NetIQ Advanced Authentication Framework- Web Service

Administrator's Guide

Version 5.1.0

Table of Contents

© NetIQ

Introduction

Strong Authentication

Compliance is important in a world where organizations have to adhere to increasingly complex rules and regulations. And information security is already vitally important for every business in our connected global society. No real solution for compliance or information security is possible without proper authentication of users. But authentication by user name and passwords is not reliable anymore. There are lots of stronger authentication methods on the market, but regrettably there is not the best solution in respect to cost, reliability and user convenience in every situation.

Strong Authentication Advantage

Many organizations discovers that traditional password-based authentication systems frustrate users and administrators, while remaining costly to the organization. A recent study cites that password-related calls account for more than 30% of all Helpdesk calls. Unlike passwords, strong authentication systems do not require the hassle of memorizing a series of letters, numbers, and symbols, nor do they require periodic changing.

NeIQ connects to all leading authentication methods. Although there is still a lot of good old username password combination in use for authentication in the virtual world, there is general consensus that passwords are too vulnerable and not secure anymore. New, more stronger, authentication methods have been launched into the market. Authentication methods like contact cards, contactless cards, biometric technologies, one time password generators, hardware/software tokens and many more have all gained traction on the market. But regrettably none of these solutions is the best fit for every authentication requirement within a modern company. There is an urgent need for a more generic solution that enables companies to select a bundle of methods simultaneously, that serves specific and generic needs and is future proof. Adding new methods to NetIQ is a simple standardized and fast process.

NetIQ Advanced Authentication Framework - Web Service Overview

The NetIQ Advanced Authentication Framework - Web Service allows you to authenticate non-domain joined clients.

How Does It Work?

NetIQ Advanced Authentication Framework - Web Service enables users to authenticate in Active Directory/ Novell DSfW using their own authenticators on non-domain joined clients. It interacts with NetIQ Advanced Authentication Framework - Authenticore Server, which performs the actual authentication. The NetIQ Advanced Authentication Framework – Web Service acts as a proxy between the Authenticore server and for example a Linux based client.

The NetIQ Advanced Authentication Framework – Web Service is using SOAP version 1.1 for exchanging structured information and WSDL for describing network service.

Benefits:

- Authenticate users in Domain from the outside of the Domain.
- Users use authenticators of their domain accounts.
- You can use not only Microsoft Windows-based clients, but, for example, Linux-based client or browser on your cell phone or tablet.

Integrates with NetIQ Advanced Authentication Framework Edition

NetIQ Advanced Authentication Framework Edition is a strong authentication security solution that enables users to log on to their workstation and Windows domains. Designed and tested for enterprise-level deployments, NetIQ tightly integrates with Active Directory to allow administrators to secure network and workstation access. See NetIQ Advanced Authentication Framework Administrator's Guide for detailed configuration information, or contact an NetIQ sales representative to learn more about this product.

Getting Started

The following chapters will provide the details on how to get started using the NetIQ Advanced Authentication Framework – Web Service.

What is Included

The distributive of the NetIQ Advanced Authentication Framework - Web Service includes next files:

_webservice
- webservice.msi – NetIQ Advanced Authentication Framework - Web Service installer.

GUIDES
- Web Service - Administrator's Guide.pdf – NetIQ Advanced Authentication Framework - Web Service documentation in pdf format.

NetIQ Advanced Authentication Framework – Web Service

NetIQ Advanced Authentication Framework – Web Service performs the following challenges:

1. System.Guid Logon(string domain, string username, string subsystem, byte[] identifier)
The function provides authentication using domain name, username, subsystem name and identifier wrapped by BioAPI.

Example:

```
1.  string domain = "authasas"; // short name of domain
2.  string username = "jsmith";
3.  Guid guid = new Guid("{c7d6704e-f66a-4ef0-93a3-c5ef13f0c7a2}"); // GUID of
    authentication provider
4.  byte[] ident = …; // Specific identifier of authentication provider
5.
6.  Guid sessionId = LogonRaw(domain, username, guid.ToByteArray(),"", ident);
7.
8.  // If the username is in the UPN format then:
9.  string username = "jsmith@authasas.local";
10. Guid sessionId = LogonRaw("", username, guid.ToByteArray(),"", ident);
```

2. System.Guid Logon1N(string domain, string subsystem, byte[] identifier)
The function provides 1-N authentication using domain name, subsystem name and identifier wrapped by BioAPI. Can be used along with Flash Drive + PIN and Universal Card authentication service providers.

Example:

```
1.  string domain = "authasas"; // short name of domain
2.  Guid guid = new Guid("{c7d6704e-f66a-4ef0-93a3-c5ef13f0c7a2}"); // GUID of
    authentication provider
3.  byte[] ident = …; // Specific identifier of authentication provider
4.
5.  Guid sessionId = Logon1NRaw(domain, guid.ToByteArray(),"", ident);
```

3. System.Guid LogonRaw(string domain, string username, byte[] bspId, string subsystem, byte[] rawIdentifier)
The function provides authentication using domain name, username, authentication service provider identifier, subsystem name and raw authentication data. See Preparing raw authentication data.

Example is the same as for Logon function, only ident differs for Logon and LogonRaw functions.

4. System.Guid Logon1NRaw(string domain, byte[] bspId, string subsystem, byte[] rawIdentifier)
The function provides authentication using domain name, authentication provider identifier, subsystem name and raw authentication data. Can be used along with FlashDrive+PIN and Universal Card authentication providers. See Preparing raw authentication data.

Example is the same as for Logon1N function, only ident differs for Logon1N and Logon1NRaw functions.

After successful authentication NetIQ Advanced Authentication Framework - Web Service can return user password and/or username if you need. These actions use the following functions:

- string GetPassword(System.Guid sessionId);
- string GetUserName(System.Guid sessionId).

Example:

```
1.  string username = GetUserName(sessionId);
2.  string password = GetPassword(sessionId);
```

5. String GetUserNameByCardId(string cardId, long nameFormat)
The function returns the user name that is linked with a smart card ID.
CardId - hex string represents actual card ID.
NameFormat sets format in what user name will be shown:

- LDAP name format = 1,
- GUID name format = 2,
- UPN name format = 3,
- NT4 name format = 4.

Example:

```
1.  string cardId = "000000000001006e";
2.
3.  // LDAP Name Format - 1
4.  // GUID Name Format - 2
5.  // UPN Name Format - 3
6.  // NT4 Name Format - 4
7.
8.  int nameFormat = 1;
9.
10. // returns username in one of the supported formats
11. string username = service.GetUserNameByCardId(cardId, nameFormat);
```

6. string GetUserMemberGroups(Guid sessionId)
The function returns the list of groups of the current authenticated user.

