NICE Authentication 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 describes how access to Devices and Apps are managed, including the life cycle for the Device from its manufacture to its deployment and decommissioning and similarly how Apps are deployed. It describes how these processes are managed by the NICE License Authority and NICE Account Service.

2. Overview

The NICE System is made up of Devices, Data Services and Apps which interact with each other. The Authentication Specification describes how these interactions are managed by the NICE License Authority and the NICE Account Service. The Device has the following stages in its life cycle: Manufacture, Installation, Usage and Decommissioning. Through each stage of the Device's life cycle the Device is provisioned with credentials that enable it validate Apps and Data Services that interact with the Device. The operation of the Device through each stage of its life cycle is described in the NICE Device Life Cycle section.

Similarly for Apps and Data Services there are requirements for credentials to be distributed to Apps and Data Services to enable them to interact with Devices or other Data Services. This is described in the App/Service Specification.

The NICE License Authority provides keys, credentials and certificates for Devices that are manufactured. The Device Manufacturer shall insert these into the Device in accordance to the security requirements for each item. The NICE LA shall manage the transition of the Device from manufacture to first installation. During first installation the Device shall be linked to a User's Account on a NICE Account Service. The user logs into their account on the BSS, selects an option to add a Device their account. This results in the user being redirected to the NICE LA which validates the DeviceID, Password and that the Device is not currently linked to an account. This process enables the NICE Account Service to manage the Device on behalf of the User who has the account with the BSS.

A Device can be decoupled from the User's Account and returned to the same state that it was when first manufactured where the NICE License Authority is responsible for managing the access to the Device. The Device may be installed and associated with a different User Account on either the same or different NICE Account Service.

The App developer develops Apps and shall register the App as being available to be linked to User Accounts. The User shall use their User Account on the BSS and the NICE Account Service to link the App to the Devices and Data associated with the User's Account. When the User has provided permission to the NICE Account Service to link the App to their User Account, the NICE Account Service shall use the DataPipelineController to provide Access Tokens to the App Instance or the App Provider's server to enable interaction with one or more of the Devices and Data Services provided for the User.

The process of managing access to Devices or Data Services shall be performed in accordance to the OAuth 2.0 specification. An Access Token is provided to each Entity to enable it to interact with the Device or Data Service. The Access Token contains the parameters for access (time window, permissions etc.) and is protected using the public key provided in the X.509 certificate for the Device or Data Service and authenticated by the NICE Account Service key. This enables a Device or Data Service to determine that the Entity that is requesting access has been granted permission by the NICE Account Service.

All communication between Devices, Data Services and Apps shall be protected using either Transport Layer Security or Datagram Transport Layer Security.

3. NICE Device Life Cycle

The NICE Device has stages in its life cycle where different authorities have control over aspects of the Device. At manufacture of the Device the NICE licensing authority provides credentials to the Device that enable secure communication, secure access and data protection. The NICE Device Life Cycle contains the following steps:

1. Manufacture of Device
2. Installation Phase
3. Operational Phase
4. Uninstallation Phase

▲ Top

3.1. Manufacture of Device

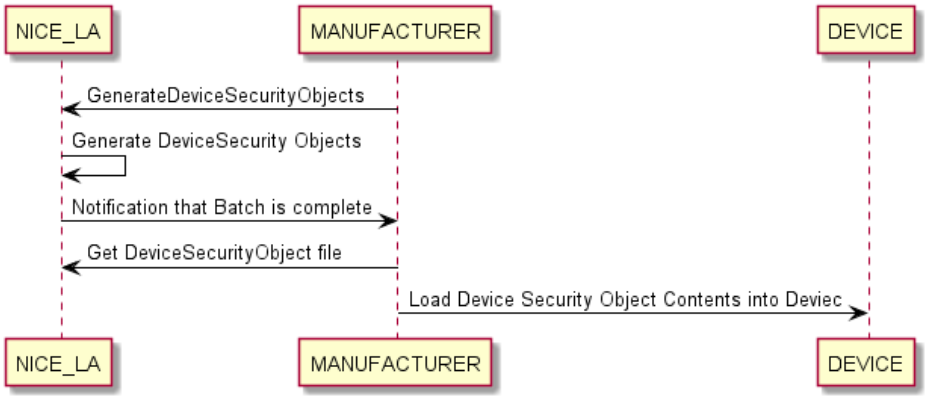

Figure 1. Manufacture of Device

The NICE License Authority securely transfers the Device credentials in the "Device Security Object" to the Manufacturer. Device Security Objects are encrypted with a public key provided by the manufacturer. The manufacturer is responsible for the implementation and security of the insertion of the credentials into the Device.

