SafeNet Authentication Service Authentication API for Microsoft .NET

Developer Guide

gemalto
security to be free

All information herein is either public information or is the property of and owned solely by Gemalto NV. and/ or its subsidiaries who shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information.

Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/ or industrial property rights of or concerning any of Gemalto's information.

This document can be used for informational, non-commercial, internal and personal use only provided that:

- The copyright notice below, the confidentiality and proprietary legend and this full warning notice appear in all copies.
- This document shall not be posted on any network computer or broadcast in any media and no modification of any part of this document shall be made.

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided "AS IS" without any warranty of any kind. Unless otherwise expressly agreed in writing, Gemalto makes no warranty as to the value or accuracy of information contained herein.

The document could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Furthermore, Gemalto reserves the right to make any change or improvement in the specifications data, information, and the like described herein, at any time.

Gemalto hereby disclaims all warranties and conditions with regard to the information contained herein, including all implied warranties of merchantability, fitness for a particular purpose, title and non-infringement. In no event shall Gemalto be liable, whether in contract, tort or otherwise, for any indirect, special or consequential damages or any damages whatsoever including but not limited to damages resulting from loss of use, data, profits, revenues, or customers, arising out of or in connection with the use or performance of information contained in this document.

Gemalto does not and shall not warrant that this product will be resistant to all possible attacks and shall not incur, and disclaims, any liability in this respect. Even if each product is compliant with current security standards in force on the date of their design, security mechanisms' resistance necessarily evolves according to the state of the art in security and notably under the emergence of new attacks. Under no circumstances, shall Gemalto be held liable for any third party actions and in particular in case of any successful attack against systems or equipment incorporating Gemalto products. Gemalto disclaims any liability with respect to security for direct, indirect, incidental or consequential damages that result from any use of its products. It is further stressed that independent testing and verification by the person using the product is particularly encouraged, especially in any application in which defective, incorrect or insecure functioning could result in damage to persons or property, denial of service or loss of privacy.

© 2023 Gemalto All rights reserved. Gemalto and the Gemalto logo are trademarks and service marks of Gemalto N.V. and/ or its subsidiaries and are registered in certain countries. All other trademarks and service marks, whether registered or not in specific countries, are the property of their respective owners.

Document Part Number: 007-012398-002, Rev. K

Release Date: February 2023

Table of Contents

1 Preface 5
Audience 5
Related Documents 5
Sample Code Usage 5
Support Contacts 6
Customer Support Portal 6
Telephone Support 6

2 Introduction 8
Applicability 8
Prerequisites 8
Security Notes 8

3 Installation and Upgrade 10
Installing Authentication API for Microsoft .NET 10
Upgrading Authentication API for Microsoft .NET 14

4 API C# Class 15
Constructors 15
Methods 15
Authenticate Method 15
VerifySignature Method 17
CheckServerStatus Method 18
getGridSureImageAsBase64 Method 18
getGridSureImage Method 19
SSL Server Certificate Validation 20

5 Manager User Interface 21
Authentication 22
Verify Signature 22
Authentication Server Test 22

6 Use Case Scenarios 23
Basic Authentication 23
Challenge-Response Authentication 23
Outer Window Authentication 24
PIN Authentication 24
User-Changeable PIN Stored on Server 25
Server-Changeable PIN Stored on Server 25
Static Password Authentication 25

7 Agent Key File and Additional Deployment 26

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Agent Key File....26
Loading Key File....26
Registering Key File Certificate....27
Logging....27
API Example....27
Deploying on Additional Computers....28

8 Troubleshooting....29
System.Exception:'Empty Path name is not legal'....29
Possible causes....29
Solution....29

Preface

Audience

This document is intended for personnel responsible for maintaining your organization's security infrastructure.

All products manufactured and distributed by Gemalto are designed to be installed, operated, and maintained by personnel who have the knowledge, training, and qualifications required to safely perform the tasks assigned to them. The information, processes, and procedures contained in this document are intended for use by trained and qualified personnel only.