The sessionId parameter is the identifier of the current session that can be retrieved after calling one of the Logon functions. The function returns the jagged array of strings, known as an array of arrays. Every element of this array is an array on its own, containing two values: the first value is a group SID, the second is a group sAMAccountName.

Example:

```
1.  string[][] groups = service.GetUserMemberGroups(sessionId);
2.
3.  string[] entry = groups[0];
4.  string sid = entry[0];
5.  string name = entry[1];
```

7. bool IsUserMemberOfGroup(Guid sessionId, String group, bool isSid);
The function checks whether the current authenticated user belongs to the provided group, specified by the group parameter.
The sessionId parameter is the identifier of the current session that can be retrieved after calling one of the Logon functions. Depending on the value of the isSid parameter, the group parameter represents either the group SID (if isSid equals true) or the group sAMAccountName (if isSid equals false).

Example:

```
1.  string groupName = "admins";
2.  string groupSid;
3.  bool isMember = service.IsUserMemberOfGroup(sessionId, groupName, false);
4.  bool isMember = service.IsUserMemberOfGroup(sessionId, groupSid, true);
```

ℹ️   String GetUserMemberGroups (Guid sessionId) and bool IsUserMemberOfGroup (Guid sessionId, String group, bool isSid) work for all type of groups, except primary (CN=Users).

Extra examples:

Getting session ID:

```
1.  string session = service.GetSessionID();
```

Getting last error:

```
1.  uint result = service.GetLastError(); // HResult
```

Logon by password:

```
1.  string domain = "authasas"; // short name of domain
2.  string username = "jsmith";
3.  string password = "Qwerty123";
4.
5.  Guid sessionId = LogonByPassword (domain, username, password,"");
```

Getting username in specified format:

```
1.  // LDAP Name Format - 1
2.  // GUID Name Format - 2
3.  // UPN Name Format - 3
4.  // NT4 Name Format - 4
5.
6.  int nameFormat = 1;
7.
8.  String username = service.GetUserName2(nameFormat, sessionId);
```

Terms

1-N authentication – is an authentication mode when there is no need to input username. It will be detected automatically after authentication by NetIQ Advanced Authentication Framework - Authenticore Server.

Authentication provider – is a component that implements communication between NetIQ Advanced Authentication Framework and authentication device.

BSP – see *Authentication provider*.

One-Time Password – is a temporary password which is automatically generated at the current time in according to special algorithm.

OTP – see *One-Time Password*.

Subsystem – is an object of Active Directory that contains the data such as usernames, passwords, records, custom data and etc. These data are used by NetIQ Advanced Authentication Framework.

Checking State of NetIQ Web Service

You can check a state of NetIQ Advanced Authentication Framework – Web Service simply by opening the following webpage in your browser: https://<servername>/Service.svc?wsdl, where <servername> is a name or IP-address of your IIS server. If you will see XML-page containing AuthLogon definitions name, then the NetIQ Advanced Authentication Framework – Web Service works correctly. In case of any problems please follow the next instruction:

1. Open IIS Manager and check status of AuthWebService.

2. If AuthWebService site uses not default port, please try to use the next link: https://<servername>:<portnumber>/Service.svc?wsdl, where <portnumber> - port number using by AuthWebService.

3. If you got Certificate Error please install required certificate.

Identifiers of Authentication Providers

You can use the following authentication providers GUIDs for *bspId* parameter:

- BIO-key Biometric Provider Version 1.9 – {EC4AC729-B969-6E46-BD2F-56B6055E18F8}
- Universal Card Authentication Provider – {ED2D1872-4DAC-A84B-AF7C-188642267D56}
- USB Flash Drive Authentication Provider – {1AF29AB5-0A30-0046-95DB-4FDA28989051}
- OATH OTP Authentication Provider – {C7D6704E-F66A-4EF0-93A3-C5EF13F0C7A2}
- RADIUS Authentication Provider – {E4828EC2-B520-46FC-9624-EB98487A7F2B}

Preparing Raw Authentication Data

In this chapter:

- [Authentec Authentication Provider](#)
- [Authentec + Card Authentication Provider](#)
- [BIO-key+PIN Authentication Provider](#)
- [Digital Persona Authentication Provider](#)
- [Email Authentication Provider](#)
- [FIDO U2F Authentication Provider](#)
- [Flash Drive + PIN Authentication Provider](#)
- [Hitachi Fingervein Authentication Provider](#)
- [Hitachi Fingervein+Card Authentication Provider](#)
- [Hitachi Fingervein+Card+Password Authentication Provider](#)
- [Innovatrics Authentication Provider](#)
- [Intrinsic Authentication Provider](#)
- [Live Ensure Authentication Provider](#)
- [Lumidigm Authentication Provider](#)
- [Lumidigm+Card Authentication Provider](#)
- [OATH Authentication Provider](#)
- [RADIUS Authentication Provider](#)
- [Security Questions Authentication Provider](#)
- [Smartphone Authentication Provider](#)
- [SMS Authentication Provider](#)
- [Universal Card Authentication Provider](#)
- [Voice Call Authentication Provider](#)

Authentec Authentication Provider

For both enroll and logon templates raw data are obtained from Authentec SDK (without any wrappers and headers).

Authentec + Card Authentication Provider

Both enroll and logon templates do not use TVL. The following hardcoded structure should be filled:

- Wide char (UTF-16) string "REMOVABLE_DEVICE_SIGN" with trailing zero (c-style string).
- Unsigned integer (4 bytes) with length of card ID string. Always 33.

- Wide char (UTF-16) string with card ID. 33 chars. Card ID format is described in [Universal Card Authentication Provider](#) chapter.
- Unsigned integer (4 bytes) with length of fingerprint data.
- Raw data obtained from Authentec SDK.

BIO-key+PIN Authentication Provider

Both enroll and logon templates use TVL with the following fields:

- 0 - UTF-16 string - PIN/password. Should be empty for enroll template if domain password is used.
- 1 - binary - fingerprint data.
- 2 - UTF-16 string - user SID. Used to check PIN cache.

Digital Persona Authentication Provider

Both enroll and logon templates use TVL with the following fields:

- 0 - UTF-16 string - PIN/password. Should be empty for enroll template if domain password is used.
- 1 - binary - fingerprint data.
- 2 - UTF-16 string - user SID. Used to check PIN cache.
- 3 - binary - only for enroll template if several fingers were enrolled (supported in last versions of BSP). In this case for each finger the following data will be present:
  - Fingerprint data length
  - Fingerprint data

Email Authentication Provider

Enroll template is empty.

