IBM Security Secret Server
Version 10.5

# *Web Service API Guide*

# Table of Contents

*Last updated: September 20, 2018*

# Overview

Secret Server provides web services to enable 3rd party developers to interact with Secret Server in a developer-friendly way while maintaining security.

The web services use a standard messaging format called Simple Object Access Protocol (SOAP for short). SOAP is an XML-based message exchange protocol to exchange information over a standard HTTP connection.

Secret Server is built using version 1.2 of the SOAP protocol.

For technical details of the SOAP protocol, please refer to the W3C here:

http://www.w3.org/TR/2007/REC-soap12-part0-20070427/

# Accessing Web Services

Secret Server's web services can be accessed with the following URL:

http://yoursecretserverinstallation/webservices/sswebservice.asmx

For getting the WSDL definition, append **?wsdl** to the end of the URL:

http://yoursecretserverinstallation/webservices/sswebservice.asmx?wsdl

# Concepts

## Token

Secret Server requires the use of a token for most of the web service methods. A token will be returned after authentication has been successful using the one of the methods below:

- Authenticate
- AuthenticateRadius

The token is then passed to other web service methods that require it. The token contains information from when it was initially acquired, such as the date it was created for expiration purposes. Secret Server verifies that the token is still valid on the web service methods that use it.

The token should be persisted to storage if authentication sessions go beyond the lifetime of your application. When the application starts up, use GetTokenIsValid to determine if the token is still valid.

If the token is no longer valid, you must re-authenticate to acquire a new token.

### Return Types

Secret Server web service methods that can fail will return a web service type that includes an Errors property, which is a collection of strings. If there are one or more strings in the Errors collection, the service method is assumed to have failed. The Errors collection contains details of the nature of the failure.

For service methods that return a type without an Errors property, the service method should succeed in all cases unless otherwise noted.

### Windows Authentication

Secret Server supports using Windows Authentication with Active Directory accounts. A different web service is available for Windows Authentication. It is available at:

https://yoursecretserverinstallation/winauthwebservices/sswinauthwebservice.asmx

This web service functions identically as the standard web service, however the web methods do not accept a token. Instead, the Windows Identity that the web service is running as is used to obtain the current account.

## Common Errors

Web service methods that require a token may return any of the following errors in addition to web service specific errors:

**"Bad token."**

The token is no longer valid.

**“IP Address is not allowed.”**

The IP address from which the authentication is being performed is not allowed for the user. See the Secret Server User Guide for more information about IP address restrictions.

**"Your licensing is not up to date. Please check your licensing for enabled users."**

Secret Server has more active users than it is licenses for. You must update your licensing or disable users.

**"Secret Server instance you are connected to is running in Limited Mode."**

Secret Server’s licenses have not been activated, and are past the grace period. You must activate your licenses.

**"Please check your email for a confirmation message to validate your email address. You will need to validate your email address before adding passwords to this application. Your email will only be used for resetting a forgotten password."**

The email address for the account, which the token was created from, has not had its email address verified yet.

# Sample Code

Sample code is available at the Thycotic GitHub page.

GitHub page: https://github.com/thycotic/SecretServerWebServices

Zip: https://github.com/thycotic/SecretServerWebServices/zipball/master

# Web Service Methods

## AddDependency

Adds a Dependency onto a Secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**dependency**

Type: Dependency
Required: yes
The dependency to add. See the type for more information.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user associated with the token does not have view permission on the Secret to add dependencies.

**“Access Denied - User does not own Secret”**

The user associated with the token does not have owner permission on the Secret to add dependencies.

**“Secret Dependency X - Y already exists on Secret (Z)”**

The dependency already exists on the Secret.

**“Unable to add Dependency (X). Type 'Y' does not exist”**

The dependency type ID that was specified is not a valid dependency type.

**"PowershellScriptId is invalid"**

The PowerShell script ID is not a valid PowerShell script ID if the Dependency type is a PowerShell script.

**"Access Denied to PrivilegedAccountSecretId"**

The user associated with the token does not have view permission on the Secret that is being used as a privileged account for the dependency.

## AddGroupToActiveDirectorySynchronization

Adds an Active Directory group that users can by synced from.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**GroupName**

Type: string
Required: yes
The name of the Active Directory Group to add.

**DomainId**

Type: int
Required: yes
The Id of the Active Directory domain in Secret Server that the group gets added to.

**DomainName**

Type: string
Required: no
The name of the Active Directory domain in Secret Server

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permissions to manage Active Directory Sync.

**"GroupName is a required field"**

An Active Directory group name is required.

**“DomainName is required or a DomainId greater than 0”**

The domain name or domain Id of the Active Directory instance in Secret Server is required

## AddNewSecret

This web method is used to create a new Secret. When created, a “WEBSERVICECREATE” audit is made with the Secret. If the user that the token was created with does not have permission to create a Secret due to Role permissions, a SecurityException is returned with an HTTP Error Code of 500 instead of a normal error being returned.

### Return Type

Secret

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secret**

Type: Secret
Required: yes
The Secret object that will be created in Secret Server.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

This web method may throw an exception, resulting in the server returning an HTTP 500 error code if the user does not have permission to create a Secret due to Role permissions.

**“Secret name exceeds max length ({0} characters)”**

The Secret name specified exceeds the maximum length.

{0} – The number of characters that the Secret name cannot exceed.
As of Secret Server 4.0, the name limit was 1991 characters.

**“Secret name cannot be empty.”**

The supplied Secret name is zero characters in length. White space is ignored when checking if the Secret name is empty.

**“Secret item value exceeds max length ({0} characters).”**

A Secret item supplied in the secretItemValues exceeds the maximum length.

{0} – The number of characters that the Secret item cannot exceed.
As of Secret Server 7.8.000036, the item length limit is 10,000 characters. Prior to that version, the limit is 1991 characters.

**"The folder does not exist or user does not have access."**

The folderId supplied does not exist.

**"User does not have permission to edit the folder."**

The user associated with the current token does not have Edit or Owner permission on the folder specified.

**"Provided Secret type is invalid."**

The supplied secretTypeId is not a valid Secret template Id.

**"Secret template is out of date."**

The number of Secret template fields specified in the secretFieldIds parameter does not match the number of fields on the template, or the secretFieldsIds were not passed in the order that they are defined on the Secret template.

**"Password does not meet the minimum length requirement of {0}."**

A Secret item supplied in the secretItemValues parameter that is a password does not meet minimum length requirements as defined on the Secret template.

{0} – The minimum length as defined by the Secret template.

**"Password does not meet the complexity requirements."**

A Secret item supplied in the secretItemValues parameter that is a password does not meet complexity requirements as defined on the Secret template.

**"You have gone over the limit of 100 Secrets created in a single day for free users. You may wait until tomorrow or purchase a license."**

For Secret Server Online users only. This error occurs for free accounts that attempt to create more than 100 Secrets in a 24-hour period.

## AddScript

This web method is used to create a new PowerShell, SSH, or SQL Script.

### Return Type

AddScriptResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**ScriptId**

Type: integer
Required: yes
The Id of the new script that will be created. It must be set to 0.

**ScriptName**

Type: string
Required: yes
The name of the Script.

**ScriptDescription**

Type: string
Required: no
A Description of the Script being created.

**Active**

Type: boolean
Required: yes
Sets whether or not the Script is Active

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

This web method may throw an exception, resulting in the server returning an HTTP 500 error code if the user does not have permission to create a Secret due to Role permissions.

**"Only a new script without a ScriptId may be added."**

The Script Id must be set to 0.

**"Access Denied."**

The user account does not have permissions to Create scripts.

## AddSecret

This web method is used to create a new Secret. When created, a "WEBSERVICECREATE" audit is made with the Secret. If the user that the token was created with does not have permission to create a Secret due to Role permissions, a SecurityException is returned with an HTTP Error Code of 500 instead of a normal error being returned.

### Return Type

AddSecretResult

### Parameters

**token**

Type: string

Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretTypeId**

Type: integer
Required: yes
The Id of the Secret template that the new Secret will be based off of . A Secret template Id can be found by enumerating available templates with the SearchSecretsByFieldValue web method.

**secretName**

Type: string
Required: yes
The name of the Secret. This must not exceed 1991 characters in length.

**secretFieldIds**

Type: integer array
SOAP Type: arrayOfInteger
Required: yes
An array of field Ids from the Secret template. All fields must be specified, and the order should correspond to the secretItemValues parameter.

**secretItemValues**

Type: string array
SOAP Type: arrayOfString
Required: yes
An array of values for the Secret fields. The order must correspond to the order that the secretFieldIds parameter was supplied.

**folderId**

Type: integer
Required: yes
The Id of the folder the Secret should belong to. Specify negative one (-1) if the Secret should not belong to any folder.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

This web method may throw an exception, resulting in the server returning an HTTP 500 error code if the user does not have permission to create a Secret due to Role permissions.

**"Secret name exceeds max length ({0} characters)."**

The Secret name specified exceeds the maximum length.

{0} – The number of characters that the Secret name cannot exceed.
As of Secret Server 4.0, the name limit was 1991 characters.

**“Secret name cannot be empty.”**

The supplied Secret name is zero characters in length. White space is ignored when checking if the Secret name is empty.

**“Secret item value exceeds max length ({0} characters).”**

A Secret item supplied in the secretItemValues exceeds the maximum length.

{0} – The number of characters that the Secret item cannot exceed.
As of Secret Server 7.8.000036, the item length limit is 10,000 characters. Prior to that version, the limit is 1991 characters.

**“The folder does not exist or user does not have access.”**

The folderId supplied does not exist.

**“User does not have permission to edit the folder.”**

The user associated with the current token does not have Edit or Owner permission on the folder specified.

**“Provided Secret type is invalid.”**

The supplied secretTypeId is not a valid Secret template Id.

**“Secret template is out of date.”**

The number of Secret template fields specified in the secretFieldIds parameter does not match the number of fields on the template, or the secretFieldsIds were not passed in the order that they are defined on the Secret template.

**“Password does not meet the minimum length requirement of {0}.”**

A Secret item supplied in the secretItemValues parameter that is a password does not meet minimum length requirements as defined on the Secret template.

{0} – The minimum length as defined by the Secret template.

**“Password does not meet the complexity requirements.”**

A Secret item supplied in the secretItemValues parameter that is a password does not meet complexity requirements as defined on the Secret template.

**“You have gone over the limit of 100 Secrets created in a single day for free users. You may wait until tomorrow or purchase a license.”**

For Secret Server Online users only. This error occurs for free accounts that attempt to create more then 100 Secrets in a 24-hour period.

## AddSecretCustomAudit

This web service adds a custom audit record to an existing Secret. The Action will be CUSTOMAUDIT.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: Yes
The Id of the Secret that the audit will be added to.

**notes**

Type: string
Required: no
The additional notes that will be added the new Secret audit record.

**ipAddress**

Type: string
Required: no
The IP Address that will be associated with the audit record.

**referenceId**

Type: int
Required: no
If using a ticketing system in Secret Server, this number will be the referenced ticket number for the audit record.

**userId**

Type: int
Required: Yes
The ID for the user that the audit will be generated for.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"User does not have permission to add custom audits to a Secret."**

The user associated with the token does not have permission to add custom audits.

**"Access Denied."**

The user associated with the token does not have Owner permission of the Secret.

## AddSecretPolicy

Creates a new Secret Policy

### Return Type

AddSecretPolicyResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**SecretPolicyId**

Type: int
Required: yes
The Id of the Secret Policy that will be created. Enter 0 for the SecretPolicyId

**SecretPolicyName**

Type: yes
Required: yes
The name of the Secret Policy.

**SecretPolicyDescription**

Type: string
Required: no
A description of the Secret Policy

**Active**

Type: boolean
Required: yes
Specifies whether the Secret Policy is Active or Inactive.

Zero or more repetitions may be made of the items below.

**SecretPolicyItemMapId**

Type: int
Required: yes
Ignored on Create. Enter 0 for the SecretPolicyItemMapId

**SecretPolicyItemId**

Type: int
Required: yes
An array of field Ids from the Secret Policy Items. All fields must be specified, and the order should correspond to the secretpolicyitems parameter.

**PolicyApplyCode**

Type: string

Required: yes
Sets the policy item to Not Set, Default, or Enforced. Not Set = 0, Default = 1, Enforced = 2

**EnabledValue**

Type: boolean
Required: yes
Sets whether the policy item is enabled or disabled

**IntegerValue**

Type: int
Required: no
Set the value of the Secret Policy item if it requires a value, e.g. the checkout time in minutes.