Related Documents

The following document contains related information:

- *SafeNet Authentication Service Authentication API for Microsoft .NET 1.3.0: Customer Release Notes*

Sample Code Usage

Sample codes are provided for demonstration (and test) purposes and must never be used in the production environment. For any damage arising out of such use, Gemalto shall not be liable, whether in contract, tort or otherwise. We do not claim the copyright to certain sample codes, as they may belong to (related / unrelated) third-parties.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Support Contacts

If you encounter a problem while installing, registering, or operating this product, refer the documentation. If you cannot resolve the issue, contact your supplier or **Gemalto Customer Support**.

Gemalto Customer Support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between Gemalto and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Customer Support Portal

The Customer Support Portal, at **https://supportportal.gemalto.com**, is a where you can find solutions for most common problems. The Customer Support Portal is a comprehensive, fully searchable database of support resources, including software and firmware downloads, release notes listing known problems and workarounds, a knowledge base, FAQs, product documentation, technical notes, and more. You can also use the portal to create and manage support cases.

NOTE: You require an account to access the Customer Support Portal. To create a new account, go to the portal and click on the REGISTER link.

Telephone Support

If you have an urgent problem, or cannot access the Customer Support Portal, you can contact Customer Support by telephone. Calls to Customer Support are handled on a priority basis.

Region	**Telephone number** (Subject to change. An up-to-date list is maintained on the Customer Support Portal)
Global	+1-410-931-7520
Australia	1800.020.183
China	North: 10800-713-1971 South: 10800-1301-932
France	0800-912-857
Germany	0800-181-6374
India	000.800.100.4290
Israel	180-931-5798
Italy	800-786-421
Japan	0066 3382 1699

Korea	+82 2 3429 1055
Netherlands	0800.022.2996
New Zealand	0800.440.359
Portugal	800.863.499
Singapore	800.1302.029
Spain	900.938.717
Sweden	020.791.028
Switzerland	0800.564.849
United Kingdom	0800.056.3158
United States	(800) 545-6608

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

1
Introduction

This document describes the SafeNet Authentication Service (SAS) Authentication API for Microsoft .NET that allow agents to support all the functions required to interact with the SAS. A SAS agent is essentially a third-party application having embedded plug-in code that passes the collected User Names and One-Time Passwords (OTPs) to the SAS for authentication.

Applicability

The information in this document applies to the following:

- **SafeNet Authentication Service - Cloud (SAS Cloud)** — The SafeNet's cloud-based authentication service.
- **SafeNet Authentication Service - Service Provider Edition (SAS SPE)** — The on-premises, server version targeted at service providers interested in hosting SAS in their data center(s).
- **SafeNet Authentication Service - Private Cloud Edition (SAS PCE)** — The on-premises, server version targeted at organizations interested in hosting SAS in their private cloud environment.

Prerequisites

Microsoft .NET v4.6.1 Framework (or later).

NOTE: .Net 6 (**APITestClientDotNetCore** application) is only supported on 64-bit operating system.

Security Notes

To enhance security, we strongly recommend the following actions.

Subject	Action	More Details
Installation Folder	During installation, change the default destination folder to a system-protected folder accessible only by an account with management (read) privileges and by the account that initiates the API call.	See **Installation** (Step **5**)
SSL Server Certificate Validation	Activate SSL server certificate validation.	See **SSL Server Certificate Validation**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Subject	Action	More Details
Agent Key File	Place the Key File in a system-protected folder accessible only by an account with management (read) privileges and by the account that initiates the API call.	See **Loading Key File**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

