

MS-ASPROV Test Suite Specification

**Abstract:** This document provides information about how to configure the test suite and how the MS-ASPROV test suite is designed to test the MS-ASPROV Open Specification usability and accuracy. It describes test assumptions, scope and constraints of the test suite. It also specifies test scenarios, detailed test cases, test suite architecture and adapter design.

Contents

[1 Configuring the test suite 3](#_Toc387412115)

[1.1 Configuring the test suite client 3](#_Toc387412116)

[1.1.1 Configuring the test suite client manually 3](#_Toc387412117)

[1.1.2 Configuring the test suite client by scripts 3](#_Toc387412118)

[1.2 Configuring the system under test (SUT) 3](#_Toc387412119)

[1.2.1 Configuring the SUT manually 3](#_Toc387412120)

[1.2.2 Configuring the SUT by scripts 3](#_Toc387412121)

[1.3 Configuring the SHOULD/MAY requirements 3](#_Toc387412122)

[2 Test suite design 5](#_Toc387412123)

[2.1 Assumptions, scope and constraints 5](#_Toc387412124)

[2.2 Test suite architecture 5](#_Toc387412125)

[2.3 Technical dependencies and considerations 7](#_Toc387412126)

[2.4 Adapter design 7](#_Toc387412127)

[2.4.1 Adapter overview 7](#_Toc387412128)

[2.4.2 Technical feasibility of adapter approach 7](#_Toc387412129)

[2.4.3 Adapter abstract layer 8](#_Toc387412130)

[2.4.4 Adapter details 9](#_Toc387412131)

[2.5 Test scenarios 10](#_Toc387412132)

[2.5.1 S01\_AcknowledgePolicySettings 11](#_Toc387412133)

[2.5.2 S02\_RemoteWipe 11](#_Toc387412134)

[2.5.3 S03\_ProvisionNegative 11](#_Toc387412135)

[2.6 Test case design 11](#_Toc387412136)

[2.6.1 Traditional test case design 11](#_Toc387412137)

[2.6.2 Test case description 12](#_Toc387412138)

# Configuring the test suite

## Configuring the test suite client

### Configuring the test suite client manually

Before you run the test suite, update the values in the MS-ASPROV\_TestSuite.deployment.ptfconfig file. The MS-ASPROV\_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-ASPROV\TestSuite\MS-ASPROV\_TestSuite.deployment.ptfconfig.
2. Update the following value to specify the common configuration file.

Property name="CommonConfigurationFileName" value="ExchangeCommonConfiguration.deployment.ptfconfig"

**Note** This property can be removed or set to empty if the required properties are copied to the test suite specific configuration file. Any other changes to this property will cause all test cases in the test suite to fail during execution. The test suite searches through its specific configuration file and uses those properties, if they are defined, before looking for them in the common configuration file (if specified).

1. Update the following properties' values to match SUT settings and configuration.

* Property name="User1Name" value="MSASPROV\_User01"
* Property name="User1Password" value="Password01!"
* Property name="User2Name" value="MSASPROV\_User02"
* Property name="User2Password" value="Password01!"
* Property name="User3Name" value="MSASPROV\_User03"
* Property name="User3Password" value="Password01!"

### Configuring the test suite client by scripts

To configure the test suite client using scripts, see section 5.2.1 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.docx).

## Configuring the system under test (SUT)

### Configuring the SUT manually

To manually configure the SUT, see section 5.1.2 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.docx).

### Configuring the SUT by scripts

To configure the SUT using scripts, see section 5.1.1 of the [ExchangeEASTestSuiteDeploymentGuide.docx](../ExchangeEASTestSuiteDeploymentGuide.docx).

## Configuring the SHOULD/MAY requirements

Implementation of the SHOULD/MAY and endnote-related requirements are pre-configured in the format "<Property name="RXXXEnabled" value="XXXX"/>" for the product versions in the following config files:

* MS-ASPROV\_ExchangeServer2007\_SHOULDMAY.deployment.ptfconfig
* MS-ASPROV\_ExchangeServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-ASPROV\_ExchangeServer2013\_SHOULDMAY.deployment.ptfconfig

If RXXXEnabled is set to true, the requirement must be checked. If false, the requirement must not be checked. For Microsoft product versions, all values should not be changed. For third-party products, the closest Microsoft product version should be chosen, and the value of RXXXEnabled should be updated according to the real product behavior. For example, if Exchange Server 2010 is chosen,user can open **MS-ASPROV\_ExchangeServer2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled accordingly.

