RSA Authentication Agent 7.1 for Web for IIS 7.0 and 7.5 Developer's Guide

The Security Division of EMC

Contact Information

Go to the RSA corporate web site for regional Customer Support telephone and fax numbers**:** **www.rsa.com**

Trademarks

RSA and the RSA logo are registered trademarks of RSA Security Inc. in the United States and/or other countries. For the most up-to-date listing of RSA trademarks, go to **www.rsa.com/legal/trademarks_list.pdf**. EMC is a registered trademark of EMC Corporation. All other goods and/or services mentioned are trademarks of their respective companies.

License agreement

This software and the associated documentation are proprietary and confidential to RSA, are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright notice below. This software and the documentation, and any copies thereof, may not be provided or otherwise made available to any other person.

No title to or ownership of the software or documentation or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software and the documentation may be subject to civil and/or criminal liability.

This software is subject to change without notice and should not be construed as a commitment by RSA.

Note on encryption technologies

This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when using, importing or exporting this product.

Distribution

Limit distribution of this document to trusted personnel.

RSA notice

The RC5™ Block Encryption Algorithm With Data-Dependent Rotations is protected by U.S. Patent #5,724,428 and #5,835,600.

April 2012

Contents

Preface

About This Guide

This guide describes how to develop custom programs using the application programming interfaces (APIs) of RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5. It is intended for web developers, system engineers, and other trusted personnel. Do not make this guide available to the general user population. Developers must understand the CGI environment as defined for use in the C, Java, Perl, ISAPI, or ASP web application development environments.

RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5 Documentation

For more information about RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5, see the following documentation:

Release Notes. Provides information about what is new and changed in this release, as well as workarounds for known issues. The latest version of the *Release Notes* is available from RSA SecurCare Online at **https://knowledge.rsasecurity.com**.

Installation and Configuration Guide. Describes detailed procedures on how to install and configure the Web Agent.

Developer's Guide. Provides information about developing custom programs using the Web Agent application programming interfaces (APIs).

Integrating RSA Authentication Agent for Web with RSA Authentication Manager Express Guide. Describes detailed procedures on how to install and configure the web agent to work with Authentication Manager Express(AMX).

RSA Authentication Agent Control Panel Help. Describes how to perform test authentications and configure advanced registry settings in the RSA Authentication Agent control panel. To view Help, click the **Help** button in the RSA Authentication Agent control panel.

RSA Web Agent Configuration Help. Describes how to administer the web access authentication properties of the IIS web server. To view Help, click the **Help** link on the RSA SecurID page of any virtual web site in the Internet Information Services (IIS) Manager.

Related Documentation

For more information about products related to RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5, see the following:

RSA Authentication Manager documentation set. The full documentation set for RSA Authentication Manager 6.1.2 is included in the ***InstallPath*\RSA Security\RSA Authentication Manager\doc** directory. The full documentation set for RSA Authentication Manager 7.1 SP4 is included in the ***InstallPath*\doc** directory.

Getting Support and Service

RSA SecurCare Online	**https://knowledge.rsasecurity.com**
Customer Support Information	**www.rsa.com/support**
RSA Secured Partner Solutions Directory	**www.rsasecured.com**

RSA SecurCare Online offers a knowledgebase that contains answers to common questions and solutions to known problems. It also offers information on new releases, important technical news, and software downloads.

The RSA Secured Partner Solutions Directory provides information about third-party hardware and software products that have been certified to work with RSA products. The directory includes Implementation Guides with step-by-step instructions and other information about interoperation of RSA products with these third-party products.

Before You Call Customer Support

Make sure you have direct access to the computer running the Web Agent software.

Please have the following information available when you call:

- ❑ Your RSA Customer/License ID.
- ❑ RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5 software version number. To find this information, click **Start** > **Settings** > **Control Panel**, and double-click **RSA Authentication Agent**. The version number appears in the **Installed RSA Agents** section in the RSA Authentication Agent control panel.
- ❑ The make and model of the machine on which the problem occurs.
- ❑ The name and version of the operating system under which the problem occurs.

1 Using the Web Authentication API

This chapter describes the Web Authentication application programming interface (API). Use this API to add, modify, and delete data within a custom section of the web access authentication browser cookie.

The data is signed by the API as part of the cookie and can be guaranteed against tampering. For privacy, the API also provides a facility to encrypt custom data using the RC5 encryption algorithm.

All API functions described in this document are thread-safe, which means that they can safely be called from multithreaded applications without program failure or data corruption.

Important: United States export regulations impose a limit of six different encrypted custom fields, each of which consists of a tag and its data string. Duplicate tags with the same or different data string do not add to the field count. A maximum of 30 bytes of data can be encrypted in a field. The system returns an error if you exceed these limits.