2
Installation and Upgrade

Installing Authentication API for Microsoft .NET

1. Run the required installation file:

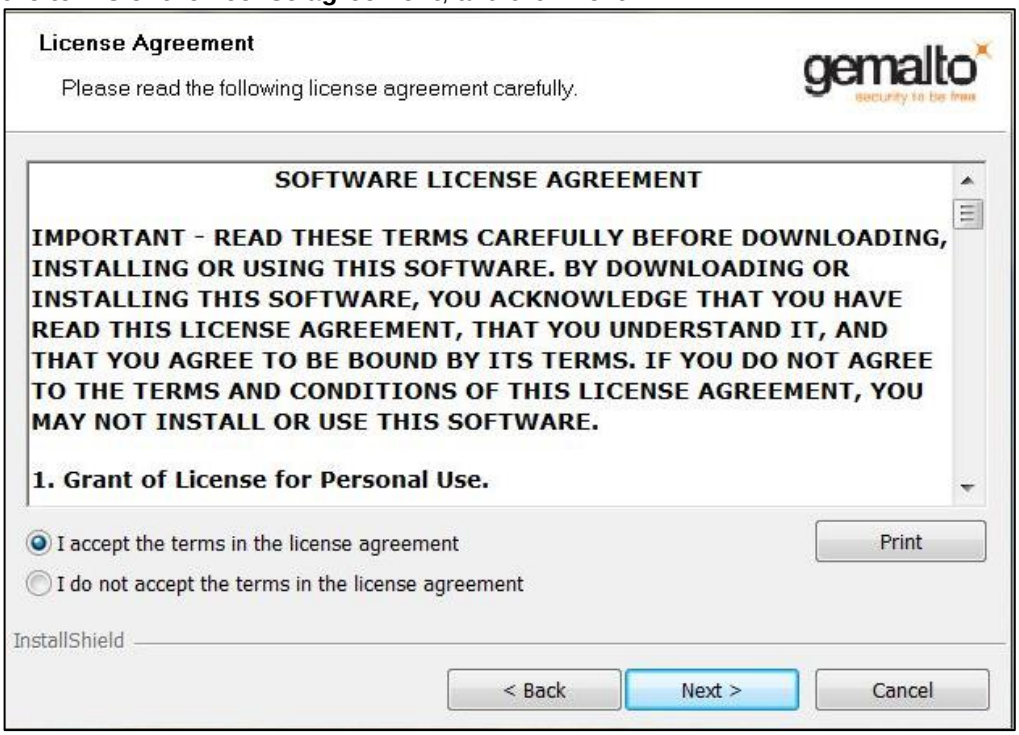
   32-bit: SafeNet Authentication Service .NET Authentication API x86.exe

   64-bit: SafeNet Authentication Service .NET Authentication API x64.exe

2. On the **Welcome to the InstallShield Wizard for SafeNet Authentication Service .NET Authentication API** window, click **Next**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

3. On the **License Agreement window**, read the software license agreement and to proceed, select **I accept the terms of the license agreement**, and click **Next**.



4. On the **Customer Information** window, perform the following steps:
   a. In the **User Name** field, enter your user name.
   b. In the **Organization** field, enter the name of your organization (any names can be used).
   c. Click **Next**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

5. On the **Destination Folder** window, perform one of the following steps:
   - To change the installation folder, click **Change** and navigate to the required folder, and then click **Next**.
   - To accept the default installation folder as displayed, click **Next**.

> **NOTE:** We strongly recommend installing the application in a system protected folder accessible only by an account with management (read) privileges and by the account that initiates the API call.
>
> The default location is in the **Program Files** folder, which is not read protected. This means that the location must be changed to a protected location that cannot be read by non-administrative accounts.

6. On the **Authentication Service Setup** window, complete the following fields and click **Next**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Location	Enter the hostname or IP address of the primary SAS server.
Connect using SSL (requires valid certificate)	Select to use the SSL. This option requires installation of a valid certificate on the NPS server.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

7. On the **Ready to Install the Program** window, click **Install**.

8. Once the installation is completed, the **InstallShield Wizard Completed** window is displayed. Click **Finish** to exit the installation wizard.

Upgrading Authentication API for Microsoft .NET

Upgrade to SAS Authentication API for Microsoft .NET 1.3.0 is not supported.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

3
API C# Class

The SAS authentication API is represented by a single C# class named, **`BSIDAPI`**.

This class includes the following:

- A default constructor that loads its configuration information from a default Registry location.
- An alternate constructor that allows the user to define an alternate location in the Registry from which to load its configuration.

The API can be configured manually or through the **Manager User Interface**.