**StringValue**

Type: string
Required: no
Used to set a string value of the Secret Policy Item

**SecretId**

Type: int
Required: no
Sets the Id of a Secret if required for the Secret Policy Item

**Name**

Type: string
Required: no
The name of the Secret Policy Item being used

**Description**

Type: string
Required: no
The Description of the Secret Policy Item being used

**ParentSecretPolicyItemId**

Type: int
Required: no
The name of the Secret Policy Item being used. If AutoChange is enabled then the Autochange Schedule item is a child of the Autochange item.

**SectionName**

Type: string
Required: no
The Section Name of the Secret Policy Item.

**UserGroupMap**

Zero or multiple repetitions of this can be used.

**ID**

Type: int
Required: no
The Group Id or User Id used for Secret Policy Items that require approval or privileged accounts. Zero or multiple repetitions of this can be used.

**UserGroupMapType**

Type: string
Required: no
Specifies whether the UserGroupMap is a group or user account. Use either 'Group' or 'User'

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permissions to create Secret Policies.

**"Check Out Interval is invalid."**

The Custom Checkout Interval (minutes) must be greater than 0 and Checkout must be enabled.

**"The specified Web Password Account Type does not exist or is not active."**

The user associated with the token does not have permissions to manage Active Directory Sync.

**"Approval Group is invalid."**

The Approval Group for Check Out must be a valid group in Secret Server

## AddUser

Creates a new user in Secret Server. This user can be either a local Secret Server user or a user in a domain that syncs with Secret Server.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**newUser**

Type: User
Required: yes

The new user to be created in Secret Server. The Id field should be set to null when making this call.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"You do not have permission to use this feature."**

The user associated with the token does not have the administer users application permission.

**"User Name is required."**

The userName field is null or empty.

**"Display name is Required."**

The displayName field is null or empty.

**"Invalid username."**

The username contains invalid characters.

**"Invalid email address."**

The email address entered is not in a valid email format.

**"Active Directory requires Professional Edition or higher."**

The user can not have a domain Id set if Secret Server is not at least licensed for Professional Edition.

**"Domain does not exist or is not active."**

The domain Id on the user object is not correct.

**"Password is required."**

The password field is required if the user is not a domain user.

**"User Name is already in use."**

There is already an existing user in Secret Server with that user name.

**"User licensing limit has been reached."**

Secret Server does not have any available user licenses to use.

**"RADIUS requires Professional Edition or higher."**

RADIUS cannot be enabled on a user if Secret Server is not at least licensed for Professional Edition.

**"User may not have Email Pincode and RADIUS Two Factor enabled."**

The EmailTwoFactor and RadiusTwoFactor values cannot both be set to true on the user object.

**"User must have email for Email Two Factor Authentication."**

If EmailTwoFactor is set to true, then the email address on the user object must be entered.

**"Enterprise Plus is required to create an Application Account."**

If the user has the IsApplicationAccount value set to true, Secret Server must be licensed for Enterprise Plus Edition

**“Application users may not have Two Factor enabled.”**

If the user has the IsApplicationAccount value set to true, the EmailTwoFactor and RadiusTwoFactor fields must be set to false.

## ApproveSecretAccessRequest

A member of the “Approvers” list for a Secret can use this method to approve an access request.

### Return Type

ApproveSecretAccessRequestResult

### Parameters

**approvalId**

Type: string
Required: yes
The unique identifier (GUID) for a specific user and request. This value is embedded in the link that is sent in the request for approval email.

**hours**

Type: string
Required: yes
The amount of time, in hours, for which the user will be approved to access the Secret. This value must be an integer value greater than zero.

**userOverride**

Type: boolean
Required: yes
Specify whether you wish to override an existing approval or denial for this request.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Allow Approval from Email is not enabled.”**

Allow Approval from Email is a setting under ADMIN > Configuration in the Secret Server UI that will need to be enabled to allow approval to be initiated from an email link.

**“This request cannot be processed.”**

An incorrectly formatted approvalId was submitted.

**“The Secret Access Request could not be found.”**

No access request exists for the approvalId provided, or the request was canceled.

**“You are outside of the 24 hour period during which requests may be handled through email. Please log into the user interface if you would like to make a change.”**

It has been 24 hours since the request for approval was made, after which it cannot be approved/denied via an email link.

**“The User is not enabled.”**

The user who made the original request is no longer enabled for access to Secret Server.

**“The request has already been approved.”**

The request for approval was already approved by another approver.

**“The request has already been denied.”**

The request for approval was already denied by another approver.

**“The approval period specified is invalid.”**

The hours value provided is not a valid amount of time for the approval.

**“Please enter 256 characters or less in "Reason for Request".”**

If submitting an advance approval without a request, the reason for request must contain 256 characters or less in the Reason for Request description.

**“Start date must be greater than now.”**

The start date specified in the approval must begin in the future.

**“Expiration date must be greater than two minutes from now.”**

The expiration date specified in the approval must be more than two minutes in the future.

**“Email Failed”**

An error occurred during the send process for the request notification emails. Check the email server settings under ADMIN > Configuration > Email in Secret Server and verify that the email server is currently running.

**“Email Failed: No emails are available to send a request to.”**

The requesting user doesn’t have an email address associated with their account in Secret Server.

**“Email Failed: Must have from email address and smtp server set in configuration.”**

SMTP server settings have not been configured under ADMIN > Configuration > Email.

## AssignAgent (Obsolete as of Secret Server 8.9)

This method is deprecated. It was used for a feature in Secret Server that is no longer supported.

## AssignSite

A web method that adds the specified secret policy to a specific site.

### Return Type

AssignSiteResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int
Required: yes
The Id of the Secret that the site is being set on

**siteId**

Type: int
Required: yes
The Id of the Site being used

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user authenticating to the web service does

**"This feature requires Professional Edition or higher."**

This feature requires Professional Edition or higher.

**"Engine does not exist or is not Active."**

The Site is not active or does not have an active Engine.

**"The requested Secret is Double Locked.**

A double locked secret cannot be managed through web methods.

**"This Secret is not configured for Remote Password Changing.**

Remote Password Changing must be enabled on the secret in order to use Engine.

## AssignSecretPolicyForSecret

A web method that adds the specified secret policy to a specific secret.

### Return Type

AssignSecretPolicyForSecretResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int
Required: yes
The Id of the Secret that the policy is being set on

**SecretPolicyId**

Type: int
Required: yes
The Id of the Secret Policy that is being set on the Secret.

**Inherit**

Type: boolean
Required: yes
Sets whether or not the Secret Policy should inherit from a parent folder

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user authenticating to the web service does

**"Assign Secret Policy Required"**

A secret Policy Id is required.

**"Invalid Secret Policy"**

The Secret Policy Id is required.

**"Cannot apply Secret Policy without access to required Privileged Account."**

A Privileged account is required to use this web method.

**"Cannot apply Secret Policy without access to required Associated Secret."**

The associated Secret Id is not valid

## AssignUserToGroup

Assigns an existing User to a local Secret Server Group. A user cannot be moved in to an Active Directory synchronized group. To do this, make the membership change in Active Directory and it will be reflected in Secret Server after the next user synchronization.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**userId**

Type: integer
Required: yes
The Id of the User to assign to the Group.

**groupId**

Type: integer
Required: yes
The Id of the Group to assign the User to. A list of Groups can be retrieved using the GetAllGroups method.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied."**

The authenticated user does not have the Administer Groups role permission, or is not an owner of the target group.

**"The specified group does not exist."**

The groupId does not actually map to any existing group.

**"The specified group is not valid."**

The group specified by the groupId is not for your organization or is a system group that a user cannot manage membership for.

**"Adding a user to an Active Directory Group is not supported by this method."**

The target group is an Active Directory synchronized group and membership is not managed in Secret Server.

**"The specified user does not exist."**

The userId parameter does not match a user in your organization.

**"The specified user is not valid.**

The userId parameter maps to a system user account that cannot be managed by users.

## Authenticate

This method is used to authenticate a username and password, and gives a token if the authentication was successful.

### Return Type

Token

### Parameters

**username**

Type: string
Required: yes
The username for authentication.

**password**

Type: string
Required: yes
The password for authentication.

**organization**

Type: string
Required: no
The organization code if using Secret Server Online. For installed Secret Server, you must specify null or an empty string or authentication will fail.

**domain**

Type: string
Required: no
The domain if attempting to authenticate using a domain account. For non-domain accounts, passing null, empty string, or “(local)” indicates it is not a domain account.

### Errors

**“Login failed.”**

The credentials specified are not valid, or the account is disabled, or locked out.

**“IP Address is not allowed.”**

The IP address from which the authentication is being performed is not allowed for the user. See the Secret Server User Guide for more information about IP address restrictions.

**"Secret Server web services are disabled. Please contact your Secret Server administrator."**

Web service access is not enabled. For web services to work, they must be enabled in Secret Server.

**"Web services do not work for users who have pin code two factor authentication enabled."**

Secret Server's web services do not support users that require a pin code or email-based two-factor authentication.

## AuthenticateRadius

This method is used to authenticate a username and password in addition to a radius token, and gives a token if the authentication was successful.

### Return Type

Token

### Parameters

**username**

Type: string
Required: yes
The username for authentication.

**password**

Type: string
Required: yes
The password for authentication.

**organization**

Type: string
Required: no
The organization code if using Secret Server Online. For installed Secret Server, you must specify null or an empty string or authentication will fail.

**domain**

Type: string
Required: no
The domain if attempting to authenticate using a domain account. For non-domain accounts, passing null, empty string, or "(local)" indicates it is not a domain account.

**radiusPassword**

Type: string
Required: yes
The password for the radius token.

### Errors

Errors are returned as part of the Error property of the return type.

**“Login failed.”**

The credentials specified are not valid, or the account is disabled, or locked out.

**“IP Address is not allowed.”**

The IP address from which the authentication is being performed is not allowed for the user. See the Secret Server User Guide for more information about IP address restrictions.

**"Secret Server web services are disabled. Please contact your Secret Server administrator."**

Web service access is not enabled. For web services to work, they must be enabled in Secret Server.

**"Web services do not work for users who have pin code two factor authentication enabled."**

Secret Server’s web services do not support users that require a pin code or email-based two-factor authentication.

**"AuthenticateRADIUS does not work for Online Edition."**

Radius support is not available in Secret Server Online.

## ChangePassword

This web service method is used to change the password of the current user.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**currentPassword**

Type: string
Required: yes
The current password for the user.

**newPassword**

Type: string
Required: yes
The password to be changed to.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“As an Active Directory user, you must change your Active Directory Domain password on the network/workstation.”**

Active Directory accounts cannot change their password within Secret Server. The account’s password must be changed in Active Directory, and Secret Server will begin respecting the new Active Directory password.

**“The current password and the new password must be different.”**

The newPassword and oldPassword are identical. The web method does not permit changing the password to the existing one.

**“Current password does not match user's password.”**

The oldPassword parameter does not match the user’s current password.

**“You can not change your password as it has not yet met the minimum age specified by your Administrator.”**

The password change cannot be complete because not enough time has elapsed since the last password change.

**“You cannot change your password because it has been used too recently.”**

The password change cannot be complete because the password has been used more recently than the Secret Server password policy allowed.

**“Please enter a password with symbols.”**

The newPassword requires symbols. Symbols are defined as any character that is not a-z, A-Z, or 0-9.

**“Please enter a password with lowercase letters.”**

The newPassword must contain a lowercase letter.

**“Please enter a password with uppercase letters.”**

The newPassword must contain an uppercase letter.

**“Please enter a password with numbers.”**

The newPassword must contain a number. A number is defined as 0-9.

**“Please enter a password that is at least {0} characters long.”**

The newPassword supplied is not long enough as defined by the Secret Server’s password requirements.

{0} – The number of characters that the password must meet or exceed in length.

## CheckIn

Checks a Secret in that is currently checked out by the user that is calling the web service.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret that needs to be checked in.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Secret is not CheckedOut to current user."**

The user associated with the token is not the user that has the Secret checked out.

## CheckInByKey

Checks in a Secret based off of a current launcher session Id. This is an anonymous method that can be called without having a valid web service token.