# Test suite design

## Assumptions, scope and constraints

Assumptions

This test suite assumes that authentication has been performed by the underlying protocols.

Scope

In scope

* This test suite will verify the accuracy and integrity of the technical content in the Open Specification against the results returned from the protocol server by using one command: Provision.
* This test suite will verify the server-side and testable requirements by running all test cases on both HTTP and HTTPS.
* This test suite will verify requirements from Exchange ActiveSync: WAP Binary XML (WBXML) Algorithm (MS-ASWBXML) and Exchange ActiveSync: Data Types (MS-ASDTYPE).

Out of scope

* This test suite will not verify the requirements related to client behaviors.
* This test suite will not verify the requirements related to server internal behaviors.
* This test suite will not verify the internal implementations of its transport protocol stack.

Constraints

None.

## Test suite architecture

This test suite verifies the server-side and testable requirements obtained from the Open Specification. The following figure shows the architecture of this test suite.



The architecture of the test suite

The details of the MS-ASPROV test suite architecture

* SUT hosts the security policy service which this test suite runs against.
* From third-party user’s point of view, the SUT is the protocol server implementation.
* The following products have been tested with the MS-ASPROV test suite on the Windows platform.
* Microsoft Exchange Server 2007 Service Pack 3 (SP3)
* Microsoft Exchange Server 2010 Service Pack 3 (SP3)
* Microsoft Exchange Server 2013 Service Pack 1 (SP1)
* The test suite acts as the client to communicate with the SUT and validates the requirements gathered from MS-ASPROV Open Specification.
* Test cases use the MS-ASPROV protocol adapter to call and get the results of the MS-ASPROV commands. Test cases also use the SUT control adapter to configure and retrieve information from SUT.
* MS-ASPROV protocol adapter is used in the test cases. The test cases call the methods in the interfaces to invoke the MS-ASPROV protocol adapter’s commands.
* MS-ASPROV protocol adapter uses ActiveSyncClient to send command request and retrieve command response.
* ActiveSyncClient encodes and decodes ActiveSync commands defined in [MS-ASCMD] by using MS-ASWBXML and communicates with the SUT via MS-ASHTTP.
* SUT control adapter is used in the test cases. The test cases call the method in the interface to configure the SUT.

## Technical dependencies and considerations

Technical dependencies

* This test suite depends on HTTP protocol or HTTPS protocol to transmit the messages.
* This test suite depends on Protocol Test Framework (PTF) to derive managed adapters.
* This test suite depends on MS-ASWBXML to encode request bodies into WBXML for transmission to an ActiveSync server.
* This test suite depends on the xsd.exe tool in the .NET Framework SDK to generate structures used in the MS-ASPROV request and response.
* This test suite depends on MS-ASHTTP to synchronize data with the data that is stored on the server.

Encryption consideration

Transportation of MS-ASPROV includes HTTP and HTTPS, and encryption will be handled by HTTPS.

## Adapter design

### Adapter overview

One protocol adapter and one SUT control adapter are used in this test suite.

Protocol adapter

* MS-ASPROV protocol adapter
* The MS-ASPROV adapter is a managed adapter, which is derived from the ManagedAdapterBase class in PTF.
* The MS-ASPROV adapter has the following functionalities
* Construct requests of two MS-ASPROV commands;
* Communicate with the SUT by sending command requests and receiving the corresponding responses from the SUT;
* Parse the response messages and validate the messages according to the XSD schema;
* Choose HTTP or HTTPS for transport.

SUT control adapters

* SUT control adapter
* The SUT control adapter will be a PowerShell scripted adapter.
* The SUT control adapter has the following functionalities
* Wipe all data of the user's device;
* Remove the device from the user's mobile list.
* The SUT control adapter is invoked by the test cases.