Logon template uses TLV with the following fields:

- 0 - 1 byte - Ident type. Should be 1.
- 2 - UTF-16 string - domain password.
- 3 - binary - SHA1 hash of entered OTP.
- 4 - 1 byte - check password flag. If set to 0, password should not be checked. This flag is used only in NCA. By default 1 should be passed.

FIDO U2F Authentication Provider

Both enroll and logon template use TLV with the following fields:

- 0 - binary - public key.
- 1 - binary - key handle.
- 2 - binary - hash of U2F token data.
- 3 - binary - signature of hash.
- 4 - UTF-16 string - domain password.

All these fields are stored in enroll template only but should be present in logon template also (empty in this case).

Flash Drive + PIN Authentication Provider

Authenticator's File

When you enroll Flash Drive + PIN authenticator NetIQ Advanced Authentication Framework creates hidden file named FlashPinBspLogon.dat on enrolled USB flash drive. This file contains encrypted private key. Private key is generated via RSA algorithm with 1024 bit length. Encryption type is DES. The encryption key is SHA-1 hash of the specified PIN code.

Preparing Raw Authentication Data

This is the description of algorithm for forming raw authentication data for Flash Drive + PIN authentication provider:

1. Generate 20 random bytes.
2. Get SHA-1 hash of 20 random bytes.
3. Get signature - sign hash by a private key from FlashPinBspLogon.dat file on enrolled USB Flash Drive using CryptoAPI.
4. Write array of the following data:

   - "REMOVABLE_BSP_SIGN" string in UTF-16 encoding with zero at the end.
   - Length of USB flash drive serial number (4 bytes, decimal).
   - USB flash drive serial number in UTF-16 encoding without zero at the end.
   - Length of 20 random bytes (4 bytes).
   - 20 random bytes.
   - Length of signature (4 bytes).
   - Signature.

Both enroll and logon templates use the following fixed structure:

- Wide char (UTF-16) string "REMOVABLE_DEVICE_SIGN" with trailing zero (c-style string).
- Unsigned integer (4 bytes) with length of flash drive ID string. Always 33.
- Wide char (UTF-16) string with flash drive ID. 33 chars. MD5 hash from UTF-16 representation of volume serial number.
- Unsigned int (4 bytes) – template version. Always 1.
- BYTE array with MD5 hash of user PIN or domain password. 16 bytes.
- BYTE array with random data used as template data. 128 bytes.

Hitachi Fingervein Authentication Provider

Both enroll and logon templates use the following format:

- Binary data (sizeof(BioAPI_BIR)) store BioAPI template header.
- Binary data (size from header) store biometric data (fingervein info).
- Binary data (size from header) store security data (usually empty).

Hitachi Fingervein+Card Authentication Provider

Both enroll and logon templates use the following format:

- Wide char (UTF-16) string "REMOVABLE_DEVICE_SIGN" with trailing zero (c-style string).
- Unsigned integer (4 bytes) with length of card ID string. Always 33.
- Wide char (UTF-16) string with card ID. 33 chars. Card ID format will be described in Universal Card BSP.
- Fingervein data (TLV) with the following fields:
  - 1 - Binary - BioAPI header
  - 2 - Binary - biometric data
  - 3 - Binary - security data

Hitachi Fingervein+Card+Password Authentication Provider

Both enroll and logon templates use the following format:

- Wide char (UTF-16) string "REMOVABLE_DEVICE_SIGN" with trailing zero (c-style string).
- Unsigned integer (4 bytes) with length of card ID string. Always 33.
- Wide char (UTF-16) string with card ID. 33 chars. Card ID format will be described in Universal Card BSP.

- Fingervein data (TLV) with the following fields:
    - 1 - Binary - BioAPI heder
    - 2 - Binary - biometric data
    - 3 - Binary - security data
    - 4 - UTF-16 string - domain password (empty for enroll template)

Innovatrics Authentication Provider

For both enroll and logon templates raw data are obtained from Innovatrics SDK (without any wrappers and headers).

Intrinsic Authentication Provider

Both enroll and logon templates use TLV with the following fields:

- 0 - binary - HSD ID (hardware security ID).
- 1 - binary - DAK (filled in enroll template, empty in logon template).
- 2 - binary - Kc (filled in enroll template, empty in logon template).
- 3 - binary - Challenge (empty in enroll template, filled in logon template, random data).
- 4 - binary - Response (empty in enroll template, filled in logon template).
- 5 - UTF-16 - optional PIN or domain password (domain password is empty in enroll template).

For more information on these fields, check Intrinsic specifications.

Live Ensure Authentication Provider

Enroll template uses TLV with the following fields:

- 0 - UTF-16 - user e-mail.

Logon template uses TLV used with the following fields:

- 0 - UTF-16 - user e-mail.
- 2 - UTF-16 - user account name

One of these fields should be present. If e-mail is present, it will be used. In other case, if account name is present, user e-mail will be obtained from AD.

Lumidigm Authentication Provider

Both enroll and logon templates use TLV with the following fields:

- 0 - UTF-16 - optional PIN or domain password (domain password should be empty for enroll template).
- 1 - binary - fingerprint.
- 2 - UTF-16 - User SID.

Lumidigm+Card Authentication Provider

Both enroll and logon templates use TLV with the following fields:

- 0 - UTF-16 - Card ID (for more information, check Universal Card Authentication Provider).
- 1 - binary - fingerprint.

OATH Authentication Provider

OATH TOTP (Time-based One-Time Password) is based on TOTP algorithm. We use seed which contains 40-hex digits only when enrolling an authenticator. Later mobile application generates temporary code which contains 6 decimal digits depending on seed, generation interval and current time. User inputs this temporary code for authentication. OATH Authentication Provider generates few temporary codes for the situation if the time on mobile device and NetIQ Workstation and NetIQ Authentication Server differs from each other. The number of temporary codes depends on NetIQ TOTP checking window policy. Please read NetIQ OATH Authentication Provider User Guide.

Enroll template uses TLV format with the following fields:

- 0 - 1 byte - ident type (0 for enroll template).
- 1 - 1 byte - OTP type (0 - TOTP, 1 - HOTP).
- 14 - unsigned int (4 bytes) - OTP length (in digits).
- 3 - binary - key (seed).
- 4 - binary - template GUID (16 bytes).
- 7 - UTF-16 string - User SID.
- 15 - UTF-16 string - token ID.
- 16 - unsigned int (4 bytes) - counter (for HOTP).
- 5 - binary - MD5 hash of PIN as UTF-16 string.
- 6 - UTF-16 string - PIN in clear text (available in new version).

