[MS-DVRD]:

Device Registration Discovery Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
8/8/2013	1.0	New	Released new document.
11/14/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
2/13/2014	1.0	None	No changes to the meaning, language, or formatting of the technical content.
5/15/2014	2.0	Major	Significantly changed the technical content.
6/30/2015	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The discovery of information needed to register devices is accomplished through the protocol defined in this specification: the Device Registration Discovery Protocol. Registration of a device in the **device registration service** by using the information provided by the Device Registration Discovery Protocol is handled by the Device Registration Enrollment Protocol [MS-DVRE].

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

device registration service: A service that allows registration of computing devices on a corporate network. These devices might not be controlled by the administrator of the network.

Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS): An extension of HTTP that securely encrypts and decrypts webpage requests.

relying party (RP): A web application or service that consumes security tokens issued by a security token service (STS).

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC4234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005, http://www.rfc-editor.org/rfc/rfc4234.txt

[RFC4346] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006, http://www.ietf.org/rfc/rfc4346.txt

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, http://www.rfc-editor.org/rfc/rfc6749.txt

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[XMLSCHEMA1] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

1.2.2 Informative References

[MS-DVRE] Microsoft Corporation, "Device Registration Enrollment Protocol".

1.3 Overview

This document defines a protocol for returning information about a server that implements the Device Registration Enrollment Protocol [MS-DVRE] as structured RESTful resources.

The Device Registration Discovery Protocol is a single REST-based endpoint that returns XML or JavaScript Object Notation (JSON) formatted data in the response message. This information can be used to connect and register a device with a server that implements the Device Registration Enrollment Protocol.

This document defines and uses the following terms:

Server: Refers to the server that implements the REST web service that accepts and responds to device registration discovery requests using the Device Registration Discovery Protocol.

Client: Refers to the client that creates and sends a discovery request to the server using the Device Registration Discovery Protocol.

Device registration service (DRS) server: Refers to the server that implements the Device Registration Enrollment Protocol [MS-DVRE] for device registration.

OAuth2 server: Refers to the server that implements the OAuth2 protocol [RFC6749] and provides authentication services for the device registration service (DRS) server.

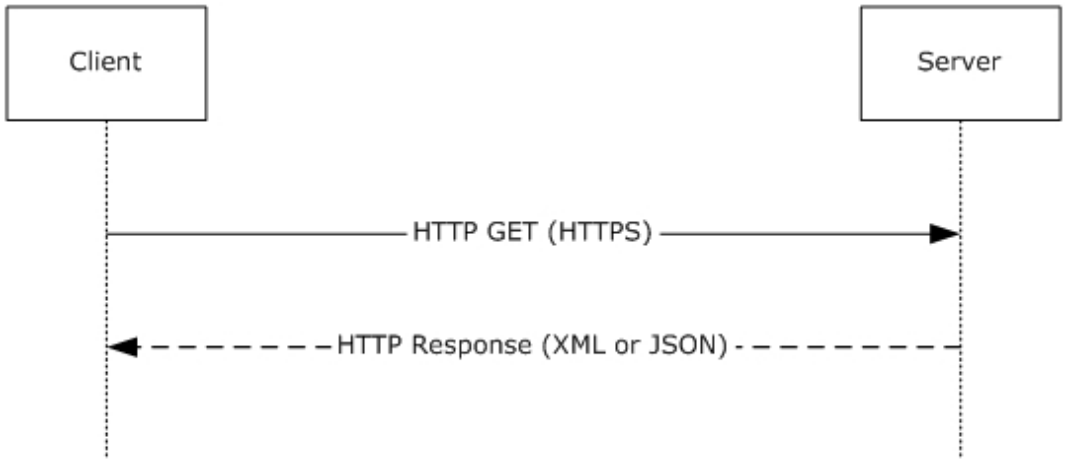

Figure 1: Device discovery sequence

1.4 Relationship to Other Protocols

The following figure illustrates the relationship of this protocol to other protocols.