Sample Source Code

Note: The samples are provided only to illustrate the usage of the cookie APIs.They do not ensure complete protection against security vulnerabilities. It is the developer's responsibility to use appropriate security pratices in the production environment.

The RSA Authentication Agent 7.1 for Web for Internet Information Services 7.0 and 7.5 installer includes sample source code that demonstrates the basic calling sequence and usage in the development environments. You can run the sample code on Windows platforms only.

Note: When running CGI scripts, POST data must be 6036 bytes or less.

Configuring Cookie Expiration Times

If you replace a cookie before a customized web access authentication browser cookie expires, the replacement cookie supersedes the customized cookie. As a result, you lose any third-party data that you are setting using the Web Authentication API.

To prevent the loss of third-party data, use the following guidelines to configure your Web Agent cookie expiration times:

- If the expiration time for idle cookies is greater than the overall cookie expiration time, the idle cookie feature becomes invalid, and the cookie is not replaced.
- If the expiration time for idle cookies is less than three minutes and less than the overall cookie expiration time, the cookie is replaced every 30 seconds.
- If the expiration time for idle cookies is greater than three minutes but less than the overall cookie expiration time, the cookie is replaced every 60 seconds.

For instructions on configuring cookie expiration times, see the Help topics "Configuring Cookie Expiration Times" and "Setting Cookie Expiration Times."

Enable the Web Authentication API

The Web Agent automatically installs **rsacookieapi.dll** in the web server ***%SystemRoot%\system32*** directory. Before you can use **rsacookieapi.dll**, you must enable the use of the Web Authentication API.

Note: You must run **rsacookieapi.dll** on the same server that is running the Web Agent.

To enable the Web Authentication API:

1. Click **Start > Settings > Control Panel**, and then double-click **RSA Web Agent**.
2. In the Connections pane, double-click ***server name*** **> Sites**, and then click the name of the web site whose properties you want to view.
3. Click the **Features View** tab, and in the *web_site* Home pane, double-click **RSA SecurID**.
4. Under **Other Settings**, clear **Disable Cookie API processing**.
5. In the Actions pane, click **Apply**.
6. Restart the Internet Information Services (IIS) web server.

2 API Functions for the C/C++ Environment

This chapter describes the functions you can use in a C development environment.

To use the API functions on Windows, you have to use the **samples/web/include/rsacookieapi.h** file, and link your CGI C executable with the **samples/web/lib/rsacookieapi.lib** file. These files are included with the Web Agent installer.

RSAGetLastError

Description

```
unsigned int RSACOOKIEAPI_API RSAGetLastError(void);
```

The RSAGetLastError function returns the last error code value.

Architecture

This function returns the last error code value. The Output and Post Conditions section of each reference page describes the error codes.

Input Arguments

None.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use this function immediately when a function returns to check for error conditions. Subsequent functions overwrite older error codes.

RSAGetShellField

Description

```
LPCSTR RSACOOKIEAPI_API RSAGetShellField(
    LPCSTR szInstance, LPCSTR Cookie,
    LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent);
```

The RSAGetShellField function retrieves the Default Shell Field value stored in the web access authentication cookie. The value in the cookie is the same as the Default Shell Field value stored in the RSA Authentication Manager database for the user.

Architecture

This function returns the Default Shell Field value as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSAGetShellField function returns a NULL pointer, the caller can use RSAGetLastError to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 11.

Input Arguments

The following table describes the input arguments of the RSAGetShellField function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.

Argument	Description
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.

Output and Post Conditions

This function returns the Default Shell Field as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 11.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSAGetTagField

Description

```
LPCSTR RSACOOKIEAPI_API RSAGetTagField(
    LPCSTR szInstance, LPCSTR Cookie,
    LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent,
    LPCSTR Tag, BOOL Encrypted);
```

The RSAGetTagField function returns a developer-defined field identified by name. The name of the field is given by the Tag argument. The field is assumed to have been stored in the web access authentication cookie by a previous call to the RSASetTagField function.

Architecture

This function returns the field as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSAGetTagField function returns a NULL pointer, the caller can use RSAGetLastError to retrieve one of the defined error codes. For more information, see the "Output and Post Conditions" on page 13.

Input Arguments

The following table describes the input arguments of the RSAGetTagField function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.

Argument	Description
Tag	Name of the field to retrieve.
Encrypted	A Boolean flag indicating that the content of the field was encrypted when it was created by the call to the RSASetTagField function. **Note:** This flag is included for backward compatibility with the Web Authentication API used in RSA ACE/Agent 4.4 for Windows NT, which did not include the embedded Encrypted flag. The RSAGetTagField function ignores this flag.