### Return Type

WebServiceResult

### Parameters

**sessionKey**

Type: string
Required: yes
The unique identifier for the current Secret Session. This value can be obtained as a command line parameter for custom launchers.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"No active Secret Session was found."**

The session key parameter that was passed in does not match a valid Secret Session.

## DeactivateSecret

A web method that allows deleting a Secret. Deletions in Secret Server are soft deletes, the Secret is not actually deleted from the system, and it is just not visible. Please refer to the Secret Server User Guide for more information about soft deletions.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret to delete.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"User does not have permission to delete secrets."**

The user associated with the token does not have the Delete Secret Role Permission.

**"Access denied"**

The user does not have Edit or Owner permission on the Secret to perform the deletion.

## Delete SSH CommandMenu

A web method that allows deleting a SSH CommandMenu. Deletions in Secret Server are soft deletes, the menu is not actually deleted from the system, and it is just not visible. Please refer to the Secret Server User Guide for more information about soft deletions.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes

The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**sshCommandMenuId**

Type: integer
Required: yes
The Id of the SSH command menu to delete.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access denied"**

The user the token belongs to doesn't have the Administer SSH Menus role permission which is required to perform the deletion.

## DenySecretAccessRequest

This method is used to deny a request for approval to access a Secret. Access to the Secret is request when GetSecret is called with the appropriate CodeResponse.

### Return Type

DenySecretAccessRequestResult

### Parameters

**approvalId**

Type: string
Required: yes
The unique identifier (GUID) for a specific user and request. This value is embedded in the link that is sent in the request for approval email.

**userOverride**

Type: boolean
Required:
Specify whether you wish to override an existing approval or denial for this request.

## DownloadFileAttachment

A web method that downloads a file attachment stored on a Secret.

### Return Type

FileDownloadResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret that the file attachment will be downloaded from.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"File attachment not found."**

There was not a valid file attachment found on the Secret.

## DownloadFileAttachmentByItemId

A web method that downloads a file attachment stored on a Secret. This is similar to the DownloadFileAttachment web service, but is meant to be used when a Secret has multiple file attachment fields. By setting the third parameter value, the user can choose which file to download.

### Return Type

FileDownloadResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret that the file attachment will be downloaded from.

**secretItemId**

Type: integer
Required: yes
The Id of the SecretItem that is storing the file attachment.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"File attachment not found."**

There was not a valid file attachment found on the Secret.

**"The Secret requires CheckOut."**

The Secret has CheckOut enabled and Secret Server was not able to automatically check out the Secret.

## ExpireSecret

ExpireSecret is used to cause the Secret to be considered expired immediately by Secret Server. Please refer to the Secret Server User Guide for more information about Secret expiration.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret to mark as expired.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have Owner permission of the Secret.

## FolderCreate

FolderCreate is used to create a new folder.

### Return Type

CreateFolderResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**name**

Type: string
Required: yes
The name of the new folder being created. The name must not contain a slash.

**parentFolderId**

Type: integer
Required: yes
The Id of the parent folder for the new folder being created. If the folder should not have a parent folder, pass negative one (-1). If this is specified, the folder being created will automatically inherit permissions from the parent folder.

**folderTypeId**

Type: integer
Required: yes
The type of the folder being created. The folder type affects the icon of the folder in Secret Server.
Possible values:

1 – Folder
2 – Customer
3 – Computer

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Invalid folder Type: Please specify 1 (Folder), 2 (Customer), or 3 (Computer)."**

The folder type specified is not valid. Please see the folderTypeId parameter for more information.

**"User does not have the Administer Folder permission."**

The user associated with the token does not have the Administer Folder Role Permission.

**"User does not have permission on the parent folder or folder does not exist."**

The user associated with the token does not have permission to the parent folder, or the parent folder does not exist.

**"User does not have permission on the parent folder."**

The user associated with the token does not have the Edit or Owner permission on the parent folder.

**“Cannot save a folder with a slash in it.”**

The folder name specified in the name parameter contains a backslash (\), which is not allowed.

## FolderExtendedCreate

FolderExtendedCreate is used to create a new Folder using a pre-built Folder object.

### Return Type

FolderExtendedCreateResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**folder**

Type: Folder (Extended)
Required: yes
The Folder object that will be created in Secret Server.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Please enter a folder name.”**

The folder name is not specified on the Folder object.

**"The folder does not exist or user does not have access."**

There is no folder associated with the ParentFolderID supplied in the Folder object or the user does not have access to the folder associated with the supplied ParentFolderID.

**"Please enter a valid parent folder ID."**

The ParentFolderID supplied as part of the Folder object is not valid.

**"You can not inherit permissions for a Root Folder."**

The user specified that the Folder should inherit permissions from its parent but did not supply a ParentFolderID.

## FolderExtendedGet

The FolderExtendedGet web method gets a specific folder by Id with advanced permissions.

### Return Type

FolderExtendedGetResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**folderId**

Type: integer
Required: yes
The Id of the folder to retrieve.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"The folder does not exist or user does not have access."**

The user associated with the current token does not permission to view the folder.

## FolderExtendedGetNew

The FolderExtendedGetNew web method returns a folder object that can be passed into the FolderExtendedCreate web method.

### Return Type

FolderExtendedGetNewResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**folderExtendedGetNewRequest**

Type: folderExtendedGetNewRequest
Required: yes
A Folder object that can be used as the basis for a Folder to create in Secret Server.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Please enter a folder name.”**

The folder name is not specified on the Folder object.

**"The folder does not exist or user does not have access."**

There is no folder associated with the ParentFolderID supplied in the Folder object or the user does not have access to the folder associated with the supplied ParentFolderID.

**"Please enter a valid parent folder ID."**

The ParentFolderID supplied as part of the Folder object is not valid.

**"You can not inherit permissions for a Root Folder."**

The user specified that the Folder should inherit permissions from its parent but did not supply a ParentFolderID.

## FolderExtendedUpdate

A web method that can update the name and parent folder information on a folder using advanced permissions.

### Return Type

FolderExtendedUpdateResult

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**folder**

Type: Folder (Extended)
Required: yes
The Folder object that will be modified in Secret Server.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“The folder does not exist or user does not have access.”**

The user associated with the token does not have access the folder, or it is not found.

## FolderGet

The FolderGet web method gets a specific folder by Id.

### Return Type

FolderGetResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**folderId**

Type: integer
Required: yes
The Id of the folder to retrieve.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"The folder does not exist or user does not have access."**

The user associated with the current token does not permission to view the folder.

## FolderGetAllChildren

A web method that returns all child folders for a particular folder.

### Return Type

GetFoldersResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**parentFolderId**

Type: integer
Required: yes
The Id of the folder for whose children the method will return.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"The folder does not exist or user does not have access."**

The user associated with the token does not have access the folder, or it is not found.

## FolderUpdate

A web method that can update the name and parent folder information on a folder.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**modifiedFolder**

Type: Folder
Required: yes
The modified folder to update.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"The folder does not exist or user does not have access."**

The user associated with the token does not have access the folder, or it is not found.

## GeneratePassword

A web method that generates a new password for a Secret field.

### Return Type

GeneratePasswordResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretFieldId**

Type: integer
Required: yes
The Id of the Secret Item that the password will be generated for. This is used to determine the password requirement and generation rules.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## GetAllGroups

A web method that returns all of the active Groups for the Secret Server installation.

### Return Type

GetAllGroupsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user authenticating to the web service does not have permission to View Groups, Administer Groups, and is not a Group Owner of any groups.

## GetAllScripts

A web method that returns all of the scripts created in Secret Server.

### Return Type

GetAllScriptsResult

### Parameters

**token**

Type: string
Required: yes

The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**includeInactiveUserScripts**

Type: boolean
Required: yes
Returns inactive scripts if set to 1.

## GetAllSSHCommandMenus

Returns all SSH Command Menu information.

### Return Type

GetSSHCommandMenusResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**includeInactive**

Type: boolean
Required: no
Setting this parameter to “true” will return both active and inactive menus. False returns only active menus.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user the token belongs to doesn’t have the Administer SSH Menus or View SSH Menus role permission.

## GetAgents – (Obsolete as of Secret Server 8.9)

A web method that returns all of the active agents for the Secret Server installation.

### Return Type

GetAgentsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Remote Agents are not enabled"**

Remote Agents are not enabled in Secret Server.

**"This feature requires Professional Edition or higher."**

Secret Server must be at least licensed for Professional Edition in order to call this web service.

**"You do not have permission to use this feature."**

The user associated with the token does not have either the View Remote Password Changing role permission or the Administer Remote Password Changing role permission.

## GetCheckoutStatus

Returns the Checkout Status of a Secret.

### Return Type

GetCheckOutStatusResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret that will be checked.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have Owner permission of the Secret.

## GetDependencies

Retrieves a list of dependencies for a Secret.

### Return Type

GetDependenciesResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The ID of the Secret from which to retrieve dependencies.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have view permission on the Secret to view dependencies.

## GetDistributedEngines

Retrieves a list of sites that host Distributed Engines.

### Return Type

GetSitesResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Sites are not enabled"**

The Distributed Engines feature is not enabled.

**"This feature requires Professional Edition or higher"**

Secret Server must be at least licensed for Professional Edition in order to call this web service.

**"You do not have permission to use this feature"**

The user associated with the token does not have permissions to view Distributed Engines.

## GetFavorites

This web method is used to get all of the favorites Secrets for the current user.

### Return Type

SearchFolderResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## GetNewSecret

A web service method that will create and return a new Secret object.

### Return Type

GetSecretResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretTypeId**

Type: integer
Required: yes
The Id of the SecretType of the new Secret that will be created.

**folderId**

Type: integer
Required: true
The Id of the folder that the new Secret will be placed in. If the Secret should not be in a folder, supply -1 or 0 for this value.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Provided Secret template is invalid.”**

The provided template Id is not valid.

**"Secret template is not in your Organization."**

The provided template Id does not exist for the organization of the user associated with the token.

**"The folder does not exist or user does not have access."**

The provided folder Id does not exist, or the user associated with the token does not have view access to the folder.

**"User does not have permission to edit the folder."**

The user associated with the token does not have create Secret permission in that folder.

## GetNewSecretPolicy

Used to get a list of all Secret Policies

### Return Type

GetNewSecretPolicyResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

## GetScript

A web method that returns the specified script created in Secret Server.

### Return Type

GetScriptResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**userScriptId**

Type: int
Required: yes
The Id of the script to return

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user authenticating to the web service does not have permission to View Groups, Administer Groups, and is not a Group Owner of any groups.

## GetSSHLoginCredentials

Used to retrieve temporary SSH credentials for connecting to Secret Server when Secret Server is running as a proxy server. This is only available for Secrets with PuTTY or a custom launcher that is a proxied SSH process. These credentials allow you to connect to Secret Server's SSH Server, and Secret Server will proxy the connection to the target on the Secret.

### Return Type

GetSSHLoginCredentialsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The ID of the Secret to get temporary proxy credentials for.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"No Secret Access"**

The user associated with the token does not have permission to view the Secret.

**"The SSH Proxy is not enabled."**

Secret Server is not using an SSH Proxy

**"No SSH Launchers have been configured for this Secret."**

There are no PuTTY or Custom SSH Launchers configured. These need to be configured so Secret Server can return the proper mapped fields to make the connection.

## GetSSHLoginCredentialsWithMachine

Used to retrieve temporary SSH credentials for connecting to Secret Server when Secret Server is running as a proxy server. This is functionally the same as GetSSHLoginCredentials except with an additional parameter to use if the Secret has an IP Address blacklist / whitelist. If there is no blacklist / whitelist on the Secret use GetSSHLoginCredentials

### Return Type

GetSSHLoginCredentialsWithMachineResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The ID of the Secret to get temporary proxy credentials for.

**machine**

Type: string
Required: yes
The IP or machine name to connect to. This is validated against the Secret's blacklist/whitelist.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"No Secret Access"**

The user associated with the token does not have permission to view the Secret.

**"The SSH Proxy is not enabled."**

Secret Server is not using an SSH Proxy

**"No SSH Launchers have been configured for this Secret."**

There are no PuTTY or Custom SSH Launchers configured. These need to be configured so Secret Server can return the proper mapped fields to make the connection.

**"The Machine field is required for Secrets with restricted host inputs."**

The machine parameter was not passed in and the Secret has a host blacklist / whitelist.

**"The Machine you are trying to access is not allowed."**

The machine passed in is not allowed based on the blacklist or whitelist on the Secret.