Constructors

The following constructor is used to read the registry key defined by **`ConnectionInformation`** string.

```
BSIDAPI()
{
// reads registry SOFTWARE\CRYPTOCard\BlackShield ID\BSIDAPI
}
BSIDAPI(string ConnectionInformation)
{
// reads registry key defined by ConnectionInformation
}
```

NOTE: By using the connection information in the constructor, an agent may specify one or more SAS servers to implement failover authentications. The connection information for each SAS server can be stored in its own key.

Methods

Authenticate Method

All actions related to authentication make use of a single API call:

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

```
Authenticate
(
    string user,
    string org,
    string ipaddress,
    string passcode,
    ref string challenge,
    ref string state
);
```

where,

- **user** – A string representing the user name of the individual who is authenticating.
- **org** – A string representing the organization to which the individual who is authenticating, belongs. This should be passed as an empty string to represent the default organization.
- **ipaddress** – A string representing the IP address from which the authentication request originated. If this parameter is an empty string, the SAS will attempt to auto-detect it.
- **passcode** – A string representing the user's passcode, in one of the following formats:
    - **[PIN+OTP]** – For server-side PIN authentication
    - **[OTP]** – Token-side PIN or no PIN
    - **[PIN]** – When responding to a server-side change request for a user-changeable PIN
    - **[StaticPassword]** – User has a static password enabled or is responding to a static password change
    - **[null]** – Indicates that a challenge is required
- **challenge** – A string passed by reference that may be populated with a challenge/ PIN change/ outer window authentication message.
- **state** – A string passed by reference that contains a state attribute. When returning a challenge, the same state must be passed back to the server.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

The API call returns the following values:

- int
  - 0 – Authentication Failed
  - 1 – Authentication Succeeded
  - 2 – Challenge
  - 3 – Server provided PIN
  - 4 – User needs to provide PIN
  - 5 – Authentication in outer window. Re-authenticate.
  - 6 – User must change their static password.
  - 7 – Static password change does not satisfy policies.
  - 8 - PIN provided doesn't meet requirements. Please provide a new PIN.

VerifySignature Method

This method verifies the token's signature for a given hash.

```
VerifySignature
(
      string SerialNumber,
      string Hash,
      string Signature
);
```

where,

- **SerialNumber** – A string representing the token's serial number.
- **Hash** – A string representing the hash value for the signature verification.
- **Signature** – A string representing the signature to be verified for the given hash.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

The method returns the following values:

- int
  - 0 – Signature is incorrect for the given hash.
  - 1 – Signature is correct for the given hash.

CheckServerStatus Method

This method monitors the operational status of the SAS authentication server.

```
CheckServerStatus
(
      // void
);
```

The method returns the following values:

- int
  - 0 – Server is down
  - 1 – Server is operational

NOTE: Use the **CheckServerStatus** method to monitor health of the SAS and to determine when to initiate failover to a secondary server.

getGridSureImageAsBase64 Method

This method creates a Base64 encoded Bitmap (BMP) image based on the incoming challenge parameter.

```
getGridSureImageAsBase64
(
string inChallenge,
ref string outBase64EncodedBitmap,
ref string outErrorMessage
);
```

where,

- inChallenge – A string representing a GrID challenge returned from the Authenticate method of length 25, 36, 47, or 64.
- outBase64EncodedBitmap – An output string representing the Base64 encoded bitmap GrID image.
- outErrorMessage – An output string containing a detailed error message.

The method returns the following values:

- bool
  - false – an error has occurred. Check *outErrorMessage* for details.
  - success – *outBase64EncodedBitmap* will contain encoded image.

getGridSureImage Method

This method creates a Bitmap (BMP) image based on the incoming challenge parameter.

```
getGridSureImage
(
string inChallenge,
ref System.Drawing.Image outBitmap,
ref string outErrorMessage
);
```

where,

- inChallenge – A string representing a GrID challenge returned from the Authenticate method of length 25, 36, 47, or 64.
- outBitmap – An output string representing the Bitmap GrID image.
- outErrorMessage – An output string containing a detailed error message.

The method returns the following values:

- bool
  - false – an error has occurred. Check *outErrorMessage* for details.
  - success – *outBitmap* will contain the GrID image.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

SSL Server Certificate Validation

The SSL server certificate validation is not activated by default, when using the API.