Output and Post Conditions

This function returns the field as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 14.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_ NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_ CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
101	RSACOOKIE_ERROR_ VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
102	RSACOOKIE_ERROR_ DATA_TAG_NOT_FOUND	The cookie is valid, but the required tag is not present.
103	RSACOOKIE_ERROR_ NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_ INVALID_ARGUMENT	One of the input arguments is invalid.
105	RSACOOKIE_ERROR_ CIPHERSUITE_ERROR	The decryption routine failed to decrypt the tag in the cookie.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSAGetUserName

Description

```
LPCSTR RSACOOKIEAPI_API RSAGetUserName(
    LPCSTR szInstance, LPCSTR Cookie,
    LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent);
```

The RSAGetUserName function returns the user name stored in the web access authentication cookie.

Architecture

This function returns the user name as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSAGetUserName function returns a NULL pointer, the caller can use the RSAGetLastError function to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 15.

Input Arguments

The following table describes the input arguments of the RSAGetUserName function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.

Argument	Description
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.

Output and Post Conditions

This function returns the user name as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 15.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the error value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSASetTagField

Description

```
LPCSTR RSACOOKIEAPI_API RSASetTagField(
    LPCSTR szInstance, LPCSTR Cookie,
    LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent,
    LPCSTR Tag, LPCSTR Data, BOOL Encrypted);
```

The RSASetTagField function stores the NULL-terminated string passed as the Data argument in the web access authentication cookie. If the tag identified by the Tag argument already exists, it is replaced.

Note: If more than one field is to be set, this function can accept the result of a previous call in the Cookie argument.

Architecture

This function returns a new cookie string suitable for an HTTP Set-Cookie: header as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

Note: United States export regulations impose a limit of six different encrypted custom fields (a field consists of a tag and its data string). Duplicate tags with the same or different data string do not add to the field count. A maximum of 30 bytes of data can be encrypted in a field. The system returns an error if you exceed these limits.

If the RSASetTagField function returns a NULL pointer, the caller can use the RSAGetLastError function to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 17.

Input Arguments

The following table describes the input arguments of the RSASetTagField function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.

Argument	Description
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.
Tag	Name of the field to set or replace.
Data	The data to set in the field. If binary data must be stored, use any suitable ASCII encoding function to convert the data to a NULL-terminated ANSI string. If the Encrypted flag is set, the maximum is 30 bytes of data. If the Encrypted flag is not set, the only size limitations are those imposed by the browser and available memory.
Encrypted	A Boolean flag indicating that the content of the field is to be encrypted.

Output and Post Conditions

This function returns the new cookie string as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 18.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

Error Code	Constant	Description
105	RSACOOKIE_ERROR_ CIPHERSUITE_ERROR	The decryption routine failed to decrypt the tag in the cookie.
106	RSACOOKIE_ERROR_ LONGDATALEN_ ENCRYPTION	Attempted to encrypt more than 30 bytes of data in a field.
107	RSACOOKIE_ERROR_ TOOMANY_ENCRYPTED _FIELDS	Attempted to encrypt more than six fields of custom data.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer

Error Handling

To handle errors appropriately, use the value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSADeleteTagField

Description

```
LPCSTR RSACOOKIEAPI_API RSADeleteTagField(
    LPCSTR szInstance, LPCSTR Cookie,
    LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent,
    LPCSTR Tag);
```

The RSADeleteTagField function deletes a developer-defined field identified by name. The name of the field is given by the Tag argument. The field is assumed to have been stored in the web access authentication cookie by a previous call to the RSASetTagField function.

Note: If more than one field is to be deleted, this function can accept the result of a previous call in the Cookie argument.

Architecture

This function returns a new cookie string suitable for an HTTP Set-Cookie: header as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSADeleteTagField function returns a NULL pointer, use the RSAGetLastError function to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 19.

Input Arguments

The following table describes the input arguments of the RSADeleteTagField function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.
Tag	Name of the field to delete.

Output and Post Conditions

This function returns a NULL pointer and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 20.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.

Error Code	Constant	Description
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
102	RSACOOKIE_ERROR_DATA_TAG_NOT_FOUND	The cookie is valid but the required tag is not present.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSAFreeMemory

Description

```
VOID RSACOOKIEAPI_API RSAFreeMemory(LPCSTR Buffer);
```

The RSAFreeMemory function frees memory buffers returned by any of the other C API functions.

Architecture

This function frees the memory returned by the rsacookieapi library. If you use the function with any other type of memory buffer, the program will not run to completion.

Input Arguments

The following table describes the input arguments of the RSAFreeMemory function.

Argument	Description
Buffer	The address of the buffer returned by the previous call.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Output and Post Conditions

The buffer referenced by the Buffer input argument is no longer valid.