## GetSecret

Is primarily used to get the full information of a Secret. When this service method is used, a WEBSERVICEVIEW audit is created that can be viewed in Secret Server. This contains the full information and field values of the secret.

GetSecret is used only to provide the full details of a Secret by Secret Id. For browsing and searching Secrets, see this KB article.

If the Secret requires Check Out, Secret Server will attempt to check out the Secret for the user whom to passed token belongs to. If the Secret is checked out to a different user, an error will be indicated (see Errors section, below). The Secret will be checked back in automatically based on the configured Check Out interval for that Secret.

### Return Type

GetSecretResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int (32-bit signed integer)
Required: yes
The Id of the Secret to load. Ids can be obtained from other service methods such as SearchSecrets.

**loadSettingsAndPermissions**

Type: boolean
Required: no
Whether or not to load and populate the Secret settings and permissions on the returned Secret object.

**codeResponses**

Type: CodeResponse array
Required: no
The response to a prompt for additional security information. This includes prompts such as Require Comment.

### Errors

Errors are returned as part of the Error property of the return type. Additionally, see the Common Errors section for common web service errors.

**“No Secret exists with Id {0}.”**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked."**

The Secret Id specified is a DoubleLocked Secret. Call GetSecret once more with a CodeResponse containing ErrorCode DOUBLELOCK and a Comment to access the Secret.

**"The requested Secret requires requesting approval for access."**

The Secret Id specified requires approval for access. Call GetSecret once more with a CodeResponse containing ErrorCode APPROVAL and a Comment to submit the request.

**"The requested Secret requires a comment when viewed"**

The Secret Id specified requires a comment to be viewed. Call GetSecret once more with a CodeResponse containing ErrorCode COMMENT and a Comment to access the Secret.

**“This Secret is currently checked out to {0}. It will be available in {1}.”**

The Secret is currently checked out to another user, and cannot be accessed.
{0} – the display name of the user who the Secret is currently checked out to.

{1} – the number of minutes until the Secret is available for Check Out.

**"This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password."**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

**"Access Denied."**

The authenticated user does not have Read permission to the Secret.

## GetSecretAudit

Returns the Audit records for the provided SecretId.

### Return Type

GetSecretAuditResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int
Required: yes
The Id of the Secret to get the audit for. Ids can be obtained from other service methods such as SearchSecrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"User does not have permission to view audit Secret data."**

The user associated with the token does not have the View Secret Audit role permission.

## GetSecretItemHistoryByFieldName

Use this method to retrieve the history (past values) that were audited for a specific field of a Secret.

### Return Type

GetSecretItemHistoryByFieldNameResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
ID of the Secret that contains the target field.

**fieldDisplayName**

Type: string
Required: yes
Display name of the target Secret field.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"You do not have access to view this Secret."**

The user the token belongs to does not have permissions to view the Secret indicated by secretId.

**"Secret Field Does Not Exist"**

The Secret field specified by fieldDisplayName is not found on the Secret.

## GetSecretLegacy

Is primarily used to get the full information of a Secret. When this service method is used, a WEBSERVICEVIEW audit is created that can be viewed in Secret Server. This contains the full information and field values of the secret.

GetSecret is used only to provide the full details of a Secret by Secret Id. For browsing and searching Secrets, see SearchSecrets.

Currently, GetSecret does not support viewing secrets that are DoubleLocked.

If the Secret requires Check Out, Secret Server will attempt to check out the Secret for the user whom to passed token belongs to. If the Secret is checked out to a different user, an error will be indicated (see section). The Secret will be checked back in automatically based on the configured Check Out interval for that Secret.

### Return Type

GetSecretResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int (32-bit signed integer)
Required: yes
The Id of the Secret to load. Ids can be obtained from other service methods such as SearchSecrets.

### Errors

Errors are returned as part of the Error property of the return type. Additionally, see the Common Errors section for common web service errors.

**“No Secret exists with Id {0}.”**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked and can not be accessed through a web service request."**

The Secret Id specified is a DoubleLocked Secret. DoubleLocked Secrets are not accessible via web services.

**"The requested Secret requires requesting approval for access and can not be accessed through a web service request."**

The Secret Id specified requires approval for access. Secrets requiring approval for access are not accessible via web services.

**"The requested Secret requires a comment when viewed and can not be accessed through a web service request."**

The Secret Id specified requires a comment to be viewed. Secrets requiring a comment are not accessible via web services.

**“This Secret is currently checked out to {0}. It will be available in {1}.”**

The Secret is currently checked out to another user, and cannot be accessed.
{0} – the display name of the user who the Secret is currently checked out to.
{1} – the number of minutes until the Secret is available for Check Out.

**“This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password.”**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

**"Access Denied."**

The authenticated user does not have Read permission to the Secret.

## GetSecretPolicyForSecret

A web method that returns the specified secret policy for a specific secret.

### Return Type

GetSecretPolicyForSecretResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: int
Required: yes
The Id of the secret to return

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user authenticating to the web service does

## GetSecretTemplateFields

This web method returns the Secret template field information for a particular template.

### Return Type

GetSecretTemplateFieldsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretTypeId**

Type: integer
Required: yes
The Id of the template whose fields are to be returned.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## GetSecretTemplates

This web service method is used to retrieve all active Secret templates, their fields, and other data about the template.

### Return Type

GetSecretTemplatesResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## GetSecretsByExposedFieldValue

Searches for Secrets that match a field name / search term but only on Secret Fields marked **Exposed for Display** on the Secret Template. This will return all Secrets that contain a field with the specified name and have a value in that field that contains the search term.

### Return Type

GetSecretsByFieldValueResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**fieldName**

Type: string
Required: yes
The name of the Secret Field to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be userName.

**fieldSearchTerm**

Type: string
Required: yes
The value to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be admin.

**showDeleted**

Type: boolean
Required: no
Whether or not the resulting Secrets will include deleted Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user associated with the token does not have permission to view Secrets.

## GetSecretsByFieldValue

Searches for Secrets that match a field name / search term. This will return all Secrets that contain a field with the specified name and have a value in that field that contains the search term.

### Return Type

GetSecretsByFieldValueResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**fieldName**

Type: string
Required: yes

The name of the Secret Field to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be userName.

**fieldSearchTerm**

Type: string
Required: yes
The value to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be admin.

**showDeleted**

Type: boolean
Required: no
Whether or not the resulting Secrets will include deleted Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user associated with the token does not have permission to view Secrets.

## GetSSHCommandMenu

Returns SSH Command Menu information.

### Return Type

GetSSHCommandMenuResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**sshCommandMenuId**

Type: integer
Required: yes
ID of the SSH command menu to retrieve information for.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user the token belongs to doesn’t have the Administer SSH Menus or View SSH Menus role permission.

## GetSSHCommandMenusForUserBySecret

Returns all SSH Command Menu information for a single secret for which a particular user has permissions.

### Return Type

GetSSHCommandMenusResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
ID of the secret that is associated with the SSH command menus.

**userId**

Type: integer
Required: yes
ID of the user that has access to the SSH command menus.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user the token belongs to doesn't have the Administer SSH Menus or View SSH Menus role permission.

## GetTicketSystems

Used to get a list of all active Ticket Systems.

### Return Type

### Parameters

**token**

Type: string
Required: yes

The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

## GetTokenIsValid

Used to determine if an existing token is still valid. If the Errors collection of the return type is empty, then the token is valid.

### Return Type

CodeResponse

### Parameters

**token**

Type: string
Required: yes
The token to check if it is still valid.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user the token belongs to doesn't have the Administer SSH Menus or View SSH Menus role permission.

## GetUser

Returns Secret Server user information.

### Return Type

GetUserResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**userId**

Type: integer
Required: yes
ID of the user to retrieve information for.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user the token belongs to doesn't have the Administer Users or View Users role permission.

**"The specified user does not exist."**

The userId specified does not match an existing Secret Server user ID.

## ImpersonateUser

Provides the user with a token that can be used to execute web services as a different user.

### Return Type

ImpersonateResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**username**

Type: string
Required: yes
The Username of a user to impersonate.

**organization**

Type: string
Required: yes
The organization code if using Secret Server Online. For installed Secret Server, you must specify null or an empty string or authentication will fail.

**domain**

Type: string
Required: yes
The domain if attempting to authenticate using a domain account. For non-domain accounts, passing null, empty string, or "(local)" indicates it is not a domain account.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied. You do not have the Web Services Impersonate role permission."**

The user associated with the token does not have the Web Services Impersonate role permission.

**"You cannot impersonate on behalf of someone else."**

The Token provided was for an impersonated user. Double-impersonation is not allowed.

**"The specified domain is not a valid domain. Try entering the Fully Qualified Domain Name or following the steps in this KB article: https://updates.thycotic.net/link.ashx?SSDomainNotValidError"**

Nothing was entered for the domain parameter.

**"The specified user does not exist."**

The username specified does not connect to a valid Secret Server user.

**"You cannot impersonate yourself."**

The username specified is the same as the one calling the web method.

**"This application's status is pending or rejected by the impersonated user. The user must approve it through the web interface."**

The device or calling application being used to call the web method has not been approved.

## ImportXML

Runs the advanced xml import with the provided xml.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**xml**

Type: string
Required: yes
The XML structure to import. For more information on the advanced XML Import, see: http://support.thycotic.com/KB/a160/advanced-import-with-xml.aspx

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"You do not have permission to use this feature."**

The user associated with the token does not have the Advanced Import role permission.

**"You do not have a valid license to use this feature. Please contact sales to purchase a license."**

Free Secret Server Online users cannot use this feature.

**“The uploaded file was not valid Xml”**

The XML parameter was not valid for the advanced import. There may be several additional errors having to do with permissions on Secrets, folder, groups, etc. In the case of these types of errors, the remaining XML structure that does not cause errors will still be imported.

## RemoveDependency

Removes a dependency from a secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**dependencyId**

Type: integer
Required: yes
The ID of the dependency to remove. The ID can be obtained from GetDependencies.

**secretId**

Type: integer
Required: yes
The ID of the secret to remove the dependency from.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user associated with the token does not have view permission on the Secret to remove dependencies.

**“Access Denied - User does not own Secret”**

The user associated with the token does not have owner permission on the Secret to remove dependencies.

**“Dependency not found on Secret”**

The dependencyId does not exist for the secretId specified.

## RestoreSSHCommandMenu

A web method that restores a “deleted” SSHCommandMenu to an active state. Deletions in Secret Server are soft deletes, the menu is not actually deleted from the system, and it is just not visible. Please refer to the Secret Server User Guide for more information about soft deletions.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**sshCommandMenuId**

Type: integer
Required: yes
The Id of the SSH command menu to delete.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access denied”**

The user the token belongs to doesn’t have the Administer SSH Menus role permission which is required to perform the restore.

## RunActiveDirectorySynchronization

Forces Secret Server to sync with Active Directory domains.

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied”**

The user associated with the token does not have view permission on the Secret to remove dependencies.

## SearchFolders

This web method allows the user to search for folders in the system, filtering by folder name.

### Return Type

SearchFolderResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**Term**

Type: string
Required: yes
The search term to search for. All folders that the user has access to and have names containing the search term will be returned.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SaveSSHCommandMenu

A web method that saves a SSHCommandMenu. To add a new menu, set the sshCommandMenuId field to -1.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**sshCommandMenu**

Type: SSHCommandMenu
Required: yes
The SSH Command Menu object to be saved. Setting the sshCommandMenuId to -1 denotes an “add,” otherwise the system will update by sshCommandMenuId.

**commandsText**

Type: string
Required: yes
A text representation of the commands belonging to the SSH Command Menu. The format is the command name, then an equal sign (=), followed by the SSH command itself. Each command is separated by a newline (\r\n). The SSH commands belonging to this SSH command menu will be entirely replaced by the commands in this parameter.

**deleteCommands**

Type: boolean
Required: yes
Setting this parameter to “true” will result in the deletion of all commands, regardless of the contents of the commandsText parameter.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access denied”**

The user the token belongs to doesn’t have the Administer SSH Menus role permission which is required to perform the save.

## SearchSecretPolicies

This web service method is used to search Secret Server Policies.

SearchSecretPolicyResults

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: no
The term to search Secret Policies for. This will be used to search the Secret's name, and optionally the values of the policies  depending on the Searchable status of the field. Not specifying a search term results in all Policies being returned. Returning all Policies should be avoided as this can be a large response.