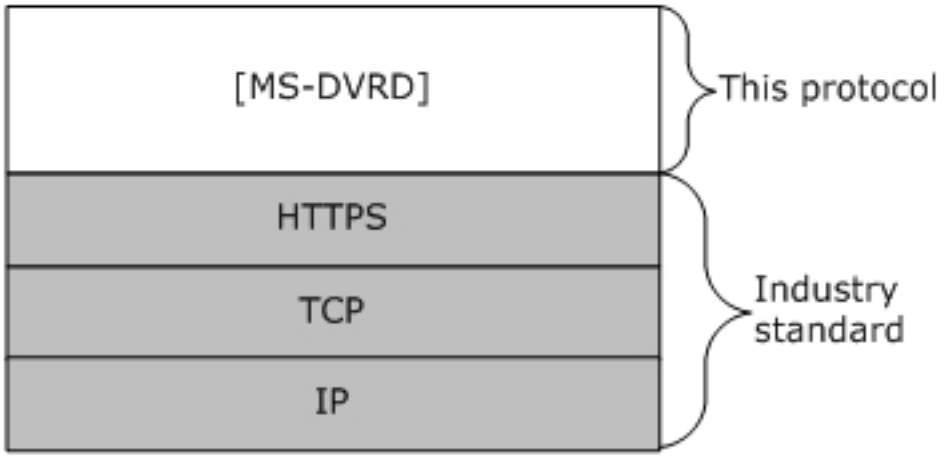

Figure 2: Protocols related to the Device Registration Discovery Protocol

1.5 Prerequisites/Preconditions

The protocol defined in this document does not provide a mechanism for a client to discover the existence and location of arbitrary data services (of the server). It is a prerequisite that the client obtain a URI to the server before the protocol can be used.

Neither the protocol defined in this document nor its base protocols define an authentication or authorization scheme.

1.6 Applicability Statement

This protocol defines a means for exposing information about a DRS server as structured RESTful resources. This protocol is applicable to both Internet and intranet client-server scenarios.

1.7 Versioning and Capability Negotiation

The protocol provides a URI parameter for specifying the desired version. See section 2.2.3.1.

1.8 Vendor-Extensible Fields

This protocol does not provide any mechanism for capability negotiation beyond that specified in section 1.7.

1.9 Standards Assignments

This protocol has not been assigned any standard parameters.

2 Messages

2.1 Transport

The Device Registration Discovery Protocol consists of a single RESTful web service.

- HTTPS over TCP/IP [RFC2616]

The protocol MUST operate on the following URI endpoint.

Web service	Location
Discovery Web Service	https://<server>:<server port>/EnrollmentServer/contract

All client messages to the server MUST use **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)** and provide server authentication using TLS 1.1 [RFC4346].

2.2 Common Data Types

2.2.1 Namespaces

This specification defines and references various XML namespaces by using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	NameSpaces URI	Reference
tns	http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities	This specification
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1]
tns1	http://tempuri.org	This specification

2.2.2 HTTP Headers

The following sections define the syntax of the HTTP headers by using the Augmented Backus-Naur Form (ABNF) syntax. ABNF syntax is defined in Augmented BNF for Syntax Specifications: ABNF, as specified in [RFC4234].

2.2.2.1 Accept

The format of the Accept header is as follows.

Accept = "application/json" / "application/xml"

2.2.3 Common URI Parameters

The following table summarizes the set of Common URI Parameters defined by this specification.

URI parameter	Description
api-version	An integer that indicates the data version that is expected by the client.

2.2.3.1 api-version

The *api-version* parameter is an integer that indicates the data version that is expected by the client. This parameter MUST be included in all client requests.

```
String = *(%x20-7E)
api-version = String
```

2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions defined by this specification.

Complex Type	Description
AuthenticationServiceData	Information about the authentication services and schemes that are supported by the DRS server. See section 2.2.4.1.
DeviceRegistrationServiceData	Information about the DRS server. See section 2.2.4.2.
Discovery	The root element. See section 2.2.4.3.
IdentityProviderServiceData	Information about the identity provider server. See section 2.2.4.5.
OAuth2ServiceData	Information about the OAuth2 server. See section 2.2.4.4.

2.2.4.1 AuthenticationServiceData

The AuthenticationServiceData type contains meta-data about all of the authentication schemes that are supported and allowed by the DRS server.

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities

```
<xs:complexType name="AuthenticationServiceData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="OAuth2" nillable="true"
type="tns:OAuth2ServiceData"/>
  </xs:sequence>
</xs:complexType>
```

OAuth2: The top-level object for OAuth2ServiceData. See section 2.2.4.4.

2.2.4.2 DeviceRegistrationServiceData

The DeviceRegistrationServiceData type contains meta-data about the DRS server. This information, along with the information from AuthenticationServiceData (section 2.2.4.1), can be used to connect and authenticate to the DRS server.

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities

```
<xs:complexType name="DeviceRegistrationServiceData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="RegistrationEndpoint" type="xs:anyURI"/>
    <xs:element minOccurs="0" maxOccurs="1" name="RegistrationResourceId" type="xs:anyURI"/>
    <xs:element minOccurs="0" maxOccurs="1" name="ServiceVersion" nillable="true"
type="xs:string"/>
  </xs:sequence>
```

```
</xs:complexType>
```

RegistrationEndpoint: The URL of the SOAP web service hosted on the DRS server.

