM and video playback in most mobile, PC and TVs. The DRM used in these devices can be used to control the playback of encrypted video.

For this to occur, the DRM servers which can address these devices shall be provided the SceneEncryptionKey to be included into the DRM rights objects supported by these systems (these include Apple FairPlay, Widevine DRM and Playready DRM).

ISO Base Media File Format encryption and authentication is described in the MPEG document "Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 4: Segment encryption and authentication". The DASH Interoperability Forum has defined how content packagers that implement this encryption can exchange SceneEncryptionKeys and other DRM information. The Privacy Management Service shall use these interfaces to request a compliant DRM implementation to generate Rights Objects that can be consumed by standard implementations of DRM in these devices. This is described in the document: "DASH-IF Implementation Guidelines: Content Protection Information Exchange Format (CPIX)".

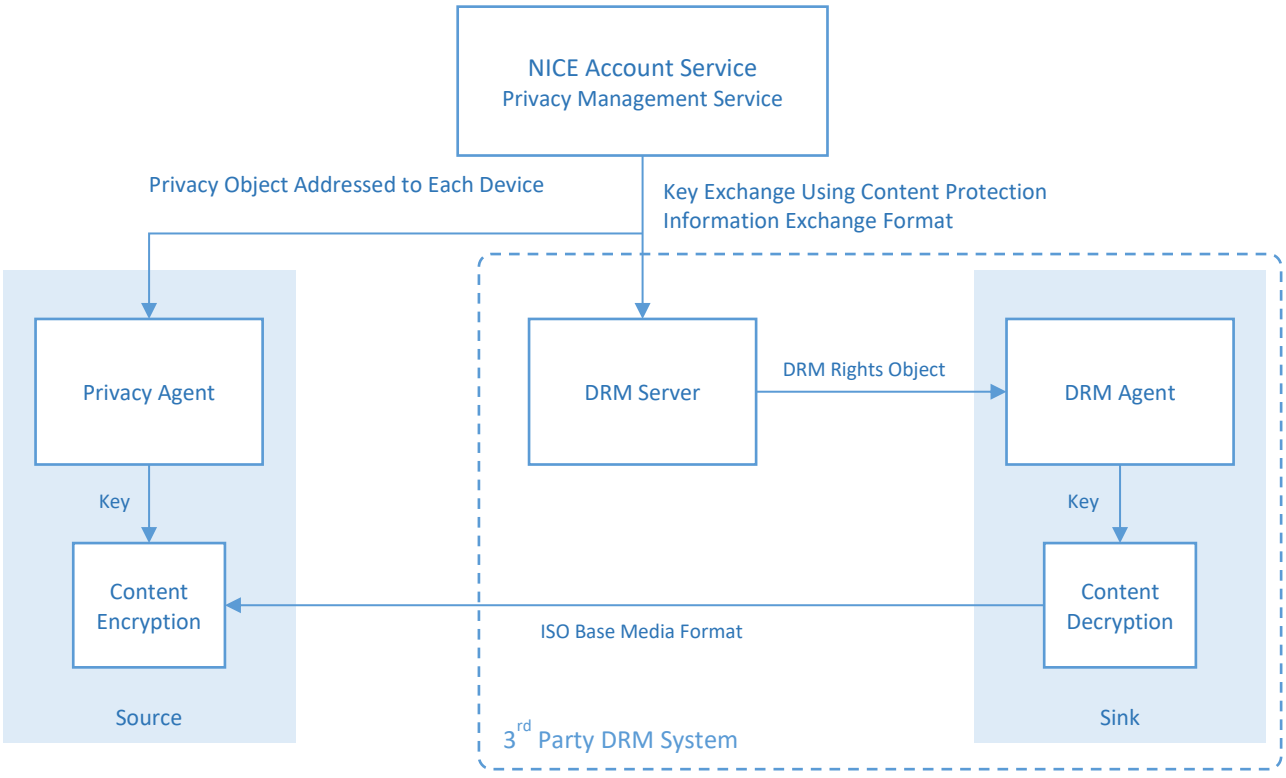

Figure 4. Interoperability with 3rd Party DRM Systems

6.4. Privacy Object

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. StorageRule: Rules governing storage of Data.
6. ExportRule: Rules concerning forwarding of Data.
7. MaskedItems: Items to be obscured or removed from output Data. For example if = Face then faces shall be blurred or blanked out in the images or video in the Data.
8. AnalysisRules: Restrictions on analysis that may be performed on the data. For example if set to FaceOnly then only Face Detection, Recognition or Characteristic may be performed.
9. SceneEncryptionKey. Object containing SceneEncryptionKey and SceneEncryptionKeyID that is used to decrypt or decrypt Data.
10. Authentication: If true then the device or node shall use its private key to authenticate SceneData and SceneMarks.