Activate SSL using the **`ServerCertificateValidationCallback`** property, as done in the following sample:

```
ServicePointManager.ServerCertificateValidationCallback = {delegate
function}
```

NOTES:

- We strongly recommend using SSL server certificate validation.
- The SSL setting influences all SSL connections created by the application (and not only SSL connections created directly by the API).

For more details, visit the following links:

- *https://msdn.microsoft.com/en-us/library/ms144153%28v=vs.110%29.aspx*
- *https://msdn.microsoft.com/en-us/library/system.net.security.remotecertificatevalidationcallback%28v=vs.110%29.aspx*

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

4
Manager User Interface

The Manager User Interface is used to verify the authentication function. It is defined on the following two frameworks:

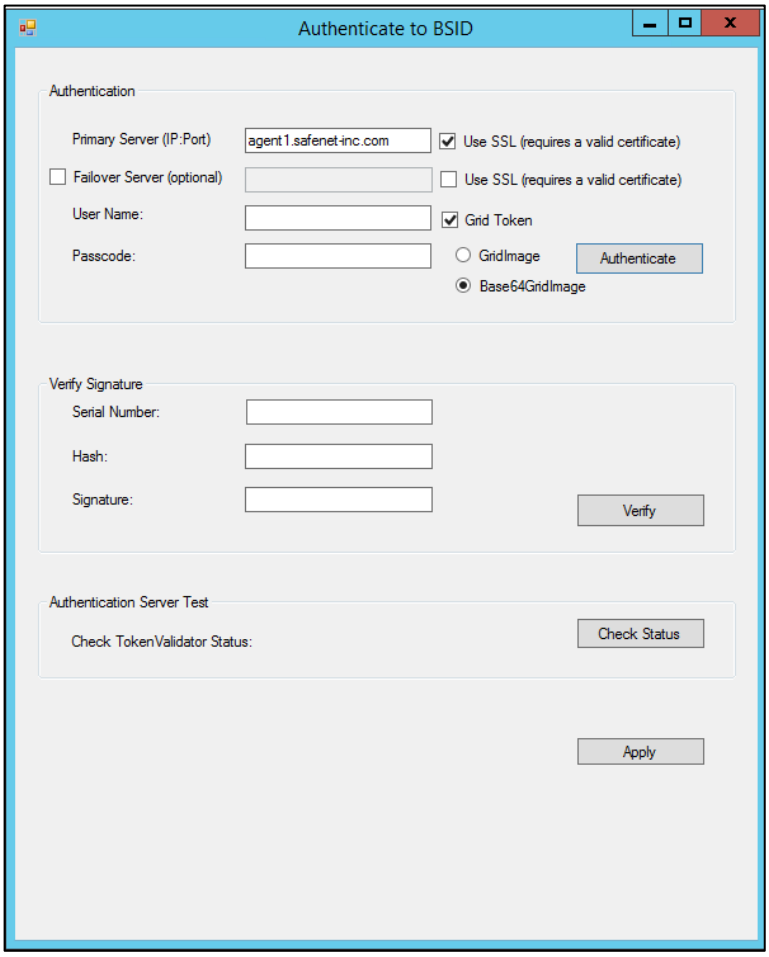
- .Net Framework 4.6.1 (**apiTestClient**)

  To open the Manager User Interface, navigate to the following path and run the **apiTestClient.exe** file:
  `Gemalto\DotNetAPI\apiTestClient\bin\Release`

- .Net 6 (**apiTestClientDotNetCore**)

  To open the Manager User Interface, navigate to the following path and run the **APITestClientDotNetCore.exe** file:
  `Gemalto\DotNetAPI\APITestClientDotNetCore\bin\Release`

The Manager User Interface window will be displayed:



Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Authentication

To connect primary and failover server(s), perform the following steps:

1. Enter the following fields:
   - **Primary Server (IP:Port)** - (Select **Use SSL**, if required)
   - **Failover server (optional)** - (Select **Use SSL**, if required)
2. Click **Apply**.
3. The **ServiceURL** and **OptionalSecondaryServiceURL** keys are updated under the following registry node: **`HKEY_LOCAL_MACHINE\SOFTWARE\CRYPTOCard\BlackShield ID\BSIDAPI`**

To verify authentication, perform the following steps:

1. Enter the following fields:
   - **User Name**
   - **Password**
   - **Grid Token** fields

   NOTE: Grid Token fields (***GridImage*** and ***Base64GridImage***) are added to test the *getGridSureImage* or *getGridSureImageAsBase64* methods conveniently.
2. Click **Authenticate**.