Logon template uses TLV format with the following fields:

- 0 - 1 byte - ident type (1 for logon template).
- 1 - 1 byte - OTP type (0 - TOTP, 1 - HOTP).
- 2 - 1 byte - Protocol (0 - PAP, 1 - CHAP, 2 - MSCHAPv2).
- 4 - binary - SHA1 hash of PIN as UTF-16 string.

RADIUS Authentication Provider

RADIUS Authentication Provider can be used along with various RADIUS servers. Authentication data of the provider can be a domain password in the case when RADIUS server uses authentication by domain password or other passwords, or OTP depending on RADIUS server settings. These data are submitted in C String (i.e. ASCII string with terminal zero ('\0').

Enroll template can be empty. In other case, account name (with optional domain) should be held in ANSI code page.

For logon template the password should be held in ANSI code page.

Security Questions Authentication Provider

Both templates enroll and logon templates have the following structure:

- Header:
    - Unsigned int (4 bytes) - signature (0x0EAAAE0)
    - Unsigned int (4 bytes) - version (1)
    - Unsigned int (4 bytes) - total length of data in bytes
    - Unsigned int (4 bytes) - number of items
- For each item (answer):
    - UTF-16 string with question ID (obtained from group policies)
    - UTF-16 string with answer

Smartphone Authentication Provider

Enroll template uses TLV with the following fields:

- 0 - binary - device ID obtained from phone.
- 1 - binary - Secret.

- 2 - 1 byte - device type (0 - iOS, 1 - Windows Phone, 2 - Android).
- 3 - unsigned int (4 bytes) - TOTP step.
- 4 - unsigned int (4 bytes) - TOTP length.
- 5 - unsigned int (4 bytes) - TOTP time interval.
- 6 - Symmetric key for Intrinsic chips on Android devices.
- 7 - UTF-16 string - User SID.

Logon template uses TLV format with the following fields:

- 0 - 1 byte - always 1.
- 1 - 1 byte - always 0.
- 2 - 1 byte - always 0.
- 4 - binary - SHA1 hash of domain password as UTF-16 string.

SMS Authentication Provider

Enroll template is empty.

Logon template uses TLV with the following fields:

- 0 - 1 byte - Ident type. Should be 1.
- 2 - UTF-16 string - domain password.
- 3 - binary - SHA1 hash of entered OTP.
- 4 - 1 byte - check password flag. If set to 0, password should not be checked. This flag is used only in NCA. By default 1 should be passed.

Universal Card Authentication Provider

Please select your version of NetIQ Advanced Authentication Framework:

- [NetIQ Advanced Authentication Framework v4.7 and earlier](#)
- [NetIQ Advanced Authentication Framework v4.8 and later](#)

Authasas Advanced Authentication v4.7 and earlier

This is the description of algorithm for forming raw authentication data for Universal Card authentication provider of NetIQ Advanced Authentication Framework v4.7 and earlier:

1. Get MD5 hash of specified PIN code using Windows CryptoAPI (16 bytes).
2. Get MD5 hash of Card serial number (16 bytes).
3. Write array of the following data:

   - "REMOVABLE_BSP_SIGN" string
   - length of Card serial number
   - MD5 hash of Card serial number
   - MD5 hash of specified PIN code
   - GUID of Card type (see Identifiers of card types)
   - length of additional card information

Identifiers of Card Types

You can use the following GUIDs for different supported card types:

- RF IDeas cards: {2976548C-9797-450E-91D4-4CA4451A14D2}
- OMNIKEY cards: {C91A345F-FDA9-4BC7-8F96-902B25E33011}
- TMC Legic cards: {6BF122E4-695E-4589-8B29-E9C797F045D3}

Below is an example of algorithm for Universal Card AP of NetIQ Advanced Authentication Framework v4.7 and earlier:

```java
1.  package com.authasas.aaa.method.smartcard;
2.
3.  import com.authasas.aaa.routines.Converter;
4.  import com.authasas.aaa.method.Identifier;
5.  import com.authasas.aaa.routines.TlvWriter;
6.  import java.security.MessageDigest;
7.  import java.util.logging.Level;
8.  import java.util.logging.Logger;
9.  import sun.security.util.Password;
10.
11. /**
12.  *
13.  * @author Lex85
14.  */
15. public class CardIdentifier implements Identifier {
16.
17.     private static final Logger logger = Logger.getLogger("NAMLogger");
18.
19.     public static final String STRING_ENCODING = "UTF-16LE";
20.     public static final String HASH_ALGORITHM = "MD5";
21.     public static final String HEADER = "REMOVABLE_DEVICE_SIGN";  |
22.     public static final int PACKAGE_SIZE = 150;
23.     public static final int CARD_HASH_LENGTH = 33;
24.     //
25.     private String cardID;
26.     private String pin;
27.     private byte[] pluginID;
28.
29.     public CardIdentifier(String cardID, String pin, byte[] pluginID) {
30.         this.cardID = cardID;
31.         this.pin = pin;
32.         this.pluginID = pluginID;
33.     }
34.
35.     @Override
36.     public byte[] getBytes() {
37.         byte[] bytes = new byte[0];
38.         //
39.         try {
40.             MessageDigest md = MessageDigest.getInstance(HASH_ALGORITHM);
41.
42.             TlvWriter writer = new TlvWriter();
43.             writer.write((byte)0, (pin + '\0').getBytes(STRING_ENCODING));
44.             writer.write((byte)1, pluginID);
45.             byte[] newIdent = writer.getBytes();
46.
47.             bytes = new byte[PACKAGE_SIZE + newIdent.length];
48.             // 0-43
49.             byte[] headerBytes = HEADER.getBytes(STRING_ENCODING);
50.             System.arraycopy(headerBytes, 0, bytes, 0, headerBytes.length);
51.             // 44-47
52.             byte[] sizeBytes = Converter.getBytes(CARD_HASH_LENGTH);
53.             System.arraycopy(sizeBytes, 0, bytes, 44, sizeBytes.length);
54.             // 48-113
55.             int offset = 48;
56.             md.update(cardID.getBytes(STRING_ENCODING));
57.             for (byte b : md.digest()) {
58.                 byte[] tempBytes = String.format("%02x", b).getBytes(STRING_ENCODING);
59.                 System.arraycopy(tempBytes, 0, bytes, offset, tempBytes.length);
60.                 offset += tempBytes.length;
```

```
61.            }
62.            // kinda a sign of a new format
63.            for (int i = 114; i <= 145; i++) {
64.                bytes[i] = -1;
65.            }
66.            // 114 - 129
67. //           md.update(pin.getBytes(STRING_ENCODING));
68. //           byte[] pinBytes = md.digest();
69. //           System.arraycopy(pinBytes, 0, bytes, 114, pinBytes.length);
70.            // 130 - 145
71. //           System.arraycopy(pluginID, 0, bytes, 130, pluginID.length);
72.            // 146 - 149
73.            sizeBytes = Converter.getBytes(newIdent.length);
74.            System.arraycopy(sizeBytes, 0, bytes, 146, sizeBytes.length);
75.            // 150 ---
76.            System.arraycopy(newIdent, 0, bytes, 150, newIdent.length);
77.
78.        } catch (Exception ex) {
79.            logger.log(Level.SEVERE, ex.toString());  |
80.        }
81.        //
82.        return bytes;
83.    }
84. }
```