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.

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 NICE LA 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 NICE LA or NICE AS. The NICE Trusted Time Service returns an encrypted and signed response which 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 Time Request 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 Time Stamp Object is signed with the Private Key of the NICE Secure Trusted Service and encrypted with the Public Key of the Device.

Each device may shall include their X.509 certificate in the request or response.

7.1. TrustedTime Request

The Device shall generate a TrustedTime Request Object and send this to the NICE Trusted Time Service.

The Device shall generate a random number which shall be used as Random Challenge. It shall be encrypted using the Public Key from either the NICE LA or NICE Account Service.

7.2. TrustedTime Response

The NICE Trusted Time Service shall return the Trusted Time Response Object with the Random Challenge correctly decrypted in the Object.

8. JSON Signing and Encryption

Any JSON object defined for Privacy Management or for Network Security shall be encrypted in accordance to the JOSE specifications for JSON encryption and authentication. **JWE Compact Serialization** defined in RFC 7516 shall be used.

Objects are encrypted according RFC 7516 and signed using RFC 7515.

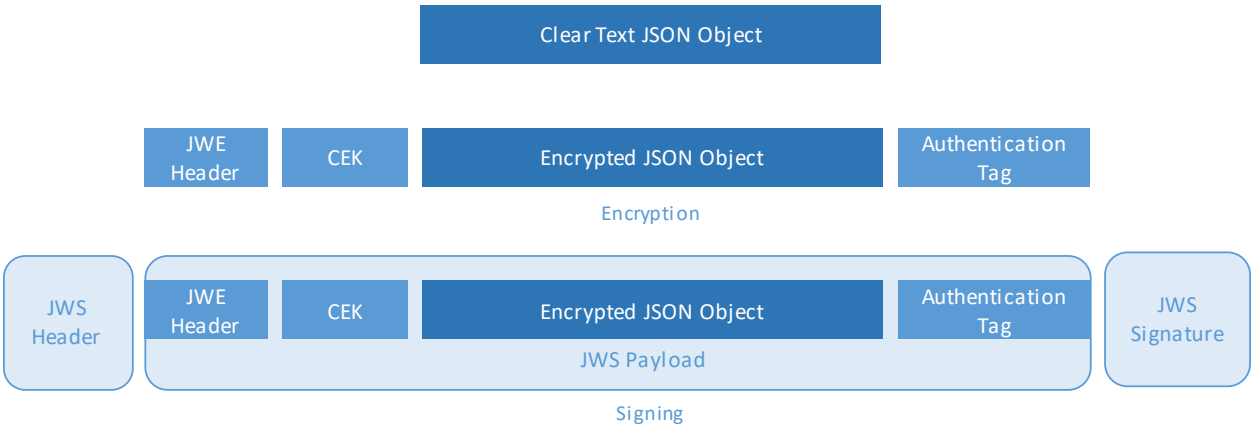

Figure 5. Encryption & Signing of JSON Objects

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

- The handling of signing of JSON objects shall be implemented according to RFC 7516.
- The public keys for the actors in the NICE system shall comply with the X.509 structure for digital certificates.
- RFC 7516 provides the linkage between the signed JSON object and the relevant X.509 certificates.

There are two categories of JSON objects:

1. Individually addressed objects that can only be processed by a specific entity (device, app or service). All Privacy, Management and Control Objects are in this category.
2. Objects that can be accessed by multiple entities. These are SceneData and SceneMark objects and media files. These are encrypted under SceneEncryptionKeys that are distributed via Privacy Objects.

The following parameters defined in RFC 7516 are compulsory for JSON objects:

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

For Individually addressed objects the "kid" references the ID of the Entity to which the object is addressed. For objects that can be accessed by multiple Entities, the "kid" shall be the SceneEncryptionKeyID for the SceneEncryptionKey used to encrypt the object.

For SceneData the "x5c" is not present as the encryption and signing of the JSON utilizes symmetric cipher.
The following is the high level structure of the signed message. The above parameters are carried in the JOSE header.