RSACookieInitializeCGI

Note: Any CGI programs using the Web Authentication API must include the RSACookieInitializeCGI call in order to initialize the API. The RSACookieInitializeCGI call must be made before any other calls are made to the Web Authentication API and for each request your CGI program handles.

Description

```
LPCSTR RSACOOKIEAPI_API RSACookieInitializeCGI ();
```

The RSACookieInitializeCGI function initializes the Web Authentication API for server-side CGI programs. It must be invoked for each request that the CGI handles. All other Web Authentication API calls will fail if this function has not been called.

Input Arguments

None.

Output and Post Conditions

Return values are as shown in the following table.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid. Make sure you have enabled the Web Authentication API in the RSA SecurID Web Access Authentication properties sheet in the Internet Information Services (IIS) Manager. For more information, see "Enable the Web Authentication API" on page 8.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSACookieInitializeExt

Description

```
LPCSTR RSACOOKIEAPI_API RSACookieInitializeExt (
    CHttpServerContext *pCtxt);
```

The RSACookieInitializeExt function initializes the Web Authentication API for ISAPI extensions. It must be invoked for each request that the extension handles. All other Web Authentication API calls will fail if this function has not been called.

Note: Any ISAPI extensions using the Web Authentication API must include the RSACookieInitializeExt call in order to initialize the API. The RSACookieInitializeExt call must be made before any other calls are made to the Web Authentication API and for each request that your CGI program or ISAPI extension handles.

Input Arguments

The following table describes the input arguments of the RSACookieInitializeExt function.

Argument	Description
pCtxt	A pointer to the ChttpServerContext, which you can retrieve from the EXTENSION_CONTROL_BLOCK structure.

Output and Post Conditions

Return values are as shown in the following table.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid. Make sure you have enabled the Web Authentication API in the RSA SecurID Web Access Authentication properties sheet in the IIS Manager. For more information, see "Enable the Web Authentication API" on page 8.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSACookieInitializeExtCtx

Description

```
LPCSTR RSACOOKIEAPI_API RSACookieInitializeExtCtx (
IHttpContext *pCtxt);
```

The RSACookieInitializeExtCtx function initializes the Web Authentication API forHttpModule applications. It must be invoked for each request that the Module handles. All other Web Authentication API calls will fail if this function has not been called.

Note: Any HttpModule using the Web Authentication API must include the RSACookieInitializeExtCtx call in order to initialize the API. The RSACookieInitializeExtCtx call must be made before any other calls are made to the Web Authentication API and for each request that your HttpModule handles.

Input Arguments

The following table describes the input arguments of the RSACookieInitializeExt function.

Argument	Description
pCtxt	A pointer to the IhttpContext, which you can receive from CHttpModule handler functions like OnBeginRequest(), OnAuthenticateRequest() etc.

Output and Post Conditions

Return values are as shown in the following table.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_ NO_ERROR	The operation was successful.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid. Make sure you have enabled the Web Authentication API in the RSA SecurID Web Access Authentication properties sheet in the IIS Manager. For more information, see "Enable the Web Authentication API" on page 8.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSACookieInitializeExtNoMFC

Description

```
LPCSTR RSACOOKIEAPI_API RSACookieInitializeExtNoMFC (
EXTENSION_CONTROL_BLOCK* pCtxt);
```

The RSACookieInitializeExtNoMFC function initializes the Web Authentication API for non-MFC based functions. It must be invoked for each request that the extension handles. All other Web Authentication API calls fail if this function has not been called.

Note: Any non-MFC ISAPI extensions using the Web Authentication API must include the RSACookieInitializeExtNoMFC call in order to initialize the API. The RSACookieInitializeExtNoMFC call must be made before any other calls are made to the Web Authentication API and for each request that your CGI program or ISAPI extension handles.

Input Arguments

The following table describes the input arguments of the RSACookieInitializeExtNoMFC function.

Argument	Description
pCtxt	A pointer to the EXTENSION_CONTROL_BLOCK structure.

Output and Post Conditions

Return values are as shown in the following table.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid. Make sure you have enabled the Web Authentication API in the RSA SecurID Web Access Authentication properties sheet in the IIS Manager. For more information, see "Enable the Web Authentication API" on page 8.

Error Handling

To handle errors appropriately, developers must use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSACookieInitializeFilt

Description

```
LPCSTR RSACOOKIEAPI_API RSACookieInitializeFilt (
```

```
HTTP_FILTER_CONTEXT* pCtxt);
```

The RSACookieInitializeFilt initializes the Web Authentication API for ISAPI filters. It must be invoked for each request that the extension handles. All other Web Authentication API calls fail if this function has not been called.