Authasas Advanced Authentication v4.8 and later

Starting from NetIQ Advanced Authentication Framework v4.8 and later a new algorithm for forming raw authentication data for Universal Card authentication provider is supported. The differences are the following:

- PluginID and PINHash are filled with 0xff
- additional card information contains TVL structure of the following format:

        TlvWriter Writer;
        Writer.Write(0, _PinOrPassword);
        Writer.Write(1, &_PluginId, sizeof(GUID));
        if (_AdditionalCardInfo.size())
        Writer.Write(2, _AdditionalCardInfo.data(), _AdditionalCardInfo.size());

- length of additional card information contains TVL structure length in bytes

TVL field has the following format:

- T - 1 byte (field type or code)
- L - 4 bytes - field data length
- V - L bytes - field data

Below is an example of algorithm for Universal Card AP of NetIQ Advanced Authentication Framework v4.8 and later:

```
1.   package com.authasas.aaa.routines;
2.
3.   import java.util.ArrayList;
4.
5.   public class TlvWriter {
6.
7.       private ArrayList<Byte> list = new ArrayList<Byte>();
8.
9.       public TlvWriter() {
10.      }
11.
12.      public void write(byte key, byte data) {
13.          write(key, new byte[]{data});
14.      }
15.
16.      public void write(byte key, byte[] data) {
17.          list.add(key);
18.          //
19.          byte[] sizeBytes = Converter.getBytes(data.length);
20.          for (int i = 0; i < sizeBytes.length; i++) {
21.              list.add(sizeBytes[i]);
22.          }
23.          //
24.          for (int i = 0; i < data.length; i++) {
25.              list.add(data[i]);
26.          }
27.      }
28.
29.      public byte[] getBytes() {
30.          byte[] bytes = new byte[list.size()];
31.          for (int i = 0; i < list.size(); i++) {
32.              bytes[i] = list.get(i);
33.          }
34.          return bytes;
35.      }
36. }
```

Voice Call Authentication Provider

Enroll template uses TLV with the following fields:

- 0 - binary - hashed PIN code (MD5).
- 1 - UTF-16 - User SID.
- 2 - UTF-16 - User domain.
- 3 - unsigned int (4 bytes) - template version (currently 3).

Logon template uses TLV with the following fields:

- 0 - 1 byte - Ident type (0 - enroll, 1 - logon).
- 1 - 1 byte - protocol (0 - PAP, 1 - CHAP, 2 - MSCHAPv2).
- 2 - binary - present if PAP used - hashed domain password (SHA1).

Error Codes Description

Here you can find the description of possible NetIQ errors. Also you can use MSDN website for detailed description of Microsoft and Windows Script Host errors (such as 0x80070005 "Access is denied").

In this chapter:

- RPC Server Errors
- SrvWrapper Errors
- Password Filter Errors
- Password Manager Errors
- EventLog Errors
- BioAPI Errors
- Authenticore Server Errors
- Authenticore Providers Errors
- Cryptography Errors
- Manager Errors
- Plugins Errors
- Licensing Errors
- Backup Provider Errors
- Administration Tools Errors
- GINA Errors
- Data Errors

RPC Server Errors

0xC0FF0001L
RPCS_E_WAIT_FOR_INSTALL
Server installation was not completed. At the moment, server is awaiting for installation completion. Server is not able to work until the process is finished.

0xC0FF0002L
RPCS_E_ALREADY_INSTALLED
Server is already installed. At the moment, it is working normally. Installation completion is not required.

0xC0FF0003L
RPCS_E_CAN_NOT_IMPERSONATE
Could not impersonalize.

0xC0FF0008L
RPCS_E_CREATE_CIPHER
Authenticore server could not create Cipher COM-object. Either the object was not registered in the process of system installation or it could not get the Enterprise Key.

0xC0FF0009L
RPCS_E_CREATE_DATA_PROVIDER
Server could not create ADUserDataProvider object. Perhaps, the object was not registered while installing the system.

0xC0FF000AL
RPCS_E_CREATE_KEYMANAGER
Authenticore server could not create KeyManager COM-object. Perhaps, the object was not registered while installing the system.

0xC0FF000BL
RPCS_E_CREATE_LOGON
Authenticore server could not create Logon COM-object. Perhaps, the object was not registered while installing the system.

0xC0FF000CL
RPCS_E_CREATE_MANAGER
Authenticore server could not create Manager COM-object. Perhaps, the object was not registered while installing the system.

0xC0FF000DL
RPCS_E_GENERATE_OR_WRITE_KEYS
Could not generate or save Enterprise Key. This computer may have problems either with the CryptoAPI or with keys storing infrastructure.

0xC0FF000EL
RPCS_E_LISTEN_CALLS
Error calling RpcServerListen.

0xC0FF000FL
RPCS_E_LOGON_USER
Could not log in as AuthenticoreService.
Possible error causes:
- there is no AuthenticoreService account in the domain;
- account password and AuthenticoreService account unsynchronized;
- AuthenticoreService account was automatically blocked;
- AuthenticoreService account does not have "batch job" logon privileges on this computer.

© *NetIQ*

0xC0FF0010L
RPCS_E_READ_USER_NAME
Server could not read the name of user account under which the server must work.

0xC0FF0011L
RPCS_E_REGISTER_INTERFACE
Server could not register RPC-interface.

0xC0FF0012L
RPCS_E_WRITE_USER_NAME
Server could not save user account name under which it must work.

0xC0FF0013L
RPCSKEY_E_WRONG_CLIENT
Server requested the Enterprise Key, is not the domain member or its request is incorrect.

0xC0FF0014L
RPCSKEY_E_GET_TICKET
Could not get Kerberos Ticket of the Authenticore server which requested the Enterprise Key.

0xC0FF0015L
RPCSKEY_E_NOT_LOCAL_CALL
This function is intended for the local call only.

0xC0FF0016L
RPCSKEY_E_CONNECT_SERVER
Could not find Authenticore server or establish connection with it.

0xC0FF0017L
RPCSKEY_E_REGISTER_SPN
Could not register Service Provider Name (SPN).