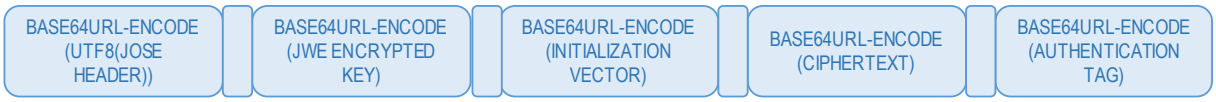

Figure 6. Structure of an Encrypted JSON Object

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

- "alg":"ECDH-ES+A256KW"
- "enc":"A256GCM"

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

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

For JSON objects carrying SceneData and SceneMarks the following settings shall be used:

- "alg":"A256KW"
- "enc":"A256GCM"

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

- "alg":"SM4"
- "enc":"SM4"

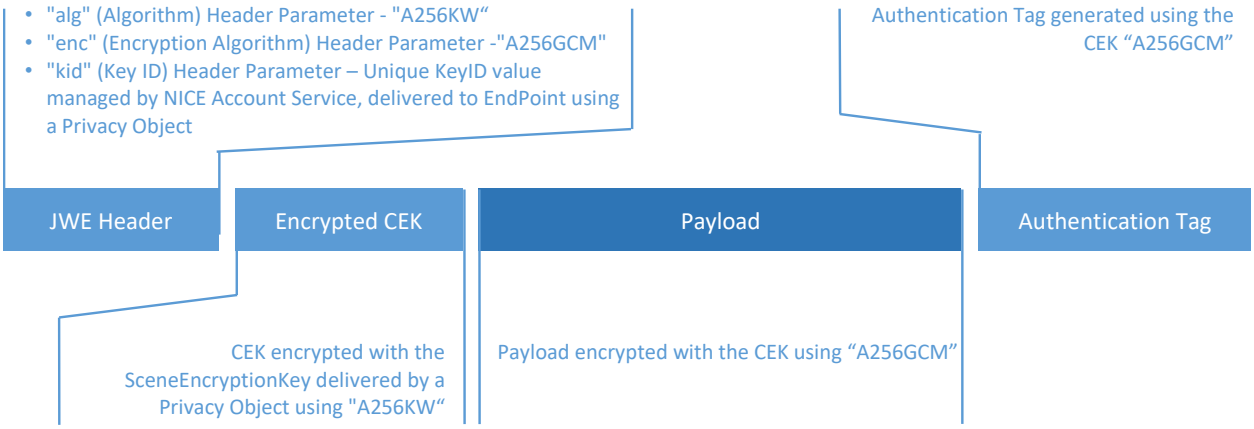

Figure 7. Encryption of SceneData and SceneMarks

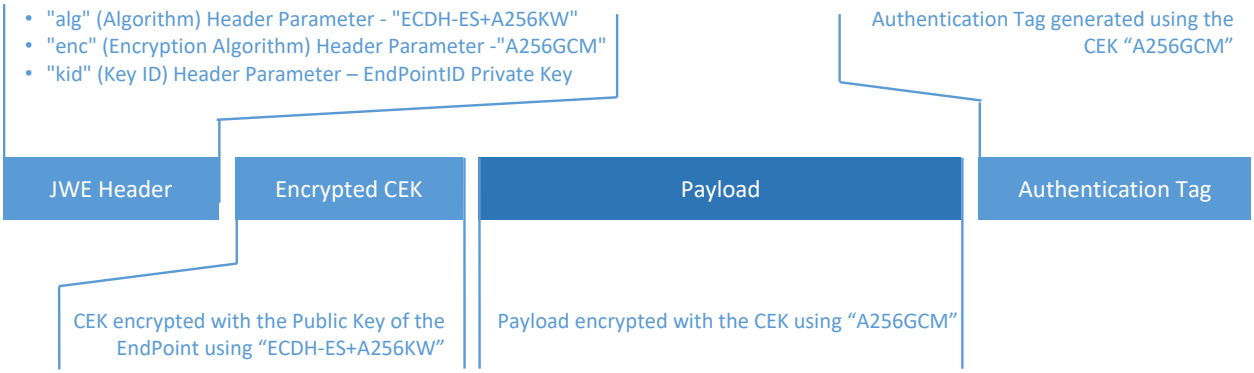

Figure 8. Encryption of Privacy, Management and Control Objects

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

The specific definitions of the ECDH-ES, A256KW 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.

The key used to wrap the SceneEncryptionKey is identified by the value defined by "kid".