**includeInactive**

Type: boolean
Determines whether or not to return deleted Secrets as results.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SearchSecrets

This web service method is used to search Secret Server, or get all Secrets. This only returns summary information of the Secrets, and does not generate view audits of the Secrets. To get the full information of the Secret, pass the Secret Id of the summary to GetSecret.

### Return Type

SearchSecretsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: no
The term to search Secrets for. This will be used to search the Secret's name, and optionally the values of the Secret depending on the Searchable status of the field. Not specifying a search term results in all Secrets being returned. Returning all Secrets should be avoided as this can be a large response.

**includeDeleted**

Type: boolean
Determines whether or not to return deleted Secrets as results.

**includeRestricted**

Type: boolean
Determines whether or not to return Secrets that have security features such as “DoubleLock,” “Require Comment” and “Requires Approval for Access” enabled.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SearchSecretsByExposedFieldValue

Searches for Secrets that match a field name / search term but only on Secret Fields marked **Exposed for Display** on the Secret Template. This will return all Secrets that contain a field with the specified name and have a value in that field that contains the search term.

### Return Type

SearchSecretsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**fieldName**

Type: string
Required: yes
The name of the Secret Field to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be userName.

**searchTerm**

Type: string
Required: yes
The value to search for. For Example: If searching for Secrets with a userName field value of ‘admin’, this value would be admin.

**showDeleted**

Type: boolean
Required: yes
Whether or not the resulting Secrets will include deleted Secrets.

**showRestricted**

Type: boolean
Required: yes
Whether or not the resulting Secrets will include Require View Comment and Require Approval For Access Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permission to view Secrets.

## SearchSecretsByExposedFieldValues

Searches for Secrets across fields with a search term but only on Secret Fields marked **Exposed for Display** on the Secret Template. This will return all Secrets that have a value in that field that contains the search term.

### Return Type

SearchSecretsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: yes
The value to search for. For Example: If searching for Secrets with a userName field value of 'admin', this value would be admin.

**showDeleted**

Type: boolean
Required: yes
Whether or not the resulting Secrets will include deleted Secrets.

**showRestricted**

Type: boolean
Required: yes

Whether or not the resulting Secrets will include Require View Comment and Require Approval For Access Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permission to view Secrets.

## SearchSecretsByFieldValue

Searches for Secrets that match a field name / search term. This will return all Secrets that have a field that is an exact match of the fieldName value and that field has a value that is an exact match of the fieldSearchTerm parameter.

### Return Type

SearchSecretsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**fieldName**

Type: string
Required: yes
The name of the Secret Field to search for. For Example: If searching for Secrets with a userName field value of 'admin', this value would be userName.

**fieldSearchTerm**

Type: string
Required: yes
The value to search for. For Example: If searching for Secrets with a userName field value of 'admin', this value would be admin.

**showDeleted**

Type: boolean
Required: yes
Whether or not the resulting Secrets will include deleted Secrets.

**showRestricted**

Type: boolean
Required: yes
Whether or not the resulting Secrets will include Require View Comment and Require Approval For Access Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permission to view Secrets.

## SearchSecretsByFolder

Web method that searches for Secrets within a folder.

### Return Type

SearchSecretsResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: yes
The search term that will be used to match against Secrets.

**folderId**

Type: integer
Required: no
The folder to search for Secrets in.

**includeSubFolders**

Type: boolean
Required: yes
Whether or not to include Secrets that are in descending folders from the search folder.

**includeDeleted**

Type: boolean
Required: no
Whether or not to include deleted Secrets in the results.

**includeRestricted**

Type: boolean
Required: no
Whether or not to include deleted Secrets that have Require View Comment and Require Approval For Access Secrets.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied"**

The user associated with the token does not have permission to view Secrets.

## SearchSecretsByFolderLegacy

Legacy method that searches for Secrets within a folder.

### Return Type

SearchSecretsByFolderLegacyResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: yes
The search term that will be used to match against Secrets.

**folderId**

Type: integer
Required: no
The folder to search for Secrets in. This value can be null.

**includeSubFolders**

Type: boolean
Required: yes
Whether or not to include Secrets that are in descending folders from the search folder.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SearchSecretsLegacy

Legacy method that searches for Secrets within a folder.

### Return Type

SearchSecretsLegacyResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: no
The search term that will be used to match against Secrets.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SearchUsers

Allows searching for a user by search term.

### Return Type

SearchUsersResult

### Parameters

**token**

Type: string
Required: yes

The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**searchTerm**

Type: string
Required: yes
The search term that will be used to match against users.

**includeInactiveUsers**

Type: boolean
Whether or not to include inactive users in the search results.

### Errors

Please see the Common Errors section for details about errors on the Error collection.

## SearchWebPasswordsForUrl (Deprecated)

This method is deprecated. It was used for a feature in Secret Server that is no longer supported. Please use either SearchSecretsByFieldValue or GetSecretsByFieldValue to search for URL values in Secrets.

## SetCheckoutEnabled

This web method can set the whether or not checkout is enabled on a Secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret that needs to be updated.

**setCheckOut**

Type: boolean
Required: yes
Whether or not the Secret will have CheckOut enabled.

**setPasswordChangeOnCheckin**

Type: boolean
Required: yes
Whether or not the Secret's password field will be reset when the Secret is checked in.

**checkOutInterval**

Type: integer
Required: false
The amount of time that the Secret will be checked out before it is checked in. If this value is null, the interval will be defaulted from the template.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Access Denied."**

The user associated with the token does not have Owner permission of the Secret.

**"Check Out must be enabled on the Remote Password Changing Configuration page for change password on check in. Please contact your administrator."**

Checkout is not enabled globally in Secret Server.

**"No Secret exists with Id {0}."**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked and can not be accessed through a web service request."**

The Secret Id specified is a DoubleLocked Secret. DoubleLocked Secrets are not accessible via web services.

**"The requested Secret requires requesting approval for access and can not be accessed through a web service request."**

The Secret Id specified requires approval for access. Secrets requiring approval for access are not accessible via web services.

**"The requested Secret requires a comment when viewed and can not be accessed through a web service request."**

The Secret Id specified requires a comment to be viewed. Secrets requiring a comment are not accessible via web services.

**"This Secret is currently checked out to {0}. It will be available in {1}."**

The Secret is currently checked out to another user, and cannot be accessed.
{0} – the display name of the user who the Secret is currently checked out to.
{1} – the number of minutes until the Secret is available for Check Out.

**"This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password."**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

## UpdateIsFavorite

Updates a Secret to set it or remove it as a user's favorite.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: secretId
Required: yes
The Secret that needs to be updated.

**isFavorite**

Type: boolean
Required: yes
Whether or not the Secret will be flagged as a favorite for the user associated with the token.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"Security Exception."**

The user associated with the token does not have access to the Secret.

## UpdateScript

This web method is used to update a PowerShell, SSH, or SQL Script.

### Return Type

UpdateScriptResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**ScriptId**

Type: integer
Required: yes
The Id of the script that will be updated.

**ScriptName**

Type: string
Required: yes
The name of the Script.

**ScriptDescription**

Type: string
Required: no
A Description of the Script being created.

**Active**

Type: boolean
Required: yes
Sets whether or not the Script is Active

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

This web method may throw an exception, resulting in the server returning an HTTP 500 error code if the user does not have permission to create a Secret due to Role permissions.

**"Access Denied."**

The user account does not have permissions to update scripts.

## UpdateSecret

The UpdateSecret web method is used to update an existing Secret. This will create a WEBSERVICEUPDATE audit record.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secret**

Type:
Required: yes
A Secret with the updated values. Using  to get an existing Secret, then updating its values and passing to this web method is the recommended approach to updating a Secret.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

This web method may throw an exception, resulting in the server returning an HTTP 500 error code if the user does not have permission to create a Secret due to Role permissions.

**"Secret name exceeds max length ({0} characters)."**

The Secret name specified exceeds the maximum length.
{0} – The number of characters that the Secret name cannot exceed.
As of Secret Server 4.0, the name limit was 1991 characters.

**"Secret name cannot be empty."**

The supplied Secret name is zero characters in length. White space is ignored when checking if the Secret name is empty.

**"Secret item value exceeds max length ({0} characters)."**

A Secret item supplied in the secretItemValues exceeds the maximum length.
{0} – The number of characters that the Secret item cannot exceed.
As of Secret Server 7.8.000036, the item length limit is 10,000 characters. Prior to that version, the limit is 1991 characters.

**"The folder does not exist or user does not have access."**

The folderId supplied does not exist.

**"User does not have permission to edit the folder."**

The user associated with the current token does not have Edit or Owner permission on the folder specified.

**"Provided Secret type is invalid."**

The supplied secretTypeId is not a valid Secret template Id.

**"Secret template is out of date."**

The number of Secret template fields specified in the secretFieldIds parameter does not match the number of fields on the template, or the secretFieldsIds were not passed in the order that they are defined on the Secret template.

**"Password does not meet the minimum length requirement of {0}."**

A Secret item supplied in the secretItemValues parameter that is a password does not meet minimum length requirements as defined on the Secret template.

{0} – The minimum length as defined by the Secret template

**"Password does not meet the complexity requirements."**

A Secret item supplied in the secretItemValues parameter that is a password does not meet complexity requirements as defined on the Secret template.

## UpdateSecretPermission

Updates a user or group's access to a Secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret to be updated.

**GroupOrUserRecord**

Type: GroupOrUserRecord
Required: yes
The group or user to update permissions on the Secret for.

**view**

Type: boolean
Required: yes
Whether or not the group or user should have view access to the Secret.

**edit**

Type: boolean
Required: yes
Whether or not the group or user should have edit access to the Secret. Edit grants view.

**owner**

Type: boolean
Required: yes
Whether or not the group or user should have owner access to the Secret. Owner grants edit and view.

## Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“Access Denied.”**

The user associated with the token does not have Owner permission of the Secret.

**“No Secret exists with Id {0}”**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked and can not be accessed through a web service request."**

The Secret Id specified is a DoubleLocked Secret. DoubleLocked Secrets are not accessible via web services.

**"The requested Secret requires requesting approval for access and can not be accessed through a web service request."**

The Secret Id specified requires approval for access. Secrets requiring approval for access are not accessible via web services.

**"The requested Secret requires a comment when viewed and can not be accessed through a web service request."**

The Secret Id specified requires a comment to be viewed. Secrets requiring a comment are not accessible via web services.

**“This Secret is currently checked out to {0}. It will be available in {1}.”**

The Secret is currently checked out to another user, and cannot be accessed.
{0} – the display name of the user who the Secret is currently checked out to.

{1} – the number of minutes until the Secret is available for Check Out.

**“This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password.”**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

## UpdateUser

Use this method to change settings for a Secret Server user account.

### Return Type

UpdateUserResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**user**

Type: User
Required: yes
The user information to be updated. Must include an existing user ID.

### Errors

**"Access Denied"**

The user the token belongs to does not have the appropriate role permission to administer users.

**"Updated user does not have ID specified. Cannot find user to update."**

The Id from the user parameter has not been provided.

**"The specified user does not exist."**

The Id from the user parameter does not match an existing user id in Secret Server.

**"Updating UserName is not allowed on existing user accounts."**

You cannot update the username for existing Secret Server users.

**"Updating DomainId is not allowed on existing user accounts."**

You cannot update the domain for existing Secret Server users.

**"Cannot set a domain user as a Secret Server application account."**

An application account cannot be a domain user.

**"Cannot set passwords on domain user accounts, domain user authentication is managed through the directory."**

A domain account cannot have a password set; the password is validated against the domain during authentication.

## UploadFileAttachment

This web method uploads a file attachment to a Secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret to be updated.

**fileData**

Type: byte array
SOAP Type: base64Binary
Required: yes
The binary data of the file to be uploaded and added to the Secret.

**fileName**

Type: string
Required: yes
The name of the uploaded file.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**"File attachment not found."**

There was not a valid File Attachment field on the Secret that was provided.

**"Secret has multiple file attachment fields. Please upload/download the file by including the Item Id."**

Use the UploadFileAttachmentByItemId web method.

**"Access Denied."**

The user associated with the token does not have edit permission on the Secret.

**"No Secret exists with Id {0}."**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked and can not be accessed through a web service request."**

The Secret Id specified is a DoubleLocked Secret. DoubleLocked Secrets are not accessible via web services.