0xC0FF0018L
RPCSKEY_E_CREATE_TIKET
Could not get Kerberos Ticket using data received from Authenticore server.

0xC0FF0019L
RPCSKEY_E_GET_TICKET_NO_SPN
Could not get Kerberos Ticket from Authenticore server, which had requested Enterprise Key: SPN is not registered. Most likely, the error occurred because Active Directory data replication had not been completed. In this case, please wait until replication is completed and then click Retry button.

0xC0FF001AL
RPCSKEY_E_CLIENT_NOT_MEMBER_OF_GROUP
Authenticore server, which has requested Enterprise Key, is not included into the Authenticore Servers group. Most likely, the error occurred because Active Directory data replication had not been completed. In this case, please wait until replication is completed and then click Retry button.

0xC0FF001BL
RPCS_E_NO_DELEGATE
The level of impersonalization, allowed by the requested side, is lower than "Delegate" level.

0xC0FF001CL
RPCS_E_WAIT_FOR_LICENSE
Server installation has not been completed. Currently the server is in progress of adding license.

0xC0FF001DL
RPCS_E_DELEGATION_DISABLED
Computer account is not trusted for delegation.

0xC0FF001EL
RPCS_E_SENSITIVE_ACCOUNT
Cannot connect to the Authenticore server. Please, ensure that for your account the "Account is sensitive and cannot be delegated" option is turned off.

0xC0FF0463L
RPCS_E_LOGON_LOGON_FAILED
Could not authenticate the user by provided authenticator.

0xC0FF044DL
RPCS_E_LOGON_LOGON_FAILED
Could not authenticate the user by provided authenticator.

0xC0FF044FL
RPCS_E_LOGON_LOGON_BY_PASSWORD_FAILED
Could not authenticate the user by the entered password.
The error could also occur if the entered account was invalid.

0xC0FF0451L
RPCS_E_ENUM_TEMPLATES_PUT_ITEM_FAILED
User could not re-enroll the authenticator.

0xC0FF0453L
RPCS_E_ENUM_TEMPLATES_ADD_FAILED
User could not add new authenticator.

0xC0FF0455L
RPCS_E_ENUM_TEMPLATES_REMOVE_FAILED
User could not remove the authenticator.

0xC0FF0456L
RPCS_E_SERVER_SHUTDOWN
Authenticore Server service is stopped.

0xC0FF045BL
RPCS_E_FIND_SERVER
Could not find Authenticore server.

0xC0FF045EL
RPCS_E_FIND_LICENSED_SERVER
Could not find Authenticore server with active license.

0xC0FF0461L
RPCS_E_ADD_LICENSE
Could not add license.

0xC0FF0463L
RPCS_E_LOGON_LOGON_FAILED_EX
Could not authenticate the user by provided authenticator.

0xC0FF0465L
RPCS_E_ADD_LICENSE_EX
Could not add license.

0xC0FF04BBL
RPCS_E_MANAGER_CREATE_FAILED
Could not permit User to use authenticators.

0xC0FF04BDL
RPCS_E_MANAGER_REMOVE_FAILED
Could not forbid authenticators for User.

0xC0FF04CCL
RPCS_E_USER_PUT_SETTINGS_FAILED

Could not initialize settings for User.

0xC0FF04CEL
RPCS_E_USER_CLEAN_AUTHENTICATORS_FAILED
Could not clear the list of enrolled authenticators of user.

0xC00004CFL
RPCS_E_COMPUTER_CANTWRITEOBJECT
Could not initialize settings for computer.

0xC0FF04D2L
RPCS_E_USER_GET_SETTINGS_FAILED
Could not obtain settings for User.

0xC0FF04D3L
RPCS_E_USER_GET_TEMPLATES_FAILED
Could not get the list of enrolled authenticators of user.

0xC0FF04D4L
RPCS_E_USER_CHANGE_PASSWORD_FAILED
Could not change password for user.

0xC0FF04D5L
RPCS_E_USER_PUT_PASSWORD_FAILED
Could not set password for user.

0xC0FF0516L
RPCS_E_SERVER_CAN_NOT_START
Could not start Authenticore Server service.

0xC0FF0517L
RPCS_E_SERVER_CAN_NOT_READ
Authenticore Server service could not read data from Active Directory.

0xC0FF0518L
RPCS_E_SERVER_CAN_NOT_WRITE
Authenticore Server service could not write data into Active Directory.

0xC0FF0519L
RPCS_E_SERVER_CAN_NOT_DECODE
Authenticore Server service could not decrypt data retrieved from Active Directory.
Either data was corrupted or the Enterprise Key has been changed.

0xC0FF051BL
RPCS_E_GETKEYS_FAILED
Could not transfer Enterprise Key to server.

0xC0FF051CL
RPCS_E_GETKEYS_FROM_FAILED
Could not get Enterprise Key from server.

0xC0FF051DL
RPCS_E_GETKEYS_FROM_FAILED
Could not get Enterprise Key from server.

0xC0FF051FL
RPCS_E_EXPORT_KEYS_FAILED
Could not export Enterprise Key.

0xC0FF0521L
RPCS_E_IMPORT_KEYS_FAILED
Could not import Enterprise Key.

0xC0FF0523L
RPCS_E_GENERATION_KEYS_FAILED
Could not generate Enterprise Key.

0xC0FF0524L
RPCS_E_AD_IS_OFFLINE
Active Directory is offline.

0xC0FF0461L
RPCS_E_ADD_LICENSE
Could not add license.

0xC0FF045EL
RPCS_E_FIND_LICENSED_SERVER
Could not find Authenticore server with valid license.

0xC0FF0465L
RPCS_E_ADD_LICENSE_EX
Could not add license.

0xC0FF001DL
RPCS_E_DELEGATION_DISABLED
Computer account is not trusted for delegation.

0xC0FF001EL

RPCS_E_SENSITIVE_ACCOUNT

Cannot connect to the Authenticore server. Please, ensure that for your account the "Account is sensitive and cannot be delegated" option is turned off.

0xC0FF06BCL

RPCS_E_LOGON_REFUSED_BY_RULES

Logon refused by security rules.

0xC0FF06BDL

RPCS_E_RULESSERVER_CALL_FAILED

Error occurred while checking security rules.

SrvWrapper Errors

0xC1050457L

SRVWRAPPER_E_SERVER_NOT_FOUND

The user could not be authenticated.

The error could occur due to:

1. Authenticore server was not found.

2. The authentication method is not supported by available Authenticore servers (required BSP module is missing on server).

3. Lost communication with Domain Controller.

4. The required subsystem was not installed.

0xC1050458L

SRVWRAPPER_LOG_E_SERVER_NOT_FOUND

The user could not be authenticated.