- These SceneEncryptionKeys are carried in the Privacy Objects and the "kid" is a unique value in the context of a single Privacy Management Service.
- 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

All Management, Control and Privacy Objects shall be signed by the party issuing these objects. SceneData and SceneMark objects may be signed by the entity that generates the object. This is set by the Encryption field in the SceneMode object used to configure the SceneMode for a Node.

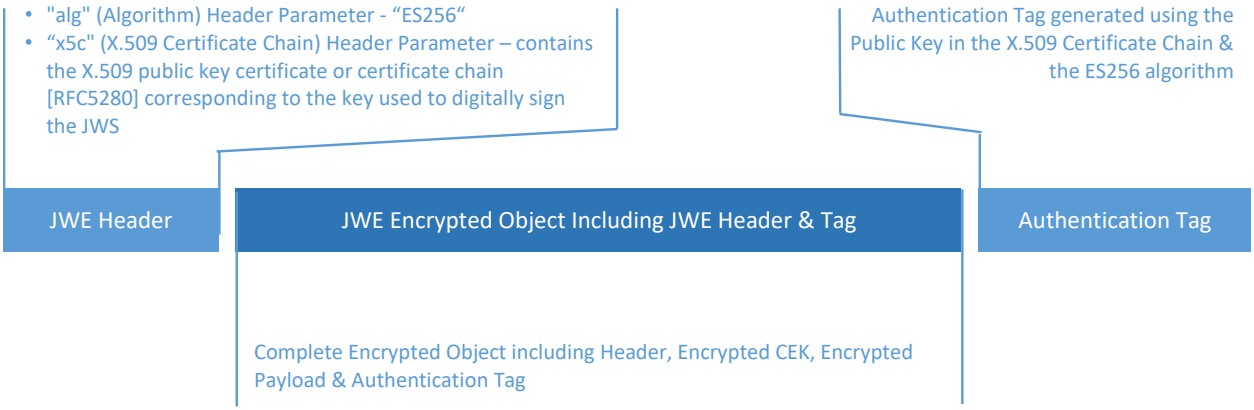

Figure 9. Signing of Privacy, Management, Control Objects, SceneData and SceneMarks

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

- "alg" (Algorithm) Header Parameter
- "x5c" (X.509 Certificate Chain) Header Parameter

9. Chaining SceneMarks and SceneData

A SceneMark shall be generated according to the configuration defined by the SceneMode. A Node may be configured by its SceneMode to receive SceneMarks from other Nodes and may add additional data fields to the SceneMark. A SceneMark may be constructed over time by different systems. The SceneMark may be referenced to years after the original SceneData was captured. For example, if a particular Scene has high importance and a new algorithm becomes available it may make sense to append the output of the new algorithm to the existing SceneMark for that scene.

It may also be important to be able to track which systems have added which data to the SceneMark and to ensure that no system has altered the SceneMark.

The Device capturing the SceneData may sign the SceneData to prove that no subsequent system has altered the original SceneData. The SceneMode for the Node defines whether the SceneData shall be signed and encrypted.

The device may have a motion detection capability. On detection of motion, the device creates a SceneMark which contains the time and a reference to the video file containing what triggered the motion

detection. This video itself may be signed and the SceneMark generated by the device is signed with the device's private key. A different system may then process the video and SceneMark to recognize a face in the video. This system may generate a new SceneMark with the identity of the face to the SceneMark. This new SceneMark may reference the old SceneMark as a "ChildSceneMark" and store a hash of the ChildSceneMark's hash. Multiple ChildSceneMarks may be referenced in the SceneMark. In this case the Hash stored in the SceneMark is the Hash for the authentication tags for the Child SceneMarks in the order that the ChildSceneMarks are listed in the SceneMark.

A party wishing to verify the origin of the data in the SceneMarks is able to verify that the first Device generated the first SceneMark and the second Device generated the second SceneMark.

RFC 7515 defines a signing of a JSON data structure.

- Using this method a sequence of signatures can be created which enable first party to sign a JSON structure.
- A second party can then include the signature generated by the first party into a JSON structure that incorporates additional SceneData to the first set of SceneData in the SceneMark.