**"The requested Secret requires requesting approval for access and can not be accessed through a web service request."**

The Secret Id specified requires approval for access. Secrets requiring approval for access are not accessible via web services.

**"The requested Secret requires a comment when viewed and can not be accessed through a web service request."**

The Secret Id specified requires a comment to be viewed. Secrets requiring a comment are not accessible via web services.

**“This Secret is currently checked out to {0}. It will be available in {1}.”**

The Secret is currently checked out to another user, and cannot be accessed.

{0} – the display name of the user who the Secret is currently checked out to.

{1} – the number of minutes until the Secret is available for Check Out.

**“This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password.”**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

## UploadFileAttachmentByItemId

This web method uploads a file attachment to a specific field on a Secret.

### Return Type

WebServiceResult

### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

**secretId**

Type: integer
Required: yes
The Id of the Secret to be updated.

**secretItemId**

Type: integer
Required: yes
The Id of the Secret Item to upload the file attachment to.

**fileData**

Type: byte array
SOAP Type: base64Binary
Required: yes
The binary data of the file to be uploaded and added to the Secret.

**fileName**

Type: string
Required: yes
The name of the uploaded file.

### Errors

In addition to the errors below, please see the Common Errors section for additional error messages.

**“File attachment not found."**

There was not a valid File Attachment field on the Secret that was provided.

**“Access Denied”**

The user associated with the token does not have edit permission on the Secret.

**“No Secret exists with Id {0}.”**

The Secret Id specified does not exist.
{0} – The Id specified in GetSecret.

**"The requested Secret is DoubleLocked and can not be accessed through a web service request."**

The Secret Id specified is a DoubleLocked Secret. DoubleLocked Secrets are not accessible via web services.

**"The requested Secret requires requesting approval for access and can not be accessed through a web service request."**

The Secret Id specified requires approval for access. Secrets requiring approval for access are not accessible via web services.

**"The requested Secret requires a comment when viewed and can not be accessed through a web service request."**

The Secret Id specified requires a comment to be viewed. Secrets requiring a comment are not accessible via web services.

**“This Secret is currently checked out to {0}. It will be available in {1}.”**

The Secret is currently checked out to another user, and cannot be accessed.
{0} – the display name of the user who the Secret is currently checked out to.

{1} – the number of minutes until the Secret is available for Check Out.

**"This Secret is currently checked out to {0}, but the system has been attempting to check it in for {1} minute(s). There may be an issue with changing the password."**

The Secret is currently checked out to another user, and cannot be accessed, however the user has since checked it in but was unable to update the password. This many indicate a Remote Password Change failure. See the Remote Password Change logs for more information.

## ValidateTwoFactor

Enables a user to authenticate against Windows Integrated web services with Radius, Duo, or Google Two Factor pin codes. Once you validate your 2FA pin code you can make subsequent calls to windows authenticated web services as normal.

Windows Integrated web services use the user identity instead of returning a token to pass into calls. To use 2FA with Windows Integrated web services, call ValidateTwoFactor with your pin code and your client will be allowed access for a length of time, based the configuration setting for remember two factor or session API timeout. If your session API timeout is greater than a day, you will be prompted for 2FA at least once every 24 hours.

### Return Type

WebServiceResult

### Parameters

**pinCode**

Type: string
Required: yes
The current pin code displayed by the Two Factor Authentication solution.

### Errors

This web method will throw exceptions if Secret Server is unable to authenticate the user with the provided pin code.

## VersionGet

Returns the current Version of Secret Server.

### Return Type

VersionGetResult

### Parameters

There are no parameters.

#### Errors

This web method will throw exceptions if Secret Server is unable to connect to the database.

### WhoAmI

Returns the current user's info.

#### Return Type

UserInfoResult

#### Parameters

**token**

Type: string
Required: yes
The token to identify the user making the request. See Concepts: Token for more information about tokens and how to obtain one.

#### Errors

This web method will throw exceptions if Secret Server is unable to connect to the database.

## Web Service Types

### AdditionalDependencyInfoJson

AdditionalDependencyInfoJson is a type used to represent additional information for Dependency.

#### Members

**Regex**

Type: string
The regular expression to be used if the Dependency type is a 4, a File regular expression.

**PowershellArguments**

Type: string
The arguments to pass to the PowerShell script if the Dependency type is 7, a PowerShell script.

### AddScriptResult

AddScriptResult is the result of the AddScript web method. If the web method was successful, it also contains information about the Script that was created.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while adding a Secret.

#### Script

Type: Script
Information about the Script that was added.

## AddSecretPolicyResult

AddSecretPolicyResult is the result of the AddSecretPolicy web method. The status or result of a request for access to a Secret.

### Members

#### SecretPolicyId

The Id of the created Secret Policy

#### SecretPolicyName

Type: string
The name of the Secret Policy

#### SecretPolicyDescription

Type: string
The name of the SecretPolicy

#### Active

Type: boolean
Whether or not the Secret Policy is active.

#### SecretPolicyItems

Type: SecretPolicyItems array
An array of the Secret Policy Items added to the created Secret Policy.

## AddSecretResult

AddSecretResult is the result of the AddSecret web method. If the web method was successful, it also contains information about the Secret that was created.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while adding a Secret.

#### Secret

Type: Secret
Information about the Secret that was added. See the type for more information.

## ApprovalInfo

The status or result of a request for access to a Secret.

### Members

#### Status

Type: Pending, Approved, Denied or Canceled
The status of the Secret access request.

#### Responder

Type: string
The user who responded to the request.

#### ResponseDate

Type: dateTime
Date and time the request was responded to.

#### ResponseComment

Type: string
Comment made by the responder when responding to the request.

#### ExpirationDate

Type: dateTime
Date and time the requester's access to the Secret will expire.

## ApproveSecretAccessRequestResult

The status or result of a request for access to a Secret.

### Members

#### ApprovalInfo

Type: ApprovalInfo

Contains the information pertaining to the access request status.

## AssignSiteResult

This is the result when using the AssignSite method.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

## AssignSecretPolicyForSecretResult

This is the result when using the AssignSecretPolicyForSecret method. This contains all relevant information of the Secret Policy on the specific secret.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### SecretId

Type: int
The Id of the returned secret

#### SecretPolicyId

Type: int
The Secret Policy Id set on the secret

#### Inherit

Type: boolean
Whether or not the secret is inheriting a policy from the parent folder.

## AuditSecret

An audit record for a Secret.

## Members

**AuditSecretId**

Type: integer
The Id of the audit record.

**SecretId**

Type: integer
The Id of the Secret that this audit is associated with.

**DateRecorded**

Type: dateTime
The date that the audit was created.

**Action**

Type: string
The Action value of the audit.

**Notes**

Type: string
The Notes value of the audit.

**UserId**

Type: integer
The Id of the user that created the audit.

**SecretName**

Type: string
The name of the Secret that this audit is associated with.

**IpAddress**

Type: string
The IP address that the audit was created from.

**ReferenceId**

Type: integer
The Id that is associated with the request in a Request to Join approval or denial audit.

**ByUserDisplayName**

Type: string
The display name of the Secret that this audit is associated with.

**TicketNumber**

Type: string
The ticketing system’s ticket number that is associated with the audit.

## AuthenticateResult

Used to describe an authentication result.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### Token

Type: string
An authentication token used for other web services methods. In errors occurred, this will be empty.

## CodeResponse

CodeResponse is the response from a prompt for require comment before viewing a Secret.

### Members

#### ErrorCode

Type: string
The Error code, informing the user what type of issue prevented the Secret from being opened. Possible options are: NoError, LOAD, DOUBLELOCK, COMMENT, CHECKOUT, APPROVAL, DELETED.

#### Comment

Type: string
The response comment from the user.

#### AdditionalComment

Type: string
The additional response comment from the user. This is used when Secret Server is configured to integrate with a ticketing system.

#### TicketSystemId

Type:Integer
Required: false
If provided in a CHECKOUT or APPROVAL request this value will specify which configured ticket system will be validated against.  If this value is not provided then the default ticket system will be used for validation.

## CreateFolderResult

CreateFolderResult is the result of the FolderCreate web method.

### Extension

CreateFolderResult is an extension of the WebServiceResult type. See the WebServiceResult type for additional members.

### Members

**FolderId**

Type: integer
The Id of the folder that was just created.

## DenySecretAccessRequestResult

Type resulting from a denial of a Secret access request using the DenySecretAccessRequest method.

### Members

**ApprovalInfo**

Type: ApprovalInfo
Contains the information pertaining to the access request status.

## Dependency

A dependency is added to a Secret for password changing.

### Members

**SecretId**

Type: integer
The ID of the Secret that the dependency belongs to.

**SecretDependencyTypeId**

Type: integer
An ID of the dependency type. Valid values include:

1=An IIS application pool
2=A Windows Service
3=A Scheduled Task
4=File Regular Expression
5=COM+
6=Application Pool recycle
7=PowerShell script

**MachineName**

Type: string
The target machine the dependency resides on.

**ServiceName**

Type: string
The name of the remote dependency. For example, the name of the IIS application pool or the name of the Windows service or scheduled task.

**PrivilegedAccountSecretId**

Type: integer
The ID of the Secret that will be used as a privileged account for changing the dependency.

**Active**

Type: boolean
True if the dependency is active, otherwise false.

**RestartOnPasswordChange**

Type: boolean
True if the dependency should be restarted after the password is updated, if supported. Otherwise false.

**WaitBeforeSeconds**

Type: integer
The number of seconds to wait before attempting to update the dependency.

**AdditionalInfo**

Type: AdditionalDependencyInfoJson
Additional information depending on the dependency type. This should be null if additional dependency information is not needed. See the definition for more details.

**Description**

Type: string
A description of the dependency.

**PowershellScriptId**

Type: integer
An ID of a PowerShell script if the dependency type is a PowerShell script. Use “0” to indicate no PowerShell script for other dependency types.

**SecretDependencyId**

Type: integer
An unique ID of the dependency. Set this to “0” if this dependency is being added.

## FileDownloadResult

FileDownloadResult is a generic result of a web method that does not return any information except for Errors.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during the operation.

#### FileAttachment

Type: byte[]
SOAP Type: base64Binary
The binary data of the file attachment.

#### FileName

Type: string
The name of the downloaded file.

## Folder

Contains information about an individual folder.

### Members

#### Id

Type: integer
The Id of the folder.

#### Name

Type: string
The Name of the folder. This does not include the full path of the folder.

#### TypeId

Type: integer
The type of folder. This is used to determine which icon is displayed for the folder in Secret Server.
Possible values:

- 1 – Folder
- 2 – Customer
- 3 – Computer

#### ParentFolderId

Type: integer
The Id of the parent folder. If the folder is a root folder, this value is negative one (-1).

## Folder (Extended)

Contains information about an individual folder.

### Extension

Folder (Extended) is an extension of the Folder type. See the Folder type for additional members.

### Members

#### Permissions

Type: FolderPermissions
The permissions applied to the folder.

#### Settings

Type: Settings (Folder)
The settings applied to the folder.

## FolderExtendedCreateResult

FolderExtendedCreateResult is the result of the FolderExtendedCreate web method.

### Extension

FolderExtendedCreateResult is an extension of the FolderExtendedResultBase type. See the FolderExtendedResultBase type for additional members.

### Members

#### Folder

Type: Folder
The information about the folder.

## FolderExtendedGetNewRequest

FolderExtendedGetNewRequest is an object that can be passed to the FolderExtendedGetNew web method.

### Members

#### FolderName

Type: string
Required: yes
The Name of the folder. This does not include the full path of the folder.

#### ParentFolderId

Type: integer
Required: no
The Id of the parent folder. If the folder is a root folder, this value is negative one (-1).

#### InheritPermissions

Type: Boolean

Required: no
Indicates whether or not the Secret inherits permissions from its parent folder.

## FolderExtendedGetResult

FolderExtendedGetResult is the result of the FolderExtendedGet web method.

### Extension

FolderExtendedCreateResult is an extension of the FolderExtendedResultBase type. See the FolderExtendedResultBase type for additional members.

### Members

#### Folder

Type: FolderExtended
The information about the folder.

## FolderExtendedGetNewResult

FolderExtendedGetNewResult is the result of the FolderExtendedGetNew web method.

### Extension

FolderExtendedCreateResult is an extension of the FolderExtendedResultBase type. See the FolderExtendedResultBase type for additional members.