RegistrationResourceId: The **relying party** identity of the DRS server as defined by the identity provider or federation provider.

ServiceVersion: An integer that indicates the discovery data version. This MUST match the version that was requested by the client. See section 2.2.3.1.

2.2.4.3 Discovery

The root element.

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities

```
<xs:complexType name="Discovery">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="DeviceRegistrationService" nillable="true"
type="tns:DeviceRegistrationServiceData"/>
    <xs:element minOccurs="0" maxOccurs="1" name="AuthenticationService" nillable="true"
type="tns:AuthenticationServiceData"/>
    <xs:element minOccurs="0" maxOccurs="1" name="IdentityProviderService" nillable="true"
type="tns:IdentityProviderServiceData"/>
  </xs:sequence>
</xs:complexType>
```

AuthenticationService: The top-level object for AuthenticationServiceData. See section 2.2.4.1.

DeviceRegistrationService: The top-level object for DeviceRegistrationServiceData. See section 2.2.4.2.

IdentityProviderService: The top-level object for IdentityProviderServiceData. See section 2.2.4.5.

2.2.4.4 OAuth2ServiceData

The OAuth2ServiceData type contains the information needed to connect to the OAuth2 server [RFC6749].

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities

```
<xs:complexType name="OAuth2ServiceData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="AuthCodeEndpoint" type="xs:anyURI"/>
    <xs:element minOccurs="0" maxOccurs="1" name="TokenEndpoint" type="xs:anyURI"/>
  </xs:sequence>
</xs:complexType>
```

AuthCodeEndpoint: The URL of the authorization endpoint on the OAuth2 server. This endpoint is used to request an authorization code.

TokenEndpoint: The URL of the token endpoint on the OAuth2 server. This endpoint is used to request access tokens in exchange for an authorization code.

2.2.4.5 IdentityProviderServiceData

The IdentityProviderServiceData type contains meta-data about the identity server that is used by the DRS server.

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities

```
<xs:complexType name="IdentityProviderServiceData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="PassiveAuthEndpoint" type="xs:anyURI"/>
  </xs:sequence>
</xs:complexType>
```

PassiveAuthEndpoint: The URL of the passive authentication endpoint of the identity provider.

3 Protocol Details

3.1 IHttpDiscoveryService Server Details

3.1.1 Abstract Data Model

The following information MUST be maintained on the server.

RegistrationEndpoint: See section 2.2.4.2 for DeviceRegistrationServiceData.

RegistrationResourceId: See section 2.2.4.2 for DeviceRegistrationServiceData.

ServiceVersion: See section 2.2.4.2 for DeviceRegistrationServiceData.

AuthCodeEndpoint: See section 2.2.4.4 for OAuth2ServiceData.

TokenEndpoint: See section 2.2.4.4 for OAuth2ServiceData.

PassiveAuthEndpoint: See section 2.2.4.5 for IdentityProviderServiceData.

3.1.2 Timers

None.

3.1.3 Initialization

The server that implements the Device Registration Discovery Protocol must be initialized. Any databases or tables that contain the information needed in the Device Registration Discovery Protocol response MUST be initialized.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

Resource	Description
contract?api-version={api-version}	An object that represents the endpoints and authentication policies for a DRS server.

3.1.5.1 contract?api-version={api-version}

api-version: An integer that indicates the data version expected by the client. This parameter MUST be included in all client requests. See section 2.2.3.1.

The following HTTP method is allowed to be performed on this resource.

HTTP method	Description
GET	Get connection and authentication meta-data for the DRS server.

3.1.5.1.1 GET

This operation is transported by an HTTP **GET**.

The operation can be invoked through the following URI:

```
contract?api-version={version}
```

3.1.5.1.1.1 Request Body

The request body SHOULD be empty. Any content MUST be ignored by the server.

3.1.5.1.1.2 Response Body

The response body is encoded in either XML or JSON format. The format is controlled by the Accept header defined in section 2.2.2.1.

```
<xs:element name="DiscoverResponse" nillable="true"
xmlns:q1="http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities"
type="q1:Discovery"/>
```

3.1.5.1.1.3 Processing Details

1. The server MUST respond only to requests that have established TLS 1.1 server authentication [RFC4346].
2. The server MUST respond only to requests that have an *api-version* URI parameter that contains the string "1.0".
3. If the Accept header is present in the request, the server MUST allow only the Accept header values as defined in section 2.2.2.1. If the Accept header is not present, the response format in step 4 below MUST be XML. Any other header value MUST be ignored and the server MUST continue processing.
4. The server MUST construct a response in either XML or JSON format based on the value of the Accept header (section 2.2.2.1), or in XML format if the Accept header is not present. The response MUST include all of the complex types defined in section 2.2.4, and use the corresponding values defined in section 3.1.1.
5. If the server encounters an error in message processing, the server MUST return an HTTP error code in the 400 range. The body of the message response is insignificant to the protocol. Clients MUST halt processing upon receiving an HTTP error.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