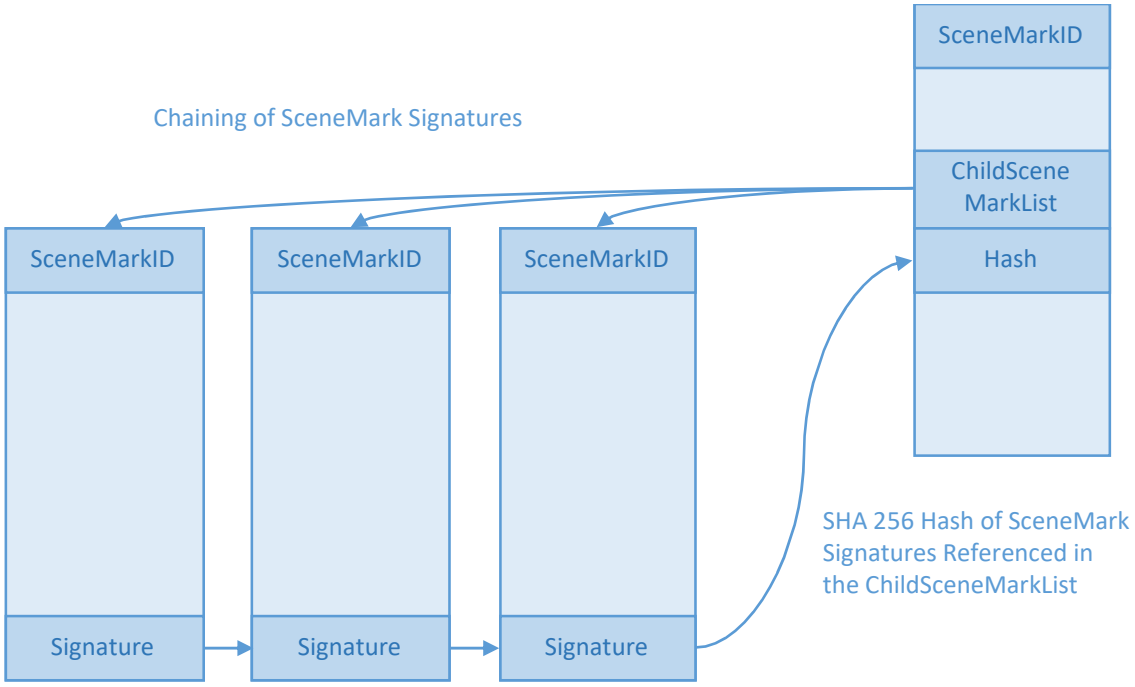

Figure 10 Chaining of SceneMarks

10. Management API

10.1. SetPrivacyObject

Function

The App and Device must have the API to accept the Privacy object configuration.

The "SetPrivacyObject" sets the Privacy Object in either the Device or App. This API overwrites the settings previously configured with this API.

Protocol(s) Used to Make Calls

MQTT

WebAPI

Direction

Caller	NICE Account Service
Callee	App, Device

Request

Privacy Object.

Acknowledgement Parameters

Empty.

10.2. GetPrivacyObject

Function

The Device or App shall be capable of requesting the Privacy Object configuration form the NICE Account Server.

Protocol(s) Used to Make Calls

MQTT

WebAPI

Direction

Caller	Device, App
Callee	NICE Account Service

Request

Privacy Object.

Acknowledgement Parameters

Empty.

10.3. DeletePrivacyObject

Function

The App and Device must have the API to accept the Privacy Object deletion.

The "DeletePrivacyObject" shall delete the Privacy Object corresponding to the PrivacyObjectID in either the Device or App if the Object is present in the Entity.