Note: Any ISAPI filters using the Web Authentication API must include the RSACookieInitializeFilt call in order to initialize the API. The RSACookieInitializeFilt call must be made before any other calls are made to the Web Authentication API and for each request that your CGI program or ISAPI extension handles.

Input Arguments

The following table describes the input arguments of the RSACookieInitializeFilt function.

Argument	Description
pCtxt	A pointer to the HTTP_FILTER_CONTEXT structure.

Output and Post Conditions

Return values are as shown in the following table.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid. Make sure you have enabled the Web Authentication API in the RSA SecurID Web Access Authentication properties sheet in the IIS Manager. For more information, see "Enable the Web Authentication API" on page 8.

Error Handling

To handle errors appropriately, developers must use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSAGetCSRFToken

Description

```
LPCSTR RSACOOKIEAPI_API RSAGetCSRFToken (
LPCSTR szInstance, LPCSTR Cookie,
LPCSTR User, LPCSTR BrowserIP, LPCSTR Agent);
```

The RSAGetCSRFToken function returns the RSA token stored in the web access authentication cookie. This token is required to protect against cross-site request forgery attacks during logoff. For more information, see the Help topic "Using the RSA Token for Cross-Site Request Forgery Protection."

Architecture

This function returns the RSA token as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSAGetCSRFToken function returns a NULL pointer, the caller can use the RSAGetLastError function to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 29.

Input Arguments

The following table describes the input arguments of the RSAGetCSRFToken function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.
Cookie	Value of the CGI variable **HTTP_COOKIE** unmodified.
User	Value of the CGI variable **REMOTE_USER** unmodified.
BrowserIP	Value of the CGI variable **REMOTE_ADDR** unmodified.

Argument	Description
Agent	Value of the CGI variable **HTTP_USER_AGENT** unmodified.

Output and Post Conditions

This function returns the RSA token as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this method to construct the logoff URL, refer to the sample code included with the Web Agent installer. Here is an example logoff URL with RSArand:

```
CSRFToken = RSAGetCSRFToken(ID,Cookie,User,BrowserIP,Agent);
printf("<a href=\"%s?logoff?RSArand=%s\">Logoff</a>",
pWebIDURL, CSRFToken);
```

A hidden field called RSArand is added to the logoff submission URL. The Web Agent checks the value of RSArand to ensure that the request actually comes from the user who authenticated with the Web Agent, and that the logoff request is not the result of a cross-site request forgery attack.

Error Handling

To handle errors appropriately, use the error value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

RSAGetWebIDURL

Description

```
LPCSTR RSACOOKIEAPI_API RSAGetWebIDURL(LPCSTR szInstance);
```

The RSAGetWebIDURL function returns the RSA Web Agent filter URL for the currently accessed RSA protected resource. This URL is required for constructing the correct logoff URL architecture

Architecture

This function returns the RSA Web Agent filter URL for the currently accessed RSA protected resource as a NULL-terminated string. To contain the string, the function allocates a buffer that must be freed by the caller when the buffer is no longer useful. To free the buffer, your code must pass the buffer to the RSAFreeMemory function.

If the RSAGetWebIDURL function returns a NULL pointer, the caller can use the RSAGetLastError function to retrieve one of the defined error codes. For more information, see "Output and Post Conditions" on page 30.

Input Arguments

The following table describes the input arguments of the RSAGetWebIDURL function.

Argument	Description
szInstance	Value of the CGI variable **INSTANCE_ID** converted to integer.

Output and Post Conditions

This function returns the RSA Web Agent filter URL as a NULL-terminated string and sets the last error code to one of the values in the following table. Use RSAGetLastError to return the value. For more information, see "Error Handling" on page 31.

Error Code	Constant	Description
0	RSACOOKIE_ERROR_NO_ERROR	The operation was successful.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the web server process to retrieve the necessary information.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

For information on field and parameter settings that result in particular error codes, see Chapter 5, "Troubleshooting C/C++ and Perl Programs."

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the error value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must call the RSAGetLastError function and take appropriate action.

3 API Functions for the COM/ASP Environment

This chapter describes the functions of the Web Authentication API that are suitable for use in a COM/ASP development environment.

Note: To use any of the functions listed in this chapter, you must create an instance of the Rsacookieapi.RSACookie COM object. This object can be instantiated only in an ASP web server page. To do this in Visual Basic script, you would use a statement of the form:

```
RSACookieAPI = Server.CreateObject("Rsacookieapi.RSACookie")
```

RSAGetUserName

Description

```
BSTR RSAGetUserName();
```

The RSAGetUserName function retrieves the user name stored in the web access authentication cookie.