   A message is displayed indicating if authentication succeeded or failed.

Verify Signature

To verify signature, perform the following steps:

1. Enter the following fields:
   - **Serial Number**
   - **Hash**
   - **Signature**
2. Click **Verify**.

   A message is displayed indicating if signature verification succeeded or failed.

Authentication Server Test

To test **TokenValidator** status, click **Check Status**.

A message will be displayed indicating if the SAS server is running or not.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

5
Use Case Scenarios

The SAS architecture supports the use of a token-side or a server-side PIN in either **QuickLog** or **Challenge-Response** **mode**. In addition, the application using the API must support challenges, inner/ outer window authentication, and static password authentication. The following sections discuss these features in detail.

LOCALIZATION NOTE: To support localization, SAS returns only necessary data in its challenge messages. The application is required to construct a localized version of it, to display to the client.

For example, SAS would return only **19863257**, but the application would display, **Please respond to the challenge: 19863257**.

Mode: Tokens can operate in either **Challenge-Response** or **QuickLog** mode. **QuickLog** mode is recommended because it simplifies the logon experience (and strengthens security) by eliminating the requirement to enter a challenge into a token to get an OTP. **QuickLog** do not rely on time to remain synchronized with the server. Instead, each time an event-based token is activated, a new token code is generated.

Basic Authentication

The communication between the application and the server uses challenge messages and states, similar to the RADIUS protocol. The following scenario shows the most basic interaction between the application and the server:

1. The application issues an authentication request that includes the user name, organization name, and a passcode.
2. The server responds with one of nine possible return codes. See **Methods**
3. **Authenticate** Method.

Challenge-Response Authentication

The challenge message and state attribute issued from the authenticating server are central to the concept of challenge-response authentication, outer window authentication, and server-side PIN changes. This mechanism is employed to authenticate tokens in challenge-response mode in the following manner:

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

1. The application issues an authentication request that includes the user name, organization name, and an empty passcode.
2. The server responds with a challenge message containing a challenge string.

   For example, **Challenge: 19863257**, and a state attribute.

   See **LOCALIZATION NOTE**.
3. The authenticating application responds to the challenge by issuing another authentication request that includes the same user name, organization name, a response, and the state attribute.

Outer Window Authentication

User authentication through inner/ outer window authentication uses challenge messages and state attributes, similar to the **Challenge-Response Authentication** scenario. In outer window authentication, users provide a match in a large look-ahead window, and respond to a follow-up challenge by providing the exact next OTP from their token. The follow sequence illustrates the process:

1. The application issues an authentication request that includes the user name, organization name, and a passcode.
2. The server finds a match for the provided OTP in the outer window, and then issues a challenge to the client containing an outer window authentication string, for example: **Please re-authenticate using the next OTP from your token**, and a state attribute.

   See **LOCALIZATION NOTE**.
3. The authenticating application responds to the challenge by issuing another authentication request that includes the same user name, organization name, a response, and the state attribute.

PIN Authentication

SAS supports several PIN types:

- No PIN
- Fixed PIN (token-side PIN validation)
- User-changeable PIN (token-side PIN validation)
- Fixed PIN stored on server
- User-changeable PIN stored on server
- Server-changeable PIN stored on server

The SAS authentication mechanism supports incoming passcodes in the following formats:

- [PIN+OTP]
- [OTP]
- [NEWPIN]
- [StaticPassword]
- [null] - empty passcode to request a challenge

PINs stored on the server can be user- or server-changeable. To accommodate this, leverage the challenge framework in the following manner:

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

User-Changeable PIN Stored on Server

1. The application issues an authentication request that includes the user name, organization name, and a passcode.
2. The server finds a match for the provided OTP and determines that the PIN must be changed.
3. The server issues a challenge to the client containing a PIN change string, for example, **Your PIN has expired. Please enter a new PIN** and a state attribute.