### Technical feasibility of adapter approach

Message generation

The MS-ASPROV adapter gets the parameter values and calls the corresponding commands in ActiveSyncClient. The ActiveSyncClient serializes the parameter values to XML elements, and then encodes the request bodies into WBXML for transmission to an ActiveSync server.

Message consumption

The messages received from the SUT will be parsed in the ActiveSyncClient and be passed upon to the MS-ASPROV adapter. Then these messages are consumed in the MS-ASPROV adapter to validate the message format and to validate the logic-related requirements in the test cases.

SUT control adapter

The SUT control adapter is designed to wipe data, remove device.

### Adapter abstract layer

Protocol adapter

MS-ASPROV adapter interface

There are eight methods declared in the MS-ASPROV adapter interface IMS\_ASPROVAdapter.

* Two methods correspond to the MS-ASPROV command: Provision and SendProvisionStringRequest. Four methods (SwitchUser, ApplyPolicyKey, ApplyDeviceType and SyncEmail) are customized methods. Other two methods correspond to the MS-ASCMD command: FolderSync, Sync.

The methods are described in the following table.

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| 1 | Provision | This method is used to request the security policy settings that the administrator sets from the server. |
| 2 | SendProvisionStringRequest | This method is used to send string request of Provision command to the server and get the response. |
| 3 | SwitchUser | This method is used to change the user authentication. |
| 4 | ApplyPolicyKey | This method is used to apply the specified PolicyKey. |
| 5 | ApplyDeviceType | This method is used to apply the specified DeviceType. |
| 6 | FolderSync | This method is used to synchronize the collection hierarchy from server. |
| 7 | Sync | This method is used to synchronize the changes in a collection between the client and the server by sending SyncRequest object. |
| 8 | SyncEmail | This method is used to find an email with specific subject. |

MS-ASPROV adapter interface methods

SUT control adapters

SUT control adapter interface

There are two methods declared in the SUT control adapter interface IMS\_ASPROVSUTControlAdapter.

The methods are described in the following table.

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| 1 | WipeData | This method is used to wipe all data of the user's device. |
| 2 | RemoveDevice | This method is used to remove the device from the user's mobile list. |

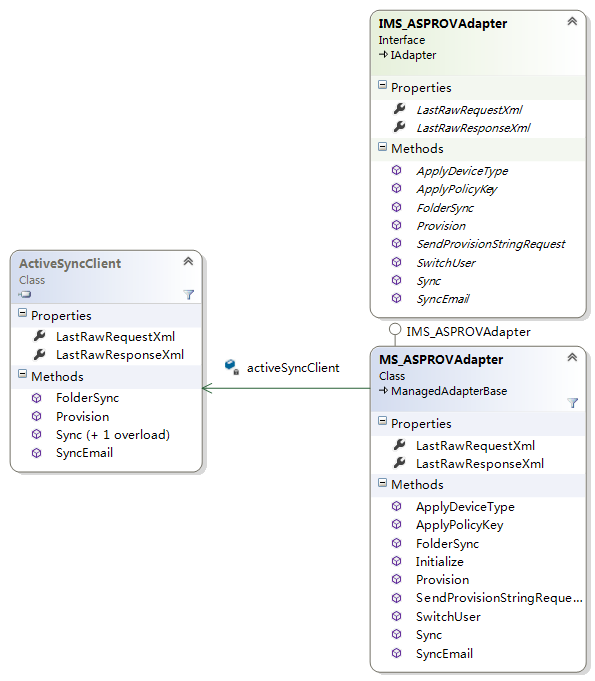
SUT control adapter interface methods

### Adapter details

#### Protocol adapter

##### MS-ASPROV protocol adapter

The following figure shows the class diagram of the MS-ASPROV protocol adapter and the relationship between MS\_ASPROVAdapter and ActiveSyncClient.



MS-ASPROV adapter and ActiveSyncClient class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_ASPROVAdapter is the interface of the protocol adapter.
* IMS\_ASPROVAdapter defines the methods invoked by test cases, including Provision, SwitchUser, SendProvisionStringRequest, ApplyPolicyKey, ApplyDeviceType, FolderSync, SyncEmail and Sync methods.
* The two properties LastRawRequestXml and LastRawResponseXml are used to get the raw request and raw response.