Architecture

This function returns the user name as a string. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

None.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the following server contexts is invalid: • INSTANCE_ID • REMOTE_USER • REMOTE_ADDR • HTTP_USER_AGENT

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSAGetShellField

Description

```
BSTR RSAGetShellField();
```

The RSAGetShellField function retrieves the Default Shell Field value stored in the web access authentication cookie. The value in this field is the same as the Default Shell Field value stored in the RSA Authentication Manager database for the user.

Architecture

This function returns the Default Shell field as a string. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

None.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the following server contexts is invalid: • INSTANCE_ID • REMOTE_USER • REMOTE_ADDR • HTTP_USER_AGENT

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSAGetTagField

Description

```
BSTR RSAGetTagField(BSTR Tag, BOOL Encrypt);
```

The RSAGetTagField function retrieves a developer-defined field identified by name. The Tag argument provides the names of the field. The field is assumed to have been stored in the web access authentication cookie by a previous call to the RSASetTagField function.

Architecture

This function returns the field data as a string. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

The following table describes the input arguments of the RSAGetTagField function.

Argument	Description
Tag	Name of the field to retrieve.
Encrypted	A Boolean flag indicating that the contents of the field were encrypted when the field was created by the call to RSASetTagField.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_ CANNOT_ACCESS_ SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_ VALID_COOKIE_NOT_ FOUND	The Cookie argument does not contain a valid RSA cookie.
102	RSACOOKIE_ERROR_ DATA_TAG_NOTFOUND	The cookie is valid, but the required tag is not present.

Error Code	Constant	Description
103	RSACOOKIE_ERROR_ NOT_ENOUGH_ MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_ INVALID_ ARGUMENT	One of the input arguments is invalid.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code provided with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSASetTagField

Description

```
BSTR RSASetTagField(BSTR Tag, BSTR Data, BOOL Encrypt);
```

The RSASetTagField function stores the NULL-terminated string passed as the Data argument in the web access authentication cookie. If the name identified by the Tag argument already exists, it will be replaced. Use the request object to set the value of the cookie directly before attempting a second call to the COM API.

Architecture

This function returns a new cookie string suitable for an HTTP Set-Cookie: header as a NULL-terminated string. The function sets the cookie header in the response object upon a successful call. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Note: Export regulations impose a limit of six different encrypted custom fields (a field consists of a tag and its data string). Duplicate tags with the same or different data string do not add to the count of fields. A maximum of 30 bytes of data can be encrypted in a field. The system returns an error if you exceed these limits.

Input Arguments

The following table describes the input arguments of the RSASetTagField function.

Argument	Description
Tag	Name of the field to set or replace.
Data	The data to set in the field. If you want to store binary data, use any suitable ASCII encoding function to convert the data to a NULL-terminated ANSI string.
Encrypted	A Boolean flag indicating that the contents of the field are to be encrypted. You must specify the same flag when retrieving the data by the call to RSAGetTagField.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.
106	RSACOOKIE_ERROR_LONGDATALEN_ENCRYPTION	Attempted to encrypt more than 30 bytes of data in a field.
107	RSACOOKIE_ERROR_TOOMANY_ENCRYPTED_FIELDS	Attempted to encrypt more than six fields of custom data.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code provided with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSADeleteTagField

Description

```
BSTR RSADeleteTagField(BSTR Tag);
```

The RSADeleteTagField function deletes a developer-defined field identified by name. The name of the field is given by the Tag argument. The field is assumed to have been stored in the web access authentication cookie by a previous call to the RSASetTagField function. Use the request object to set the value of the cookie directly before attempting a second call to the COM API.

Architecture

This function returns a new cookie string suitable for an HTTP Set-Cookie: header as a NULL-terminated string. The function sets the cookie header in the response object upon a successful call. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

The following table describes the input arguments of the RSADeleteTagField function.

Argument	Description
Tag	The name of the field to delete.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
102	RSACOOKIE_ERROR_DATA_TAG_NOT_FOUND	The cookie is valid, but the required tag is not present.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the input arguments is invalid.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSAGetCSRFToken

Description

```
BSTR RSAGetCSRFToken();
```

The RSAGetCSRFToken function returns the RSA token stored in the web access authentication cookie. This token is required to protect against cross-site request forgery attacks during logoff. For more information, see the Help topic "Using the RSA Token for Cross-Site Request Forgery Protection."