During manufacture the Device shall have the following credentials inserted into the Device:

1. Device ID: Permanent ID of the Device. Readable by User.
2. Private Key of the Device: Permanent Key of Device - Shall be securely handled and is used for generating an Application level signature of Objects generated by the Device and to decrypt encrypted objects that are sent to the Device. It cannot be updated.
3. NICELA Root Certificate: Root X.509 Certificate for the NICE Licensing Authority.
4. Master Issuer ID. This is the value of the "iss" field that must be present in a JSON Web Token.
5. NICELA URI: URI for the NICELA the Device uses this URI to download settings for its operation.
6. Allowed TLS Root Certificates: which are the Root Certificates that the Device is allowed to accept for securing TLS communication. This allows the Device to interact with browsers and other Devices using other Certificate Authorities than the NICE LA.

The above credentials, with the exception of the certificate for the Firmware Update Service, are permanent for the entire life cycle of the Device.

The Device ID, Unique Password, Private Key and X.509 certificates are issued by the NICE License Authority (LA).

3.2. Installation Phase

The end user can link Devices to their account.

1. The End User shall access the Device ID and Device Password that has been provided with the Device.
    - There may be several methods used to deliver the Device Password to the end user.
    - These may include having the Device ID and Device Password visible on the Device.
2. The Device Password shall only be used to register the Device to a User Account. The Device ID and Device Password is entered on the NICE LA to enable the NICE LA to allocate the Device to a NICE Account Service.
3. The User shall log into their NICE Account Service account. The User shall be directed to the NICE Licensing Authority. The NICE Licensing Authority shall accept the Device credentials provided by the User and shall provide an AccessToken to the User's NICE Account Service. This AccessToken enables the NICE Account Service to interact with the Device.

The NICE LA will only allow the linkage of the Device to a NICE Service Account and provide the Device's X.509 certificate if the Device Seller has already registered the Device with the NICE LA.

Each Device shall have the NICE Licensing Authority Root Certificate stored in the Device.

The Management Object is signed using the Private Signing Key of the NICE Licensing Authority (as defined in the X.509 certificate stored in the Device) and shall be encrypted with the unique Public Encryption Key corresponding to the Private Encryption Key of the Device. The Device Private Encryption Key shall be stored in the Device during manufacture.

When the User wishes to assign the Device to the NICE Account Service, the User shall initiate the NICE Account Management Application to perform the OAuth2 session with the NICE License Authority. The User shall use the factory configured Device ID and Password to enable the NICE License Authority to generate an Access Token for the NICE Account Service to access the Device.

The NICE License Authority shall provide the Management Object to the Device.

This object shall contain the NICE Account Service ID that the Device is being linked to and an expiry time for the message. If the message is received outside of the time defined by the start and expiry times it shall be ignored.