Adapter implementation

* MS\_ASPROVAdapter is the protocol adapter class of the test suite. It is used to implement IMS\_ASPROVAdapter.
* The initialize method is used to initialize an instance of ActiveSyncClient.
* The two properties LastRawRequestXml and LastRawResponseXml are used to get the raw request and raw response.

Other class

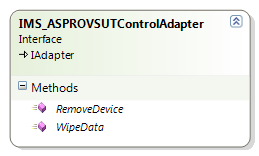
* ActiveSyncClient is used by MS\_ASPROVAdapter to send (Sync, FolderSync and Provision) command requests and retrieve command responses.

#### SUT control adapter

The following outlines details of class diagram:

##### SUT control adapter

The following figure shows the class diagram of the SUT control adapter.



SUT control adapter class diagram

The following outlines details of the class diagram:

* The IMS\_ASPROVSUTControlAdapter is the interface of the SUT control adapter which is implemented by Microsoft PowerShell script. The implementation can be substituted by other implementation for the third party’s need.

## Test scenarios

Three scenarios are designed to cover server-side, testable requirements in the MS-ASPROV test suite. The details of the scenarios are as follows.

|  |  |
| --- | --- |
| Scenario | Description |
| S01\_AcknowledgePolicySettings | Test the acknowledgement phase of Provision. |
| S02\_RemoteWipe | Test the remote wipe directive. |
| S03\_ProvisionNegative | Test negative status of Provision command. |

MS-ASPROV scenarios

### S01\_AcknowledgePolicySettings

Description

Test the acknowledgement phase of Provision command.

Commands

* Provision
* FolderSync

Prerequisites

* Set the URI (Uniform Resource Identifier), HTTP version and query value type.

### S02\_RemoteWipe

Description

Test the remote wipe directive.

Commands

* Provision

Prerequisites

* Set the URI, HTTP version and query value type.

### S03\_ProvisionNegative

Description

Test negative status of Provision command.

Commands

* Provision
* FolderSync

Prerequisites

* Set the URI, HTTP version and query value type.

## Test case design

### Traditional test case design

Traditional testing is adopted as the test approach for this test suite. The test cases are designed to cover the server-side and testable requirements.

There are 7 traditional test cases are designed to cover the three scenarios mentioned in section [2.6 Test scenarios](#_Test_scenarios). Details of the traditional test cases are specified in [section 2.7.2 Test case description](#_Test_Cases_Description). The scenarios distributions of the test cases are listed in the following table.

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_AcknowledgePolicySettings | MSASPROV\_S01\_TC01\_AcknowledgeSecurityPolicySettings |
| MSASPROV\_S01\_TC02\_WithoutAcknowledgingSecurityPolicySettings |
| S02\_RemoteWipe | MSASPROV\_S02\_TC01\_RemoteWipe |
| S03\_ProvisionNegative | MSASPROV\_S03\_TC01\_VerifyPolicyStatus3 |
| MSASPROV\_S03\_TC02\_VerifyPolicyStatus5 |
| MSASPROV\_S03\_TC03\_VerifyProvisionStatus2 |
| MSASPROV\_S03\_TC04\_VerifyInvalidPolicyKey |

Test case scenarios distribution

### Test case description

The steps in the following test cases use methods and parameters in the adapter interfaces directly.

The following tables describe the traditional test cases.

|  |  |
| --- | --- |
| S01\_AcknowledgePolicySettings | |
| Test case ID | MSASPROV\_S01\_TC01\_AcknowledgeSecurityPolicySettings |
| Description | This test case is intended to validate the acknowledgement phase of Provision. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls SwitchUser to switch the current user to User2 who has been configured with customized policy.  2. The client calls Provision command to download the policy settings.  3. The client calls Provision command to acknowledge the policy settings.  4. The client calls ApplyPolicyKey to apply the PolicyKey returned from last step.  5. The client calls FolderSync to synchronize the collection hierarchy.  6. The client calls SwitchUser to switch the current user to User1. |
| Cleanup | N/A |