The error could occur due to:

1. Authenticore server was not found

2. The authentication method is not supported by available Authenticore servers (there is no required BSP module on server).

3. Lost communication with Domain Controller.

4. The required subsystem was not installed.

0xC105045CL

SRVWRAPPER_E_LOCAL_USER

Either user account or authenticator is invalid.

0xC105045DL

SRVWRAPPER_E_NOT_BIOUSER
Authentication Failed. Press OK to try again.

0xC1050466L
SRVWRAPPER_E_CACHE_USED
Authenticore server not found.
User could not be logged in using authenticator from cache.

Password Filter Errors

0xC104057AL
PWDFILT_E_PASSWORD_SET_FAILED
Error while resetting password for user.

0xC104058BL
PWDFILT_E_PASSWORD_CHANGE_FAILED
Error while changing password for user.

Password Manager Errors

0xC1080585L
PWDMGR_E_ERROR_OCCURED
An error occurred during Password Manager work.

0xC1080586L
PWDMGR_E_CHANGE_PASSWORD_FAILED
Could not change password for user.
It is recommended to check "Minimal password age" domain setting. In case its value differs from 0, it is possible that password change can be denied because the password has been already changed within the specified time interval.
Also, password cannot be changed in case "User cannot change password" account setting is enabled.

0xC1080587L
PWDMGR_E_BAD_START_TIME
The time period specified using command prompt had expired before Password Manager was started. The service has been stopped.

EventLog Errors

0xC10705DCL
LOG_E_CANT_WRITE_REMOTE_LOG
Could not get access to remote Log Server.
There is either no Log Server, it was turned off, or being reloaded. In case the error persists, it is recommended to check Firewall settings and the correctness of the domain names permission.

BioAPI Errors

0xC1010000L
BIO_E_INITIALIZE
Could not initialize BioAPI framework.

0xC1010001L
BIO_E_LOAD_MODULE
Could not load the required BioAPI BSP module.

0xC1010002L
BIO_E_ENROLL
Could not get enrolled authenticator.

0xC1010003L
BIO_E_IDENTIFY
Could not get authenticator.

0xC1010004L
BIO_E_VERIFY
Could not compare user's authenticators.

0xC1010005L
BIO_E_DATA_CORRUPTED
Could not load authenticators from the memory. Data is corrupt.

0xC1010006L
BIO_E_COMPARE_BSP_MISMATCH
The type of enrolled authenticator does not correspond to the type of the given authenticator.

0xC1010007L
BIO_E_COMPARE_DATA_MISMATCH
Authenticator does not correspond to the enrolled authenticator.

Authenticore Server Errors

0xC1000000L
LOGON_E_CREATE_TEMPLATE
Could not create authenticator. The list of user authenticators may be corrupt.

0xC1000001L
LOGON_E_LOAD_TEMPLATE
Could not load the authenticator. The list of user authenticators may be corrupt.

0xC1000002L
LOGON_E_READ_COLLECTION
Could not read user authenticators list.

0xC1000003L
LOGON_E_WRONG_PASSWORD
Either user account or password value is invalid.

0xC1000004L
LOGON_E_WRONG_AUTHENTICATOR
Authentication Failed. Press OK to try again.

0xC1000005L
LOGON_E_CANNOT_LOGON
Authentication Failed. Press OK to try again.

0xC1000006L
LOGON_E_OPERATION_DENIED
This operation is forbidden by administrator.

0xC1000007L
LOGON_E_TOO_MANY_AUTHENTICATORS
The allowed amount of authenticators is exceeded.

0xC1000008L
LOGON_E_SERVER_NOT_FOUND
Could not set connection with the Authenticore server.
Check network connection and try again. If the error persists please contact your system administrator.

0xC1000009L
USER_E_CHANGE_PASSWORD_INVALID
The passwords were unsynchronized.

0xC100000AL
USER_E_CHANGE_PASSWORD_POLICY
Could not change password for the user. The generated value does not satisfy the security policies. It is recommended to check "Minimal password age" domain setting. In case its value differs from 0, the password change can be denied because the password has been already changed within the specified time interval.

0xC100000BL
USER_E_CHANGE_PASSWORD_ACCESS_DENIED
Could not change user password. The current security settings forbid the user to change his/her password.

0xC100000CL
USER_E_CHANGE_PASSWORD
Could not change password for the user. The reason is unknown.

0xC100000DL
LOGON_E_WRONG_DATE
Time interval from the moment the user authenticator was obtained and the moment it was delivered to the Authenticore server exceeds the value of the settings, which regulates authenticator validity period (5 minutes by default).
This error can occur as a result of either system time desynchronization between user computer and Authenticore server or criminal attempt to use authenticator intercepted over network.

0xC100000EL
LOGON_E_LOAD_BSP
Could not load BioAPI BSP module. Either the required BSP module is not installed on the Authenticore server or it failed to load. The system will attempt to authenticate on another Authenticore server.

0xC100000FL
CHANGEPWD_OUT_OF_RESOURCES
System resources are not enough to change password for the user.

0xC10006BEL
LOGON_E_LOGON_REFUSED_BY_RULES
Logon refused by security rules.

0xC10006BFL
LOGON_E_DENY_LOGON_BY_PASSWORD
Logon by password was denied.

Authentication Providers Errors

0xC1020000L
PROV_E_NO_USER
The user was not found.

0xC1020001L
PROV_E_ACCESS_DATA
Could not get access to user data.

0xC1020002L
PROV_E_PROPERTY_NOT_FOUND
The property was not found. Perhaps the Active Directory scheme is not extended by additional attributes.

0xC1020003L
PROV_E_AREADY_CREATED
User is already allowed to use authenticators.

0xC1020004L
PROV_E_CREATE_ENUMERATOR
Could not create users sorting object.

0xC1020005L
PROV_E_SEARCH_USER
Could not start user search.

0xC1020006L
PROV_E_ACCESS_DENIED
Access is denied. Not enough permissions.

0xC1020009L
PROV_E_AD_OBJECT_NOT_BIND
Unable to get object data in AD.

0xC102000AL
PROV_E_ADAM_OBJECT_NOT_BIND
Unable to get object data in ADAM.

0xC102000BL

PROV_E_ADAM_NOT_OPERATIONAL

Could not get access to ADAM server.

Cryptography Errors

0xC1030001L

CRYPT_E_USER_DATA_CORRUPTED

User data corrupted.

0xC1030002L

CRYPT_E_VERIFY_SIGNATURE

Either user data or the Enterprise Key is corrupt.

0xC1030003L

CRYPT_E_INIT_PROVIDER

Could not initialize required Crypto Service Provider (CSP).