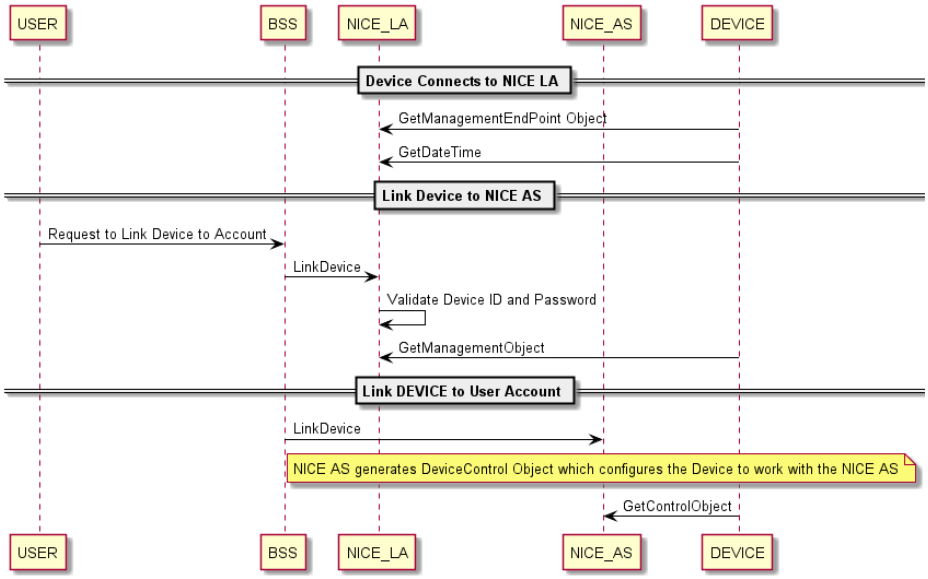

Figure 2. Linking a Device to a User Account

3.3. Operational Phase

When the Device has been associated with a user's account, the NICE Account Service shall provide the DeviceControl Object to the Device when the Device makes the request for it. The NICE Account Service uses the same Access Token to log into the Device and to supply the Device with the DeviceControl Object. The DeviceControl Object shall be encrypted using the Device Public Key and then signed with the NICE Account Service Private key.

The Device is now associated with the User's Account and may be utilized by Data Services or Apps.

3.4. Uninstallation Phase

The End User may decide to use a different NICE Account Service Provider or may give away or sell the Device to another User.

- The End User shall first delete the Device from their Account.
- The NICE Account Server shall communicate with the NICE LA to indicate that the Device is no longer associated with a User Account.
- The NICE LA shall register the Device as being in the state where the device is un-allocated to a User.
- The Device is ready for use by another User.

▲ Top

4. Device Network Security

Network Security covers how the communication links between Entities within the NICE system are secured. All communication shall be protected using either TLS or DTLS (depending on the network protocol).

The DeviceControl Object shall contain a list of end points with which the Device may communicate. The Device shall enforce the encryption settings defined for the end points in the DeviceControl Object.

5. Prevention of Replay Attacks

A replay attack is where a message is captured by the attacker and resubmitted at a later time to reset the Device into an earlier state. To prevent this type of attack, the Management and Control objects shall have a time window parameter within the object outside of which the object shall not be processed by the receiving Device. The time window is enforced by each Entity having a Trusted Time Clock. The synchronization of the clock is described in the Device Trusted Time section.

▲ Top

6. Device Trusted Time

The Device shall maintain a Trusted Time Clock when powered up. A Trusted Time Clock shall be initialized using the Trusted Time Protocol defined in this document and shall be resistant to attempts to change the value of the Trusted Time Clock or change the operation of the Trusted Time Clock. On power up the Device shall request a time stamp from the NICE time server using the Trusted Time Protocol. Once the trusted time stamp has been received it is used to synchronize the trusted time clock within the Device.

▲ Top

7. JSON Web Token Usage in the NICE System

RFC 7519 defines JSON Web Tokens.

- These are tokens containing required and optional fields defined in a JSON format.
- The encryption and authentication of these tokens are defined in the JOSE specifications also developed by the IETF.

The NICE Account Service shall issue Access Tokens that conform to the JSON Web Token definition. Access tokens are provided in the Management, DeviceControl, AppSecurity and AppControl Objects together with the end point definitions for each access token is valid.

7.1. App Access to Services

The NICE Account Service controls which SceneData that may be accessed by an App. The App shall request the AppControl Object which provides a list of end points for Control and Data. These end points also contain the relevant access tokens to enable access to the Control or Data end point. The Access Tokens are refreshed by requesting an AppControl Object.

7.2. JWT (JSON Web Tokens) Format for Application AccessToken

RFC 7519 defines the format for JSON Web Tokens. This specification defines the specific usage of fields defined in this RFC.

7.2.1. Authentication

The token shall be authenticated using the ES256 algorithm (ECDSA using P-256 and SHA-256) or PS256 (RSASSA-PSS using SHA-256 and MGF1 with SHA-256).

- The identification of the public key to be used to authenticate the message shall be carried using an X.509 certificate.
- This certificate shall be validated against the NICE LA root certificate.

7.2.2. Fields

The following example shows the JWT structure that the NICE LA would generate to enable the NICE Account Service to access a specific NICE Device.