MSASPROV\_S01\_TC01\_AcknowledgeSecurityPolicySettings

|  |  |
| --- | --- |
| S01\_AcknowledgePolicySettings | |
| Test case ID | MSASPROV\_S01\_TC02\_WithoutAcknowledgingSecurityPolicySettings |
| Description | This test case is intended to validate command could be executed successfully without acknowledging security policy settings if a security policy is set on the implementation to allow it. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls SwitchUser to switch the current user to User3 who has a policy with AllowNonProvisionableDevices parameter set to True.  2. The client calls ApplyPolicyKey to apply string.Empty to PolicyKey.  3. The client calls FolderSync without acknowledging security policy settings.  4. The client calls SwitchUser to switch the current user to User1. |
| Cleanup | N/A |

MSASPROV\_S01\_TC02\_WithoutAcknowledgingSecurityPolicySettings

|  |  |
| --- | --- |
| S02\_RemoteWipe | |
| Test case ID | MSASPROV\_S02\_TC01\_RemoteWipe |
| Description | This test case is intended to validate a successful remote wipe directive of Provision. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls SwitchUser to switch the current user to User3.  2. The client calls FolderSync to synchronize the collection hierarchy.  3. The client calls SwitchUser to switch User3 to User1.  4. The client calls FolderSync to synchronize the collection hierarchy.  5. The client calls ApplyDeviceType to apply the unique DeviceType of the client.  6. The client calls Provision command to download the policy settings.  7. The client calls Provision command to acknowledge the policy settings.  8. The client calls WipeData to request remote wipe for the device.  9. The client calls Provision command with an empty request to get remote wipe directive.  10. The client calls Provision command with Status “2” in RemoteWipe element of the request.  11. The client calls Provision command with an empty request to get remote wipe directive.  12. The client calls Provision command with Status “1” in RemoteWipe element of the request.  13. The client calls RemoveDevice to remove the device with the unique DeviceType from the device list.  14. The client calls Provision command with an empty request to get remote wipe directive. |
| Cleanup | The client calls Sync command to delete the items created in test cases. |

MSASPROV\_S02\_TC01\_RemoteWipe

|  |  |
| --- | --- |
| S03\_ProvisionNegative | |
| Test case ID | MSASPROV\_S03\_TC01\_VerifyPolicyStatus3 |
| Description | This test case is intended to validate Status 3 of Policy element. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Provision command to download policy settings with an invalid policy type. |
| Cleanup | N/A |

MSASPROV\_S03\_TC01\_VerifyPolicyStatus3

|  |  |
| --- | --- |
| S03\_ProvisionNegative | |
| Test case ID | MSASPROV\_S03\_TC02\_VerifyPolicyStatus5 |
| Description | This test case is intended to validate Status 5 of Policy element. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Provision command with a mismatched policy key. |
| Cleanup | N/A |

MSASPROV\_S03\_TC02\_VerifyPolicyStatus5

|  |  |
| --- | --- |
| S03\_ProvisionNegative | |
| Test case ID | MSASPROV\_S03\_TC03\_VerifyProvisionStatus2 |
| Description | This test case is intended to validate Status 2 of Provision element. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls SendProvisionStringRequest to download policy settings with the request which has invalid nodes. |
| Cleanup | N/A |

MSASPROV\_S03\_TC03\_VerifyProvisionStatus2

|  |  |
| --- | --- |
| S03\_ProvisionNegative | |
| Test case ID | MSASPROV\_S03\_TC04\_VerifyInvalidPolicyKey |
| Description | This test case is intended to validate the status code when the policy key is invalid. |
| Prerequisites | Common prerequisites |
| Test execution steps | 1. The client calls Provision command to download the policy settings.  2. The client calls Provision command to acknowledge the policy settings.  3. The client calls ApplyPolicyKey to apply the PolicyKey which is different with the one returned from last step.  4. The client calls FolderSync command. |
| Cleanup | N/A |

MSASPROV\_S03\_TC04\_VerifyInvalidPolicyKey