### Members

#### Folder

Type: FolderExtended
The information about the folder.

## FolderExtendedResultBase

FolderExtendedResultBase is the base type for the results returned by the Extended Folder WebServices

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the folder.

#### Success

Type: boolean
Indicates whether or not the operation was successful.

## FolderExtendedUpdateResult

FolderExtendedUpdateResult is the result of the FolderExtendedUpdate web method.

### Extension

FolderExtendedCreateResult is an extension of the FolderExtendedResultBase type. See the FolderExtendedResultBase type for additional members.

## FolderGetResult

FolderGetResult is the result of the FolderGet web method.

### Members

#### Folder

Type: Folder
The information about the folder.

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the folder.

#### Success

Type: boolean
Indicates whether or not the operation was successful.

## FolderPermission

FolderPermission contains the UserOrGroup for a permission and the Folder Access Role and Secret Access Role in both integer and string formats.

### Members

#### UserOrGroup

Type: GroupOrUserRecord
The User or Group to whom the permission applies.

#### FolderAccessRoleName

Type: string

The name of the Folder Access Role granted as part of this permission.

**FolderAccessRoleId**

Type: integer
The ID of the Folder Access Role granted as part of this permission.

**SecretAccessRoleName**

Type: string
The name of the Secret Access Role granted as part of this permission.

**SecretAccessRoleId**

Type: integer
The ID of the Secret Access Role granted as part of this permission.

## FolderPermissions

FolderPermissions contains boolean values indicating whether permissions have been changed and whether inherit permissions is enabled as well as an array of FolderPermission objects

### Members

**IsChangeToPermissions**

Type: boolean
Indicates whether a change is being made to the Folder permissions.

**InheritPermissionsEnabled**

Type: boolean
Indicates whether or not the Folder inherits permissions from its parent.

**Permissions**

Type: FolderPermission array
A complex type of a sequence of FolderPermissions with an unbounded number of occurrences.

## GeneratePasswordResult

GeneratePasswordResult is a result of a web method that generates a new random password.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during the operation.

#### GeneratedPassword

Type: string
The new randomly generated password.

## GetAllGroupsResult

GetAllGroupsResult is the result of the GetAllGroups web method. It contains summary information of the systems Groups.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving Groups.

#### Groups

Type: Group array
An array of Groups.

## GetAllScriptsResult

GetAllScriptsResult is the result of the GetAllScripts web method. It contains summary information of the scripts created in Secret Server.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving Scripts.

#### UserScripts

Type: UserScript array
An array of the scripts available.

## GetAgentsResult - (Obsolete as of Secret Server 8.9)

GetAgentsResult is the result of the GetAgents web method. It contains summary information of the systems Agents.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving Agents.

#### Success

Type: boolean
Whether or not the operation was successful.

#### RemoteAgents

Type: RemoteAgent array
An array of RemoteAgents available.

## GetCheckOutStatusResult

This is the result of the GetCheckOutStatus web method call.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### Secret

Type:
If the Secret was loaded successfully, this property contains the Secret's data. The Secret's Password field will be masked.

#### CheckOutMinutesRemaining

Type: integer
The number of minutes until the CheckOut will expire.

#### IsCheckedOut

Type: boolean
Whether or not the Secret is checked out.

#### CheckOutUserDisplayName

Type: string
The display name of the user that has the Secret checked out.

#### CheckOutUserId

Type: integer
The Id of the user that has the Secret checked out.

## GetDependenciesResult

GetDependenciesResult is a type used when returning a collections of dependencies from a call to the GetDependencies method.

### Members

#### Dependencies

Type: array of Dependency
An array of dependencies.

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving Dependencies.

#### Success

Type: boolean
True if the operation succeeded, otherwise false.

## GetFavoritesResult

GetFavoritesResult is the result of the GetFavorites web method. It contains summary information of the user's favorite Secrets.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving favorites.

#### SecretSummaries

Type: array
An array of SecretSummary which are the favorites for the user that that the token belongs to.

## GetFoldersResult

GetFoldersResult is the result of getting folders from Secret Server.

### Members

#### Folders

Type: Folder array
An array of folder information.

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the folders.

#### Success

Type: boolean
Indicates whether or not the operation was successful.

## GetAllGroupsResult

GetAllGroupsResult is the result of getting groups from Secret Server.

### Members

#### Folders

Type: Group array
An array of folder information.

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the folders.

#### Success

Type: boolean
Indicates whether or not the operation was successful.

## GetNewSecretPolicyResult

GetNewSecretPolicyResult is the result of getting policies from Secret Server using the GetNewSecretPolicy method.

### Members

#### Errors

Type: string array

SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the folders.

**Active**

Type: boolean
Indicates whether or not the policy is active

**SecretPolicyId**

Type: int
Indicates the Id of the Secret Policy

**SecretPolicyItems**

Type: UserScript array
An array of the secret policy properties.

## GetScriptResult

GetScriptResult is the result of the GetScript web method. It contains summary information of the scripts created in Secret Server.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving the script.

**UserScripts**

Type: UserScript array
An array of the scripts properties.

## GetSecretAuditResult

GetSecretAuditResult is the result of the GetSecretAudit web method. It contains the Audit records for a Secret.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving favorites.

#### AuditSecrets

Type: AuditSecret array
An array of Secret audit records.

## GetSecretItemHistoryByFieldNameResult

This contains the history (audited values) for a Secret field.

### Members

#### Success

Type: boolean
Whether or not the retrieval of Secret field history returned any values.

#### SecretItemHistories

Type: array of SecretItemHistoryWebServiceResult
One or more values that make up the Secret field history.

## GetSecretResult

This is the result when loading a Secret. This contains all relevant information of a Secret and all of the Secret values.

When loading a Secret, the Secret property contains the fields as well as additional information about the Secret, such as the folder Id, the Secret template, and its Name.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### Secret

Type: Secret
If the Secret was loaded successfully, this property contains the Secret's data. This is null (nil) if one or more errors are present.

## GetSecretPolicyForSecretResult

This is the result when using the GetSecretPolicyForSecret method. This contains all relevant information of the Secret Policy on the specific secret.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### SecretId

Type: int
The Id of the returned secret

#### SecretPolicyId

Type: int
The Secret Policy Id set on the secret

#### Inherit

Type: boolean
Whether or not the secret is inheriting a policy

## GetSecretsByFieldValueResult

GetSecretsByFieldValueResult is the result if the GetSecretsByFieldValue and GetSecretsByExposedFieldValue web methods.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting Secret template fields.

#### Secrets

Type: Secret array
An array of Secrets, containing the Secrets that match the search criteria.

## GetSecretTemplateFieldsResult

GetSecretTemplateFieldsResult is the result of the SearchSecretsByFieldValue web method.

### Members

#### Errors

Type: string array

SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting Secret template fields.

**SecretFields**

Type: Field array
An array of Secret fields, containing information about the fields for the template.

## GetSecretTemplatesResult

GetSecretTemplatesResult is the result of the GetSecretTemplates web method.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting Secret templates.

**SecretTemplates**

Type:  array
An array of Secret templates, containing information about the template.

## GetSitesResult

GetSitesResult is the result of the GetDistributedEngines web method. It contains summary information of Sites that host Distributed Engines.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving Sites.

**Success**

Type: boolean
Whether or not the operation was successful.

**Sites**

Type: Site array
An array of Sites that host Distributed Engines.

## GetSSHLoginCredentialsResult

The result of the GetSSHLoginCredentials web method.

### Members

**Username**

Type: string
A temporary username to open a session to the Secret Server SSH Proxy.

**Password**

Type: string
A temporary password to open a session to the Secret Server SSH Proxy.

**Host**

Type: string
The Secret Server SSH Proxy host to connect to.

**Port**

Type: string
The port to make the connection on that the SSH Proxy is listening on.

## GetSSHLoginCredentialsWithMachineResult

The result of the GetSSHLoginCredentialsWithMachine web method.

### Members

**Username**

Type: string
A temporary username to open a session to the Secret Server SSH Proxy.

**Password**

Type: string
A temporary password to open a session to the Secret Server SSH Proxy.

**Host**

Type: string
The Secret Server SSH Proxy host to connect to.

**Port**

Type: string
The port to make the connection on that the SSH Proxy is listening on.

## GetSSHCommandMenuResult

Contains SSH Command Menu information.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving the list of Ticket Systems.

#### SshCommandMenu

Type: SshCommandMenu
SshCommandMenu contains the ID, name and other information regarding the menu itself.

#### SshCommands

Type: string
A text representation of the commands belonging to the SSH Command Menu. The format is the command name, then an equal sign (=), followed by the SSH command itself. Each command is separated by a newline (\r\n).

## GetSSHCommandMenusResult

Contains multiple SSH Command Menu results.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while retrieving the list of Ticket Systems.

#### SshCommandMenus

Type: Array of SshCommandMenu
SshCommandMenu contains the ID, name and other information regarding the menu itself.

## GetTicketSystemsResult

Contains the Ticket System information returned by a call to the GetTicketSystems method.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.

One or more errors that occurred while retrieving the list of Ticket Systems.

**Success**

Type: boolean
Whether or not the operation was successful.

**Sites**

Type: TicketSystem array
An array of active Ticket System in the system.

## GetUserResult

Contains the user information returned by a call to the GetUser method.

### Members

**User**

Type: User
User contains the ID, username, domain, two-factor settings, and more for the user being returned.

## GroupOrUserRecord

Used for setting Secret permissions. Contains a reference to either a group, or a user.

### Members

**Name**

Type: string
The display name of the group or user.

**DomainName**

Type: string
The display name of the domain for the group or user. This is null for local groups and users.

**IsUser**

Type: boolean
Whether or not the record is referring to a user.

**GroupId**

Type: integer
The group Id if the record is referring to a group. This value is null if referring to a user.

**UserId**

Type: integer
The user Id if the record is referring to a user. This value is null if referring to a group.

## Group

Used for displaying information returned from the GetAllGroups web service method.

### Members

**Id**

Type: integer
The group Id if the record is referring to a group. This value is null if referring to a user.

**Name**

Type: string
The display name of the group.

**DomainId**

Type: integer
The id of the domain for the group. This is null for local groups.

**DomainName**

Type: string
The display name of the groups domain. This is empty for local groups.

## ImpersonateResult

ImpersonateResult is the result of the ImpersonateUser web method.

### Members

**Errors**

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

**Token**

Type: string
An authentication token used for other web services methods. In errors occurred, this will be empty.

**AuthorizeURL**

Type: string
A URL through which to access the Device Access Request Manager.

## Permission

Permission contains a user or group, the name of the access role and the Id of the access role, as well as (now deprecated) boolean values indicating whether View, Edit and/or Owner permissions are enabled for that user or group. Either SecretAccessRoleName or SecretAccessRoleId must be provided.

### Members

**GroupOrUserRecord**

Type: GroupOrUserRecord
References a user or group in Secret Server.

**SecretAccessRoleName**

Type: string
The name of the access role in Secret Server. If SecretAccessRoleId is not provided, the lookup will be done on this property.

**SecretAccessRoleId**

Type: integer
The ID of the access role. This property takes precedence over SecretAccessRoleName during a lookup.

**View [Deprecated]**

Type: boolean
Indicates whether or not View permissions are enabled. Deprecated; use SecretAccessRoleName.

**Edit [Deprecated]**

Type: boolean
Indicates whether or not Edit permissions are enabled. Deprecated; use SecretAccessRoleName.

**Owner [Deprecated]**

Type: boolean
Indicates whether or not Owner permissions are enabled. Deprecated; use SecretAccessRoleName.

## RemoteAgent - (Obsolete as of Secret Server 8.9)

An agent that is responsible for Remote Password Changing and Heartbeat on remote networks.

### Members

**Id**

Type: integer
The Id of the remote Agent.

**DisplayName**

Type: string
The display name of the remote Agent.

#### Name

Type: string
The Name of the remote Agent.

## SearchFolderResult

SearchFoldersResult is the result of the SearchFolders web method. It contains summary information of the Folders that matched the search criteria.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while searching for Secrets.

#### Folders

Type: folder array
An array of Folders that match the search criteria.

## SearchSecretsByFolderLegacyResult

Returns Secrets matching the search term(s) and folder specified by SearchSecretsByFolderLegacy.

### Members

#### SecretSummaries

Type: array of SecretSummary
An array that contains the search results returned from the specified folder for the user that the token belongs to.