   See **LOCALIZATION NOTE**.
4. The authenticating application responds to the challenge by returning a new PIN and the state attribute.

Server-Changeable PIN Stored on Server

1. The application issues an authentication request that includes the user name, organization name, and a passcode.
2. The server finds a match for the provided OTP and determines that the PIN must be changed.
3. The server issues a challenge to the client containing a PIN change string, for example, **Your new PIN is 628. Please re-authenticate using this new PIN and your next passcode** and a state attribute.

   See **LOCALIZATION NOTE**.
4. The authenticating application responds to the challenge by issuing another authentication request that includes the user name, organization name, the new PIN and OTP, and the state attribute.

Static Password Authentication

SAS offers the option of static password authentication, including, enabling the user to change the password. The challenge-response architecture can be used in the following manner:

1. The application issues an authentication request that includes the user name, organization name, and a static password.
2. If the user is not required to change the password and it is correct, the server returns access-accept.
3. If the user is required to change the password, a challenge message is issued to the client, for example, **Your password has expired. Please enter a new password** and a state attribute.

   See **LOCALIZATION NOTE**.
4. If a challenge message has been issued in step **3**, the authenticating application responds to the challenge by issuing an authentication request that includes the user name, organization name, the new static password, and the state attribute.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

6
Agent Key File and Additional Deployment

Agent Key File

The SAS API uses an encrypted key file to secure communication with the server. The key file is loaded and registered with the agent, and a matching key is registered with the authentication server.

A sample key file (**`default.bsidkey`**) is installed for evaluation purposes. This file is publicly distributed, and should be used for evaluation purposes only. It is strongly recommended that you generate your own key file for your production environment. To use an encrypted key file, load a key file, and register its certificate as detailed below.

Loading Key File

1. Open the SAS **COMMS** tab and download the agent key file from **Authentication Agent Settings** section.
2. Copy downloaded file to the following path:

   `[INSTALLDIR]\KeyFile\`

   where, **[INSTALLDIR]** is the installation directory of the API.

NOTE: We strongly recommend placing the file in a system protected folder accessible only by an account with management (read) privileges and by the account that initiates the API call.

The default location is in the **Program Files** folder, which is not read protected. This means that the location must be changed to one that is a protected location that cannot be read by non-administrative accounts.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

Registering Key File Certificate

1. Open the Registry Editor, and expand the Registry to the following path:
   My Computer > **HKEY_LOCAL_MACHINE** > **SOFTWARE** > **CRYPTOCARD** > **BLACKSHIELD ID** > **BSIDAPI**

> **NOTE:** If a custom Registry key location passes the connection information to the **BSIDAPI** constructor, expand the Registry to the custom Registry key instead, and update its **EncryptionKeyFile** information.

2. Edit the **EncryptionKeyFile** value so that it contains the fully qualified path to the agent key file that was loaded in **Loading Key File**, and save the changes.

Logging

If unexpected behavior occurs, view the log messages. The SAS API logs error messages to the Windows Event Viewer's Application log. The source appears as **BSIDAPI**.

API Example

Either of the following two ways can be used to consume the BSIDAPI SDK. Both the **apiTestClient** and **APITestClientDotNetCore** directory includes an example of how to do the following tasks:

1. DLL based approach

In this approach, a reference to the BSIDAPI.dll is added along with its dependent DLLs, which are installed using the powershell script (**Profile.ps1**) provided in the agent package.

- Add a reference to the **BSIDAPI.dll**.
- Now, install the dependent DLLs. Perform the following steps to execute the powershell script file:
  - Navigate to the agent package. Copy **Profile.ps1** script file from the **Scripts** file.
  - Place the script file at the following path: **C:\Users\{Username}\Documents\WindowsPowerShell\.**
  - Open the Visual Studio IDE and navigate to **Tools > Nuget Package Manager > Package Manager Console**.
  - In the Package Manager Console window, type **Install-Packages** and then press **Enter**.
- Use the **BSIDAPI** class to define connection information.
- Call the **Authenticate** and **CheckServerStatus** methods.