Protocol(s) Used to Make Calls

MQTT

WebAPI

Direction

Caller	NICE Account Service
Callee	App, Device

Request

PrivacyObjectID

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "PrivacyObjectID",
    "description": "ID for Privacy Object that is referenced.",
    "additionalProperties": {
        "not": {}
    },
    "properties": {
        "PrivacyObjectID": {
            "type": "string"
        }
    }
}
```

Acknowledgement Parameters

Empty.

11. Data Objects

11.1. 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.

In case of export or storage of SceneData or SceneMarks based on the decrypted SceneMark or SceneData, the Privacy Object defines whether these shall be encrypted or not and with which SceneEncryptionKey the output shall be encrypted. These are defined in the Storage and Export structures within the Privacy Object.

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 NICE AS 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.

Privacy Object

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "title": "Privacy",
    "description": "Used to distribution of SceneEncryptionKeys and rules for the
usage of SceneEncryptionKeys.",
    "additionalProperties": false,
    "properties": {
        "EndPointID": {
            "type": "number",
            "description": "Identity of client that is being enabled to access the
data"
        },
        "PrivacyObjectID": {
            "type": "number",
            "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."
        },
        "UsageCount": {
            "type": "number",
            "description": "Number of times the SceneEncryptionKey may be used to
decrypt SceneMarks or SceneData. If this field is present the Privacy Agent shall
track the usage of the SceneEncryptionKey carried in this object and ensure that it is
not used more than this number of times. "
        },
        "StorageRule": {
            "type": "object",
            "description": "Rules governing storage of Data",
            "properties": {
                "StorageAllowed": {
                    "type": "boolean",
                    "description": "If TRUE Data may be stored."
                },
                "EnforceEncryption": {
                    "type": "boolean",
                    "description": "Encryption Required to Store Data"
                },
                "SceneEncryptionKeyID": {
                    "type": "number",
                    "description": "ID of SceneEncryptionKey used to encrypt stored
Data"
                },
                "SceneEncryptionKey": {
                    "type": "string",
                    "description": "Encryption SceneEncryptionKey used to encrypt any
stored Data"
                }
            },
            "required": [
                "StorageAllowed",
                "EnforceEncryption"
            ]
```

```
        },
        "Export Rule": {
            "type": "object",
            "description": "Rules concerning forwarding of Scene Data",
            "properties": {
                "ExportAllowed": {
                    "type": "boolean",
                    "description": "If TRUE Data may be exported"
                },
                "EnforceEncryption": {
                    "type": "boolean",
                    "description": "If TRUE then Data must be encrypted using supplied
key."
                },
                "SceneEncryptionKeyID": {
                    "type": "number",
                    "description": "ID of key to be used to encrypt exported Data."
                },
                "SceneEncryptionKey": {
                    "type": "string",
                    "description": "Key to be used to encrypt exported Data."
                }
            },
            "required": [
                "ExportAllowed",
                "EnforceEncryption"
            ]
        },
        "MaskedItems": {
            "type": "array",
            "description": "Items to be obscured or removed from output Data. For
example if = Face then faces shall be blurred or blanked out in the images or video in
the SceneData ",
            "items": [
                {
                    "enum": [
                        "Face",
                        "Human",
                        "Animal",
                        "Vehicle",
                        "Label",
                        "Text/Logo/QRCode",
                        "Custom"
                    ]
                }
            ]
        },
        "AnalysisRules": {
            "type": "array",
            "description": "Restrictions on analysis that may be performed on the
data. For example if set to FaceOnly then only Face Detection, Recognition or
Characteristic may be performed. ",
            "items": [
                {
                    "enum": [
                        "NoAnalysisAllowed",
                        "FaceOnly",
                        "HumanOnly",
                        "VehicleOnly",
                        "AnimalOnly",
                        "LabelOnly",
                        "Text/Logo/QRCodeOnly",
                        "CustomOnly"
```

```
                    ]
                }
            ]
        },
        "SceneEncryption": {
            "type": "object",
            "description": "Object containing SceneEncryptionKey and
SceneEncryptionKey ID that is used to encrypt or decrypt data. In the case of a Device
receiving data this SceneEncryptionKey is used to decrypt the data, inc the case that
the Device is generating new SceneData or SceneMarks this SceneEncryptionKey is used
to encrypt the data.",
            "properties": {
                "SceneEncryptionKeyID": {
                    "type": "string"
                },
                "SceneEncryptionKey": {
                    "type": "string"
                }
            },
            "required": [
                "SceneEncryptionKeyID",
                "SceneEncryptionKey"
            ]
        },
        "Authentication": {
            "type": "boolean",
            "description": "If true then the device or node shall use its private key
to authenticate SceneData and SceneMarks."
        }
    },
    "required": [
        "PrivacyObjectID",
        "StartDateTime",
        "EndDateTime",
        "Authentication",
        "EndPointID"
    ]
}
```

11.2. DeviceSeller Object

The DeviceSeller Object describes the data that is provided by the Device Seller to the NICE LA.

DeviceSeller Object

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "title": "DeviceSeller",
    "additionalProperties": false,
    "properties": {
        "ManufacturerID": {
            "type": "string"
        },
        "ModelType": {
            "type": "string"
        },
        "DeviceSellerID": {
            "type": "string"
        },
        "DeviceID": {
            "type": "string"
        }
    },
    "required": [
        "ManufacturerID",
        "ModelType",
        "DeviceSellerID",
        "DeviceID"
    ]
}
```