## SearchSecretsLegacyResult

Returns Secrets matching the search term(s) specified by SearchSecretsLegacy.

### Members

#### SecretSummaries

Type: array of SecretSummary
An array that contains the search results returned for the user that the token belongs to.

## SearchSecretsResult

SearchSecretsResult is the result of the SearchSecrets, SearchSecretsByFieldValue, SearchSecretsByExposedFieldValue, and SearchSecretsByExposedValues web methods. It contains summary information of the Secrets that matched the search criteria.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while searching for Secrets.

#### SecretSummaries

Type: array
An array of SecretSummary which contains the search results returned for the user that that the token belongs to.

## SearchSecretPoliciesResult

SearchSecretPoliciesResult is the result of the SearchSecretPolicies. It contains summary information of the Policies that matched the search criteria.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while searching for Secrets.

#### SecretSummaries

Type: array
An array of SecretPolicySummary which contains the search results returned for the user that that the token belongs to.

## SearchUsersResult

Returned by a call to the SearchUsers method.

### Members

#### Users

Type: array of User

Returns all users meeting the search term(s) specified.

## Secret

Contains the field values and metadata about a Secret, such as name, folder Id, and template Id. This type is used by many web methods involving Secrets to allow a common structure for passing data between the web methods.

### Members

**Name**

Type: string
The display name of the Secret.

**Items**

Type: array of SecretItem
An array of SecretItems that contain the name and value of the fields within the Secret, as well as information regarding what type of field it is.

**Id**

Type: integer
The Id of the Secret. This is used to uniquely identify the Secret and can be used in other web methods.

**SecretTypeId**

Type: integer
The Id of the Secret template that was used when the Secret was created.

**FolderId**

Type: integer
The Id of the folder that the Secret belongs to. If the Secret is not in a folder, negative one (-1) is returned.

**IsWebLauncher**

Type: boolean
This property is intended for internal use only.

**Active**

Type: boolean
Indicates whether the Secret is active or inactive.

**CheckOutMinutesRemaining**

Type: integer
Returns the number of minutes remaining before the Secret will be checked back in.

**IsCheckedOut**

Type: boolean
Indicates whether or not the Secret is currently Checked Out.

**CheckOutUserDisplayName**

Type: string
The display name of the user that has currently Checked Out the Secret.

**CheckOutUserId**

Type: integer
The Id of the user that has currently Checked Out the Secret.

**IsOutOfSync**

Type: boolean
Indicates whether or not Heartbeat has found the Secret information to be out of sync with the device it is configured to authenticate to.

**IsRestricted**

Type: boolean
If true, this value indicates that the Secret is either deleted or has Doublelock, Checkout, Require Comment, or Require Approval for Access enabled.

**OutOfSyncReason**

Type: string
If applicable, indicates the reason the Secret is out of sync – either Remote Password Change or Heartbeat failed.

**SecretSettings**

Type: SecretSettings
Contains boolean values defining security and Auto-Change configurations for the Secret.

**SecretPermissions**

Type: SecretPermissions
Contains boolean values defining permissions for the Secret.

## SecretField

SecretField is a type that contains information about an individual field on a Secret template.

### Members

**DisplayName**

Type: string
The display name of the field.

**Id**

Type: integer
The Id of the Secret field.

**IsFile**

Type: boolean
Indicates whether or not the field is a file attachment.

**IsNotes**

Type: boolean
Indicates whether or not the field is a multiline text box for entering notes.

**IsPassword**

Type: boolean
Indicates whether or not the field is a password type, which allows password masking and password generation.

**IsUrl**

Type: boolean
Indicates whether or not the field is a URL.

## SecretItem

SecretItem is the name and value of a particular field of a Secret.

### Members

**FieldDisplayName**

Type: string
The display name of the field.

**FieldId**

Type: nullable integer
The Id of the field that belongs to the Secret template.

**FieldName**

Type: string
The name of the field.

**Id**

Type: nullable integer
The unique Id of the field that belongs to the Secret itself.

**IsFile**

Type: boolean
Indicates whether or not the field is a file attachment. If the field is a file attachment, then the Value field should be ignored. To obtain the file's contents, use the DownloadFileAttachmentByItemId web method with the Id.

**IsNotes**

Type: boolean
Indicates whether or not the field is a multiline text box for entering notes.

**IsPassword**

Type: boolean
Indicates whether or not the field is a password type, which allows password masking and password generation.

**Value**

Type: string
The value of the field. This property should be ignored if the field is a File Attachment. See IsFile for more information.

## SecretItemHistoryWebServiceResult

A single audit from the history of a Secret field.

### Members

**SecretItemHistoryId**

Type: integer
A unique ID for the Secret field audit.

**UserId**

Type: integer
ID of the user that made the audited change to the Secret field.

**SecretItemId**

Type: integer
ID of the Secret field the audit was made for.

**SecretId**

Type: integer
ID of the Secret the field belongs to.

**Date**

Type: dateTime
The date and time the Secret field changed was audited.

**ItemValueNew**

Type: string

**ItemValueNew2**

Type: string

## SecretPermissions

SecretPermissions is a set of values pertaining to permissions for the Secret.

### Members

**CurrentUserHasView**

Type: boolean
Indicates whether or not the current user has View permissions for the Secret.

**CurrentUserHasEdit**

Type: boolean
Indicates whether or not the current user has Edit permissions for the Secret.

**CurrentUserHasOwner**

Type: boolean
Indicates whether or not the current user has Owner permissions for the Secret.

**InheritPermissionsEnabled**

Type: boolean
Indicates whether or not the Secret inherits permissions from its parent folder.

**IsChangeToPermissions**

Type: boolean
Indicates that these settings contain a change to be applied when adding or updating a Secret.

**Permissions**

Type: Array of Permission
An array of users and their permissions for the Secret.

## SecretSettings

SecretSettings is a set of values pertaining to auto-change and security settings for a Secret.

### Members

**AutoChangeEnabled**

Type: boolean
Indicates whether or not Auto-Change has been enabled for the Secret.

**RequiresApprovalForAccess**

Type: boolean
Indicates whether or not Require Approval for Access has been enabled for the Secret.

**RequiresComment**

Type: boolean
Indicates whether or not Require Comment has been enabled for the Secret.

**CheckOutEnabled**

Type: boolean
Indicates whether or not Check Out is required for the Secret.

**CheckOutChangePasswordEnabled**

Type: boolean
Indicates whether or not Check Out is configured to change the Secret password upon Check In.

**Approvers**

Type: GroupOrUserRecord array
The array of groups and/or users that are configured as approvers for the Secret to be accessed, if Require Approval has been configured.

**IsChangeToSettings**

Type: boolean
Indicates that these settings contain a change to be applied when adding or updating a Secret.

**PrivilegedSecretId**

Type: integer
Secret ID of a Secret with higher privileges that is to be used for the purposes of Remote Password Changing. To clear a privileged Secret, remove this tag from SecretSettings. For this setting to have any effect, IsChangeToSettings must be set to a value of 1.

**AssociatedSecretIds**

Type: integer array
Secret ID(s) of associated Secrets to be used for remote password changing, such as with Unix Root Account Secrets. To clear associated Secrets, include the AssociatedSecretIds tag containing no values. For this setting to have any effect, IsChangeToSettings must be set to a value of 1.

## SecretSummary

SecretSummary is used to display summary information about a Secret without revealing the values of the Secret itself.

### Members

**SecretId**

Type: integer
The ID of the Secret. This is used to uniquely identify the Secret and can be used in other web methods.

**SecretName**

Type: string
The display name of the Secret.

**SecretTypeName**

Type: string

The display name of the Secret template that was used when the Secret was created.

**SecretTypeId**

Type: integer
The ID of the Secret template that was used when the Secret was created.

**FolderId**

Type: integer
The ID of the folder that the Secret belongs to. If the Secret is not in a folder, negative one (-1) is returned.

**IsRestricted**

Type: boolean
If true, this value indicates that the Secret is either deleted or has Doublelock, Checkout, Require Comment, or Require Approval for Access enabled.

## SecretTemplate

SecretTemplate is a type that contains information about an individual Secret template.

### Members

**Id**

Type: integer
The Id of the Secret template.

**Name**

Type: string
The name of the Secret template.

**Fields**

Type: array
An array of fields that belong to the Secret template. They are given in order that they are ordered in Secret Server.

## Settings (Folder)

Settings (Folder) contains boolean values indicating whether settings have been changed and whether the folder inherits secret policy as well the id of any applied Secret Policy.

### Members

**IsChangeToSettings**

Type: boolean
Indicates whether a change is being made to the Folder permissions.

**InheritSecretPolicy**

Type: boolean
Indicates whether or not the Folder inherits Secret Policy from its parent.

**SecretPolicyId**

Type: integer
The id of the Secret Policy applied to the folder.

## SSHCommandMenu

SSHCommandMenu is a type that represents a SSH Command Menu object.

### Members

**SSHCommandMenuId**

Type: integer
The ID of the Menu ID.

**Name**

Type: string
The name of the SSH command menu.

**Description**

Type: string
The detailed description of the SSH command menu.

**Active**

Type: boolean
Whether or not the SSH command menu is active.

## TicketSystem

Object that contains information about a configured Ticket System in the system.

### Members

**TicketSystemId**

Type: integer
The Id of the Ticket System

**Name**

Type: string
The Name of the Ticket System

#### Description

Type: string
The Description of the Ticket System

#### IsDefault

Type: boolean
This will be true if the Ticket System is set as the default for the system, otherwise it will be false.

## TokenIsValidResult

Used to determine if the token is still valid.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during authentication.

#### MaxOfflineSeconds

Type: int (32-bit signed integer)
This value is used for internal purposes only.

## UpdateScriptResult

UpdateScriptResult is the result of the UpdateScript web method. If the web method was successful, it also contains information about the Script that was update.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while adding a Secret.

#### Script

Type: Script
Information about the Script that was updated.

## UpdateUserResult

Resulting updated user information after calling UpdateUser.

### Members

**User**

Type: User
The updated user information.

## User

User is a type that represents a Secret Server user.

### Members

**Id**

Type: integer
The ID of the user. This can remain null when creating a new user.

**UserName**

Type: string
The login name of the user.

**DisplayName**

Type: string
The display name of the user.

**DomainId**

Type: integer
The ID of the domain the user belongs to. Leave this null if the user is not a on a domain (a local user).

**IsApplicationAccount**

Type: boolean
Whether or not the user is an application account. Application accounts are used for integration with the Secret Server integrated Java or .NET API.

**RadiusTwoFactor**

Type: boolean
Whether or not the user is using RADIUS two-factor to authenticate to Secret Server. Note: either RadiusTwoFactor or EmailTwoFactor must be set to false.

**EmailTwoFactor**

Type: boolean
Whether or not the user is using email two-factor to authenticate to Secret Server. Note: either RadiusTwoFactor or EmailTwoFactor must be set to false.

**RadiusUserName**

Type: string
The user name that is associated with this user on the RADIUS server.

**EmailAddress**

Type: string
The email address that is associated with the user.

**Password**

Type: string
The user's password.

**Enabled**

Type: boolean
Whether or not the user is enabled in Secret Server. Disabled users cannot log into the application and do not consume a user license.

**DuoTwoFactor**

Type: boolean
Whether or not the user is using Duo two-factor to authenticate to Secret Server.

**OATHTwoFactor**

Type: boolean
Whether or not the user is using OATH (TOTP) two-factor to authenticate to Secret Server.

## UserInfoResult

User is a type that provides identifying info for a token using the WhoAmI web service method.

### Members

**UserId**

Type: integer
The Id of the user.

**DisplayName**

Type: string
The display name of the user.

**DomainId**

Type: integer
The Id of the domain that the user belongs to. This is null if the user is a local Secret Server account.

**DomainName**

Type: string
The name of the user's domain.

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred while getting the user info from the token.

#### KnownAs

Type: string
The domain name concatenated with the username, i.e. company.domain.com\secretadmin.

## VersionGetResult

This is returned from the VersionGet web method.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during the operation.

#### Version

Type: string
The current version of Secret Server.

## WebServiceResult

WebServiceResult is a generic result of a web method that does not return any information except for Errors.

### Members

#### Errors

Type: string array
SOAP Type: ArrayOfString. A complex type of a sequence of strings with an unbounded number of occurrences.
One or more errors that occurred during the operation.