Architecture

This function returns the RSA Token as a string. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

None.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
101	RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	The Cookie argument does not contain a valid RSA cookie.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.
104	RSACOOKIE_ERROR_INVALID_ARGUMENT	One of the following server contexts is invalid: • INSTANCE_ID • REMOTE_USER • REMOTE_ADDR • HTTP_USER_AGENT

Calling or Command Sequence

For examples of how to use this method to construct the logoff URL, refer to the sample code included with the Web Agent installer. Here is an example logoff URL with RSArand:

```
<a href =<%=RSACookieAPI.RSAGetWebIDURL()%> ?logoff?
RSArand=<%=RSACookieAPI.RSAGetCSRFToken()%>>Logoff</a>
```

A hidden field called RSArand is added to the logoff submission URL. The Web Agent checks the value of RSArand to ensure that the request actually comes from the user who authenticated with the Web Agent, and that the logoff request is not a result of a cross-site request forgery attack.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

RSAGetWebIDURL

Description

```
BSTR RSAGetWebIDURL();
```

The RSAGetWebIDURL function returns the RSA Web Agent filter URL for the currently accessed RSA protected resource. This URL is required for constructing the correct logoff URL architecture

Architecture

This function returns the RSA Web Agent filter URL as a string. If the value returned is an empty string, the caller can use the error property of the RSACookie object to retrieve one of the defined error codes.

Input Arguments

None.

Output and Post Conditions

If the value that is returned by this function is an empty string, the error property will contain one of the following values.

Error Code	Constant	Description
100	RSACOOKIE_ERROR_CANNOT_ACCESS_SETTINGS	The API library cannot communicate with the IIS process to retrieve the necessary information.
103	RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	There is not enough memory to perform the requested operation.

Calling or Command Sequence

For examples of how to use this function, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, developers should use the value returned by this function at a decision point in their code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

4 API Application for the Perl Script Environment

This chapter describes the application in the Web Authentication API that is suitable for use in a Perl script development environment.

The Perl API consists of a small application named **rsacookie.exe**. Copy the **rsacookie.exe** file into the cgi-bin directory of IIS (wwwroot\cgi.bin). You can call the applet only in the context of a Perl script running as a CGI application on the web server. Attempting to call this applet directly results in an error code.

Description of the API Application

```
rsacookie -g -shell
          get contents of shell field
rsacookie -g -tag tag_name
          get data field contents from tag named tag_name
rsacookie -g -e -tag tag_name
          get encrypted data field from tag tag_name
rsacookie -g -user
          get contents of user field
rsacookie -g -CSRFToken
          get the RSA token for the current session. This
token is required to protect against cross-site request
forgery attacks during logoff. For more information refer
the Help topic "Using the RSA Token for Cross-Site Request
Forgery Protection."
rsacookie -g -webIdURL
          get the RSA Web Agent filter URL
rsacookie -s -tag tag_name -data ASCII_data
          set data field contents into tag named tag_name
rsacookie -s -e -tag tag_name -data ASCII_data
          set encrypted data field into tag named tag_name
rsacookie -d -tag tag_name
          delete tag named tag_name
```

Architecture

This application returns the requested data as standard output. If there are any errors, the output is emitted as stderr and the error code is set to indicate the type of failure. To set or delete multiple tags, use setenv to set the HTTP_COOKIE variable to the return value of **rsacookie.exe** for each call.

Note: United States export regulations limit encryption of custom data to six different fields (a field consists of a tag and its data string). Duplicate tags with the same or different data string do not add to the field count. A maximum of 30 bytes of data can be encrypted in a field. The system returns an error if you exceed these limits.

Input Arguments

The data supplied as input for the -s switch must follow the -data switch and must be a contiguous ASCII string. To include spaces and non-alphanumeric characters in the string, enclose the string in double quotation marks (“ ”). To include a quotation mark in the string, you must place a backslash directly before the quotation mark (\”).

Output and Post-Conditions

For information on field and parameter settings that result in particular error codes, see Chapter 5, “Troubleshooting C/C++ and Perl Programs.” The following table lists the error codes and their description.

Error Code	Description
0	Success. The output is the requested data.
1	The supplied arguments are invalid or incorrect.
2	The required CGI variables are missing.
3	The cookie operation failed. The first line of output in stderr contains the detailed error.
4	Attempted to encrypt more than six custom data fields.
5	Attempted to encrypt more than 30 bytes of data in a custom data field.
6	The encryption routine failed to encrypt the tag in the cookie.

Calling or Command Sequence

For examples of how to use this API application, refer to the sample code included with the Web Agent installer.

Error Handling

To handle errors appropriately, use the value returned by this function at a decision point in your code. A successful return allows processing to continue. To handle a failure, your code must examine the value of the error property and take appropriate action.

5 Troubleshooting C/C++ and Perl Programs

This chapter provides information on the error codes returned by Web Authentication API calls, depending on field and parameter settings.