4.1 Client Request

```
https://enterpriseregistration.contoso.com/enrollmentserver/contract?api-version=1.0
```

4.2 Server Response (XML)

```
<Discovery
  xmlns="http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities"
  xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <DeviceRegistrationService>
    <RegistrationEndpoint>
      https://sts.contoso.com/EnrollmentServer/DeviceEnrollmentWebService.svc
    </RegistrationEndpoint>
    <RegistrationResourceId>
      urn:ms-drs:sts.contoso.com
    </RegistrationResourceId>
    <ServiceVersion>1.0</ServiceVersion>
  </DeviceRegistrationService>
  <AuthenticationService>
    <OAuth2>
      <AuthCodeEndpoint>
        https://sts.contoso.com/adfs/oauth2/authorize
      </AuthCodeEndpoint>
      <TokenEndpoint>
        https://sts.contoso.com/adfs/oauth2/token
      </TokenEndpoint>
    </OAuth2>
  </AuthenticationService>
  <IdentityProviderService>
    <PassiveAuthEndpoint>
      https://sts.contoso.com/adfs/ls
    </PassiveAuthEndpoint>
  </IdentityProviderService>
</Discovery>
```

4.3 Server Response (JSON)

```
{"DeviceRegistrationService":{"RegistrationEndpoint":"https:\/\/sts.contoso.com\/EnrollmentSe
rver\/DeviceEnrollmentWebService.svc","RegistrationResourceId":"urn:ms-
drs:sts.contoso.com","ServiceVersion":"1.0"},"AuthenticationService":{"OAuth2":{"AuthCodeEndp
oint":"https:\/\/sts.contoso.com\/adfs\/oauth2\/authorize","TokenEndpoint":"https:\/\/sts.con
toso.com\/adfs\/oauth2\/token"}},"IdentityProviderService":{"PassiveAuthEndpoint":"https:\/\/
sts.contoso.com\/adfs\/ls"}}
```

5 Security

5.1 Security Considerations for Implementers

The Device Registration Discovery Protocol must use HTTPS as a transport. Using Secure Sockets Layer (SSL) server certificate verification ensures that the client is communicating with the real server and closes any possible man-in-the-middle attacks.

5.2 Index of Security Parameters

None.

6 Appendix A: Full XML Schema

For ease of implementation, the following sections provide the full XML schemas for this protocol.

Schema name	Prefix	Section
http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities	tns	6.1
http://tempuri.org	tns1	6.2

6.1 http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities Schema

```
<xs:schema
xmlns:tns="http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities"
targetNamespace="http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entitie
s" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="Discovery">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="DeviceRegistrationService"
nillable="true" type="tns:DeviceRegistrationServiceData"/>
      <xs:element minOccurs="0" maxOccurs="1" name="AuthenticationService" nillable="true"
type="tns:AuthenticationServiceData"/>
      <xs:element minOccurs="0" maxOccurs="1" name="IdentityProviderService" nillable="true"
type="tns:IdentityProviderServiceData"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="DeviceRegistrationServiceData">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="RegistrationEndpoint" type="xs:anyURI"/>
      <xs:element minOccurs="0" maxOccurs="1" name="RegistrationResourceId"
type="xs:anyURI"/>
      <xs:element minOccurs="0" maxOccurs="1" name="ServiceVersion" nillable="true"
type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="AuthenticationServiceData">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="OAuth2" nillable="true"
type="tns:OAuth2ServiceData"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="OAuth2ServiceData">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="AuthCodeEndpoint" type="xs:anyURI"/>
      <xs:element minOccurs="0" maxOccurs="1" name="TokenEndpoint" type="xs:anyURI"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="IdentityProviderServiceData">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="PassiveAuthEndpoint" type="xs:anyURI"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

6.2 http://tempuri.org Schema

```
<xs:schema xmlns:tns1="http://tempuri.org" targetNamespace="http://tempuri.org"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="DiscoverResponse" nillable="true"
xmlns:q1="http://schemas.datacontract.org/2004/07/Microsoft.DeviceRegistration.Entities"
type="q1:Discovery"/>
```

```
</xs:schema>
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

Note: Some of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.

- Windows 8.1 operating system
- Windows Server 2012 R2 operating system
- Windows Server 2016 Technical Preview operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated the product applicability list to include Windows Server 2016 Technical Preview.	Y	Content update.

9 Index