- Similar tokens would be generated for Apps to access a user's account or Devices.
- The fields described below shall always be present.
- The values in the fields shall depend on:
    - Who is issuing the token.
    - Who is expected to receive the token.
    - Who is going to use the token.

The following is the payload of the token.

- All the fields defined below are **required** when used in the NICE Ecosystem.

7.2.3. Access Token

- The "**iss**" field refers to the party generating the JWT.
- The "**sub**" field refers to the party that is going to use the token to access a service or a Device.
- The "**aud**" field refers to the party that will receive and validate the token.
    - In case that the token is validated the party shall grant access to the holder of the token.
- The "**exp**" field refers to the expiry date for the token.
    - After this time the token shall not be accepted.
- The "**nbf**" field refers to the date before which the token is not valid.
- The "**iat**" field refers to the date of issue of the token.
- The "**jti**" field is a unique ID for the token.
    - A specific token can be revoked using the Management Object or DeviceControl Object for the Device issued by the NICE LA or NICE Account Service.
    - The Device must check this field against the revocation list delivered in the Management Object.

- The **"Permissions"** field describes which APIs are allowed to be called by the bearer of the Access Token.

▲ Top

7.2.4. Types of Tokens

The following types of tokens are used in the NICE eco system:

	NICE LA to Enable Device to Access NICE LA	NICE LA to Enable Device to Access NICE AS	NICE Account Service to Enable AppInstance to Access NICE AS	NICE Account Service to Enable Device/AppInstance to Access DataPipelineControlle r
"iss"	NICELicenseAuthori ty	NICELicenseAuthori ty	NICEAccountServiceI D	NICEAccountServiceI D
"sub "	DeviceID	DeviceID	AppInstanceID	DeviceID/AppInstanceI D
"aud "	NICE LA ID	NICE AS ID	NICE AS ID	DataPipelineController

The entity receiving the token shall ensure that the following matches:

1. "iss" shall be the same the NICE License Authority provided during manufacture or the NICE Account Service ID provided when the Device is associated with an end user account. This is carried as a data field in the Management Object.
2. "aud" is the EndPointID.
3. The entity identified in the "iss" field should match the X.509 certificate for the entity and the public key carried in that X.509 certificate shall be used to verify the token.

▲ Top

8. Management Interface

8.1. GetDateTime

Function

The App and Device shall use this API to set the current time.

The TrustedTimeRequest Object initiates a request for an authenticated time and date stamp from the NICE Account Service or from the NICE LA. The response from this request is used to synchronize the Trusted Time clock within the trusted execution environment of the Device.

The Device calls the NICE LA for the trusted time when it is not allocated to a NICE AS. Once the Device is associated with a NICE AS through being linked to a user's account, the Device will make this call to the NICE AS.

The App calls the NICE AS for the trusted time. The App shall not send TrustedTime request to NICE LA.

Protocol(s) Used to Make Calls

WebAPI

Direction

Caller	APP, DEVICE
Callee	NICELA, NICEAS

Request Parameters

TrustedTimeRequest Object

Acknowledgement Parameters

TrustedTimeResponse Object

9. Data Objects

9.1. DeviceSecurity

The following JSON Object is provided by the NICE LA to the Device Manufacturer using a secured channel to the Manufacturer's server that is responsible for inserting these fields into the Device.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "DeviceSecurity",
    "description": "Device Security Object generated by the NICE LA",
    "properties": {
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "DeviceCertificate": {
            "type": "string",
            "description": "X.509 Certificate for the Device containing the public key
of the Device."
        },
        "DeviceID": {
            "type": "string",
            "description": "Permanent ID of the device. Readable by end user."
        },
        "DevicePassword": {
            "type": "string",
            "description": "Password used to log into NICE LA device account and
associate device with NICE Account Service Provider"
```