Getting Third-Party Tag Data from the Cookie

You can use the following C/C++ API function to get third-party data from the cookie:

```
RSAGetTagField(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent,
   const char* Tag,
   int Encrypted)
```

You can use the Perl API application with the following argument to get third-party data from the cookie:

```
rsacookie -g -tag tag_name
```

The following table shows the error codes returned in C and Perl when you get third-party tag data from the cookie.

Action	Result (C)	Result (Perl)
Field Tag set to a nonexistent flag.	Error Code 102: RSACOOKIE_ERROR_DATA_TAG_NOT_FOUND	Error Code 3
Field Tag set to an empty string.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 3
Field Tag parameter set to the Tag of an unencrypted field. Encrypted parameter set to TRUE.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0

Action	Result (C)	Result (Perl)
Field Tag parameter set to the Tag of an encrypted field. Encrypted parameter set to FALSE.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Field Tag set to a NULL pointer.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 1

Setting Third-Party Tag Data in the Cookie

You can use the following C/C++ API function to set third-party data in the cookie:

```
RSASetTagField(const char* szInstance,
      const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent,
   const char* Tag,
   const char* Data,
      int Encrypted)
```

You can use the Perl API function with the following argument to set third-party data in the cookie:

```
rsacookie -s -tag tag_name -data ASCII_data
```

The following table shows the error codes returned in C and Perl when you set third-party tag data in the cookie.

Action	Result (C)	Result (Perl)
Field Tag set to a very long string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Field Tag set to an empty string.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 3

Action	Result (C)	Result (Perl)
Field Data set to a very long string.	If process does not run out of memory, Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
	If process runs out of memory, Error Code 103: RSACOOKIE_ERROR_NOT_ENOUGH_MEMORY	Error Code 3
31-byte (or longer) encrypted Data field added. (Maximum length is 30 bytes.)	Error Code 106: RSACOOKIE_ERROR_LONGDATALEN_ENCRYPTION	Error Code 5
Seven or more encrypted Data fields added. (Maximum is six encrypted fields.)	Error Code 107: RSACOOKIE_ERROR_TOOMANY_ENCRYPTED_FIELDS	Error Code 4
Field Data set to an empty string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Field Data set to a NULL pointer.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 1

Parameter Settings

You can use the following C/C++ API functions to set various parameters:

```
RSAGetShellField(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent)

RSAGetTagField(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent,
   const char* Tag,
   int Encrypted)
```

```
RSAGetUserName(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent)

RSASetTagField(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent,
   const char* Tag,
   const char* Data,
   int Encrypted)

RSADeleteTagField(const char* szInstance,
   const char* Cookie,
   const char* User,
   const char* BrowserIP,
   const char* Agent,
   const char* Tag)
```

You can use the Perl API application with the following arguments to set various parameters:

```
rsacookie -g –shell
rsacookie -g -tag tag_name
rsacookie -g –user
rsacookie -s -tag tag_name -data ASCII_data
rsacookie -d -tag tag_name
```

Agent Parameter Settings

The following table shows the error codes returned in C and Perl when you set the Agent parameter to different values.

Action	Result (C)	Result (Perl)
Agent parameter set to a value other than the value of the *HTTP_USER_AGENT* environmental variable.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Agent parameter set to a very long string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Agent parameter set to an empty string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
Agent parameter set to a NULL pointer.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 1

BrowserIP Parameter Settings

The following table shows the error codes returned in C and Perl when you set the BrowserIP parameter to different values.

Action	Result (C)	Result (Perl)
BrowserIP parameter set to a value other than the value of the *REMOTE_ADDR* environmental variable.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
BrowserIP parameter set to a very long string.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
BrowserIP parameter set to an empty string.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
BrowserIP parameter set to a NULL pointer.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 1

Cookie Parameter Settings

The following table shows the error codes returned in C and Perl when you set the Cookie parameter to different values.

Action	Result (C)	Result (Perl)
Cookie parameter set to a value other than the value of the *HTTP_COOKIE* environmental variable.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
Cookie parameter set to a very long string.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
Cookie parameter set to an empty string.	Error Code 101: RSACOOKIE_ERROR_VALID_COOKIE_NOT_FOUND	Error Code 3
Cookie parameter set to a NULL pointer.	Error Code 104: RSACOOKIE_ERROR_INVALID_ARGUMENT	Error Code 1

szInstance Parameter Settings

The following table shows the error codes returned in C and Perl when you set the szInstance parameter to different values.

Action	Result (C)	Result (Perl)
szInstance parameter set to a value other than the value of the *SERVER_NAME* environmental variable.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
szInstance parameter set to a very long string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
szInstance parameter set to an empty string.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0
szInstance parameter set to a NULL pointer.	Error Code 0: RSACOOKIE_ERROR_NO_ERROR	Error Code 0