0xC1030004L

CRYPT_E_GENERATE_OR_EXPORT_KEYS

Could not generate or export cryptographic keys.

0xC1030005L

CRYPT_E_IMPORT_KEYS

Could not import cryptographic keys.

0xC1030006L

CRYPT_E_DATA_CORRUPTED

Data is corrupted.

Manager Errors

0x01060001L

MGR_S_LAST_TEMPLATES_REMOVED

Several authenticators were deleted because the allowed amount of authenticators was reduced.

0xC1060002L

MGR_E_LOGON_DOMAIN_REDIRECTION_OP_UNSUPPORTED

The operation is not supported while the domain redirection policy is enabled.

Plugins Errors

0xC1090000L
PLUGIN_E_NOT_REGISTRED
The specified Plug-in is not registered on the server.

0xC1090001L
PLUGIN_E_CANNOT_CREATE
Could not create registered Addon.

0xC1090002L
PLUGIN_E_USER_NOT_TRUSTED
The user was authenticated by password.

0xC1090003L
PLUGIN_E_OPERATION_DENIED
The operation is forbidden.

Licensing Errors

0xC10A0001L
LIC_E_INVALID_FORMAT
Invalid format of license data.

0xC10A0002L
LIC_E_LICENSE_NOT_FOUND
License not found.

0xC10A0003L
LIC_E_LICENSE_STORAGE_CORRUPTED
License storage data is corrupted.

0xC10A0004L
LIC_E_LICENSE_CORRUPTED
License data was changed or corrupted.

0xC10A0005L
LIC_E_RESTRICTIONS_ERROR
Your license does not match the time period restriction, the product version restriction or the domain name is wrong.

0xC10A0006L

LIC_E_PUBLICKEY_CORRUPTED

Cannot validate digital signature of the license. Certificate may be missing or corrupt.

0xC10A0007L

LIC_E_PLUGIN_DOESNT_SUPPORT_LICENSING

This Addon does not support licensing.

0xC10A0008L

LIC_E_START_LIMIT_ERROR

The actual number of installed Authenticore Servers exceeds the number allowed by the License.

0xC10A0009L

LIC_E_USERS_LIMIT_ERROR

Actual number of NetIQ-enabled accounts exceeds the number allowed by the License.

0xC10A000AL

LIC_E_DOWNGRADE

The license you are trying to add allows fewer number of licensed objects than you have now.

Backup Provider Errors

0xC10C0001L

BACKUPPROV_E_BAD_PASSWORD_OR_DATA

Bad password or data corrupted.

Administration Tools Errors

0xC10D0001L

ADMTOOLS_E_NOT_MLADMIN

You don't have rights for changing settings on this page. Please, ensure that you are the member of the NetIQ Admins group and these rights are delegated to the NetIQ Admins group.

0xC10D0002L

ADMTOOLS_E_NO_RIGHTS

You don't have rights for changing settings on this page. Please, ensure that these rights are delegated to you.

GINA Errors

0xC10B0645L
GINA_E_LOGON_BY_PASSWORD_FAILED
Could not authenticate the user by the entered password.
The error could also occur if the entered account was invalid.

Data Errors

0xC10E0001L
DATA_E_FIELD_NOT_SET
The field value is not set.

0xC10E0002L
DATA_E_VALUE_NOT_SET
The subfield value is not set.

0xC10E0003L
DATA_E_SUBSYSTEM_NOT_FOUND
Subsystem is not found.

0xC10E0004L
DATA_E_ACCESS_DENIED
Data access denied.

0xC10E0005L
DATA_E_RECORD_NOT_FOUND
Record is not found.

0xC10E0006L
DATA_E_USER_NOT_TRUSTED
The user was authenticated by password.

0xC10E0007L
DATA_E_INVALID_FIELD_NAME
Invalid field name.

0xC10E0008L
DATA_E_BAD_SCHEME_SIGNATURE
Bad schema signature.

0xC10E0009L
DATA_E_USERS_LICENSE_NOT_FOUND
Subsystem users license is not found.

0xC10E000AL
DATA_E_BASE_LICENSE_NOT_FOUND
Subsystem servers license is not found.

0xC10E000BL
DATA_E_NOT_SUBSYSTEM_USER
User is not using given subsystem.

0xC10E000CL
DATA_E_USERS_LICENSE_LIMIT_ERROR
Actual number of the subsystem-enabled accounts exceeds the number allowed by the License.

0xC10E06A5L
DATA_E_ADMIN_GET_DATA_FAILED
Unable to get the subsystem data for the user.

0xC10E06A6L
DATA_E_USER_GET_DATA_FAILED
User is unable to get the subsystem data.

0xC10E06A7L
DATA_E_ADMIN_FAILED_TO_ALLOW_TO_USE_SS
Unable to make user the client of the subsystem.

0xC10E06A9L
DATA_E_USER_FAILED_TO_BE_SS_CLENT
User failed to be a client of the subsystem.

0xC10E06ADL
DATA_E_ADMIN_CHANGE_DATA_FAILED
Unable to change the subsystem data for the user.

0xC10E06AFL
DATA_E_USER_CHANGE_DATA_FAILED
User is unable to change the subsystem data.

0xC10E06B1L
DATA_E_RESET_PASSWORD

The password was reset for user. Could not reset special data for subsystem.

0xC10E06B3L
DATA_E_ADMIN_REMOVE_SS_DATA
Unable to deny user to use the subsystem.

0xC10E06B5L
DATA_E_USER_REMOVE_SS_DATA
User was unable to quite using the subsystem.

0xC10E06B7L
DATA_E_RESET_DATA
The password was reset for user. Could not reset special data for subsystem.

0xC10E06B8L
DATA_E_RESET_DATA_FULL_RESET
The password was reset for user. Could not reset special data for subsystem. The subsystem data was reset completely.

0xC10E06BAL
DATA_E_SUBSYSTEM_LIST_INVALID_COMMON
The subsystems list for user is invalid and was cleared.

0xC10E06BBL
DATA_E_SUBSYSTEM_LIST_INVALID_SS
The subsystems list for user is invalid and was cleared.

0xC10E06C0L
DATA_E_CONTAINER_NOT_FOUND
Data container is not defined in the schema.

Troubleshooting

This chapter provides solutions for known issues. If you encounter any problems that are not mentioned here, please contact the support service.

Authentication Failed

Description:

Authentication using Web Service has failed.

Cause:

Web Service is installed separately from Authenticore Server and authentication providers are not installed on Web Service.

Solution:

In the case if Web Service is installed separately from Authenticore Server, it is necessary to install authentication providers on Web Service.

Index

OTP 11, 14, 18, 20