```
        },
        "DevicePrivateKey": {
            "type": "object",
            "description": "Permanent Private Key of Device - Shall be securely
handled. Used for Application level Decryption only.",
            "properties": {
                "EncryptionKeyID": {
                    "type": "string",
                    "description": "Key ID of the Public Key that has been used to
encrypt the AppInstancePrivateKey"
                },
                "EncryptedKey": {
                    "type": "string",
                    "description": "Key format is JWE Compact Serialization(RFC7516)."
                }
            },
            "required": [
                "EncryptionKeyID",
                "EncryptedKey"
            ]
        },
        "AllowedTLSRootCertificates": {
            "$ref": "Definitions.json#/definitions/x5c"
        },
        "NICELARootCertificate": {
            "$ref": "Definitions.json#/definitions/x5c"
        },
        "NICELAEndPoint": {
            "$ref": "Definitions.json#/definitions/NetworkEndPointSpecifier"
        }
    },
    "required": [
        "Version",
        "DevicePassword",
        "DeviceID",
        "DevicePrivateKey",
        "AllowedTLSRootCertificates",
        "NICELARootCertificate",
        "NICELAEndPoint",
        "DeviceCertificate"
    ]
}
```

▲ Top

9.2. Application AccessToken

The Access Token is used by an App or Device to Access either a Data Service or the NICE AS. The Token is signed by the issuer of the token. The issuer may be either the NICE LA or the NICE AS.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "AccessToken",
    "properties": {
```

```
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "iss": {
            "type": "string",
            "description": "Issuer of Token"
        },
        "sub": {
            "type": "string",
            "description": "Subject of Token"
        },
        "aud": {
            "type": "string",
            "description": "Audience for Token"
        },
        "exp": {
            "type": "integer",
            "description": "Expiry date of token"
        },
        "nbf": {
            "type": "integer",
            "description": "Not valid before this date"
        },
        "iat": {
            "type": "integer",
            "description": "Date of Issue"
        },
        "jti": {
            "type": "string",
            "description": "JSON Token ID unique identifier for this token. This
should be checked against revoked tokens for the device before accepting. "
        },
        "Permissions": {
            "type": "array",
            "uniqueItems": true,
            "items": {
                "type": "string",
                "description": "This defines which level of API is accessible to the
holder of this token. For example if management, then the bearer of the token may make
management API calls. ",
                "enum": [
                    "Management",
                    "Control",
                    "Data"
                ]
            }
        }
    },
    "required": [
        "Version",
        "iss",
        "sub",
        "aud",
        "exp",
        "nbf",
        "iat",
        "jti",
        "Permissions"
    ]
}
```

9.3. TrustedTimeRequest

This Object is used by an Entity that is requesting a secured time stamp from either the NICE LA or the NICE AS. The request shall be encrypted using the Public Key of the NICE AS or NICE LA. The request shall be signed using the Private Key of the Entity making the request. The random number generated as part of the challenge protocol shall be generated by a random number generator that conforms to the Device or App Compliance Rules.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "TrustedTimeRequest",
    "properties": {
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "EndPointID": {
            "type": "string"
        },
        "RandomChallenge": {
            "type": "string",
            "description": "Random Number at least 64 bytes long."
         }
    },
    "required": [
        "RandomChallenge",
        "Version",
        "EndPointID"
    ]
}
```

9.4. TrustedTimeResponse

The server shall return the following JSON Object with the Random Challenge correctly decrypted. The Object shall be encrypted using the Device Public Key and signed using a Private Key that is certified by either the NICE LA or NICE AS for this purpose.

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "TrustedTimeResponse",
    "description": "Response containing a time stamp for the requesting device.",
    "properties": {
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "EndPointID": {
            "type": "string"
```

```
        },
        "ReturnedRandomChallenge": {
            "type": "string",
            "description": "Random challenge that was carried in the
TrustedTimeRequest object to which this response is responding. "
        },
        "DateTimeStamp": {
            "type": "string",
            "description": "Time stamp in ISO 8601 format."
        }
    },
    "required": [
        "EndPointID",
        "ReturnedRandomChallenge",
        "DateTimeStamp",
        "Version"
    ]
}
```

9.5. DeviceSeller Object

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

Schema

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "type": "object",
    "title": "DeviceSeller",
    "properties": {
        "Version": {
            "type": "string",
            "enum": [
                "1.0"
            ]
        },
        "ManufacturerID": {
            "type": "string"
        },
        "ModelType": {
            "type": "string"
        },
        "DeviceSellerID": {
            "type": "string"
        },
        "DeviceID": {
            "type": "string"
        }
    },
    "required": [
        "ManufacturerID",
        "ModelType",
        "DeviceSellerID",
        "DeviceID",
        "Version"
    ]
}
```