2. Nuget package approach

In this approach, a reference to the nuget package (**BSIDAPI.1.0.0.nupkg**) is added. Perform the following steps:

- Navigate to the agent package. Copy **BSIDAPI.1.0.0.nupkg** file.
- Place the file locally on your machine.
- Open the Visual Studio IDE and navigate to **Tools > Nuget Package Manager > Package Sources**. Add the file as a package source and then click **OK**.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

- Use the **BSIDAPI** class to define connection information.
- Call the **Authenticate** and **CheckServerStatus** methods.

Deploying on Additional Computers

To deploy the completed application on an additional computer, perform the following steps:

1. Copy the **BSIDAPI** Registry key from the following path (of the system running the completed application):
   My Computer > **HKEY_LOCAL_MACHINE** > **SOFTWARE** > **CRYPTOCARD** > **BLACKSHIELD ID**

   NOTE: If a custom Registry key location passes the connection information to the **BSIDAPI** constructor, copy the custom Registry key instead.

2. Deploy and install an agent key file. See **Agent Key File**.
3. For **BSIDAPI.dll** to run correctly, ensure that the following components are installed:
   - Microsoft .NET v4.6.1 Framework
   - Microsoft Visual C++ 2008 Redistributable Package
   - CryptoCOM.dll
     Note: Deploy **CryptoCOM.dll** to the **System32** folder.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.

7
Troubleshooting

This section provides troubleshooting strategies and the solutions for common errors.

System.Exception:'Empty Path name is not legal'

Possible causes

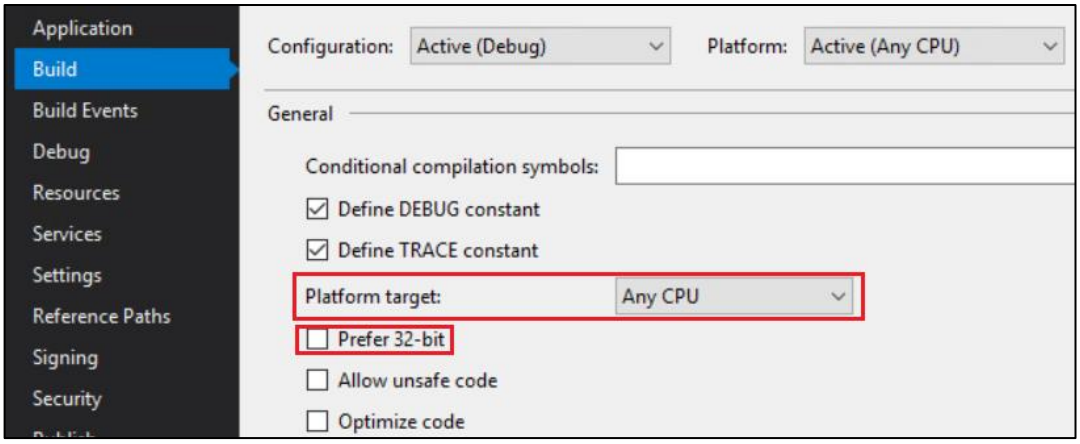
This error message is displayed in Visual Studio when the **Platform target** is set to **Any CPU** while running the console application on a **64-bit** operating system, in which it is utilizing BSIDAPI DLL for the authentication.

Solution

Perform the below steps to resolve the issue:

1. Open the console application in Visual Studio.
2. Right-click the project in Visual Studio and click **Properties**.
3. In the left pane, click **Build**.
4. Under **Build**, from the **Platform target** drop-down, select **Any CPU**.
5. Ensure that **Prefer 32-bit** checkbox is not selected.
6. Save the changes.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.



After making the above configuration change, run the application. Now, Visual Studio will not throw the exception.

Document PN: 007-012398-002, Rev. K, Copyright © Gemalto 2023. All rights reserved.