[MS-DSSP]:
Directory Services Setup Remote Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
10/22/2006	0.01		MCPP Milestone 1 Initial Availability
01/19/2007	1.0		MCPP Milestone 1
03/02/2007	1.1		Monthly release
04/03/2007	1.2		Monthly release
05/11/2007	1.3		Monthly release

Date	Revision History	Revision Class	Comments
06/01/2007	1.3.1	Editorial	Revised and edited the technical content.
06/01/2007	1.3.2	Minor	PG updates in response to TDIs
07/03/2007	1.3.3	Editorial	Revised and edited the technical content.
07/20/2007	1.3.4	Editorial	Revised and edited the technical content.
08/10/2007	1.3.5	Editorial	Revised and edited the technical content.
09/28/2007	1.4	Minor	Updated the technical content.
10/23/2007	1.5	Minor	Updated the technical content.
11/30/2007	1.5.1	Editorial	Revised and edited the technical content.
01/25/2008	1.5.2	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction

The Directory Services Setup Remote Protocol is a Microsoft proprietary client/server-based **remote procedure call (RPC)** protocol. The protocol exposes an RPC interface that a client can call to obtain domain-related computer state and configuration information.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Active Directory (AD)
Active Directory Domain
Backup Domain Controller (BDC)
Directory
Directory Service (DS)
Domain Controller (DC)
Endpoint
Forest
Globally Unique Identifier (GUID)
Microsoft Interface Definition Language (MIDL)
Mixed Mode
Native Mode
NetBIOS Name
Network Data Representation (NDR)
Operating System Upgrade
Opnum
Primary Domain Controller (PDC)
Read-Only Domain Controller
Remote Procedure Call (RPC)
RPC Protocol Sequence
RPC Transport
Server Message Block (SMB)
Universally Unique Identifier (UUID)
Well-Known Endpoint

The following terms are specific to this document:

Domain Membership Role: Quantifies the relationship between a computer and a domain. A computer can act in one of three roles:

- Joined: Linked to a domain for purposes of policy and security.
- Stand-alone: Not associated with any domain.
- **Domain Controller**: Linked to a domain and hosting that domain.

Domain Membership Role Change: It is possible to change the **domain membership role** of a computer. A stand-alone computer can become a domain-joined computer and vice versa. A computer that is not a **domain controller** can become a **domain controller**, and vice versa. The act of configuring a computer that is running a server operating system to be a **domain controller** is called "promotion." The act of configuring a **domain controller** to be a non-**domain controller** server is called "demotion."

Legacy Domain: A domain in which all the **domain controllers** are **legacy domain controllers**.

Legacy Domain Controller: A **domain controller** that supports the Security Account Manager Remote Protocol [MS-SAMR], but not the **Active Directory** protocols specified in [MS-ADTS] and [MS-DRSR].

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

1.2 References

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, http://www.opengroup.org/public/pubs/catalog/c706.htm

[MS-ADTS] Microsoft Corporation, "Active Directory Technical Specification", June 2007.

[MS-DRSR] Microsoft Corporation, "Directory Replication Service (DRS) Remote Protocol Specification", July 2007.

[MS-DTYP] Microsoft Corporation, "Windows Data Types", January 2007.

[MS-ERREF] Microsoft Corporation, "Windows Error Codes", January 2007.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", January 2007.

[MS-SAMR] Microsoft Corporation, "Security Account Manager (SAM) Remote Protocol Specification (Client-to-Server)", June 2007.

[MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol Specification", July 2007.

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

1.2.2 Informative References

No informative references have been specified for this protocol.

1.3 Protocol Overview (Synopsis)

This protocol provides a remote procedure call (RPC) interface for querying domain-related computer state and configuration data. The client end of the Directory Services Setup Remote Protocol is an application that issues method calls on the RPC interface. The server end of the Directory Services Setup Remote Protocol obtains and replies to the client with the requested data about the computer on which the server is running. If the client connects to and requests information about a **domain controller** for the **directory service**, this data includes the status of any pending promotion or demotion of that domain controller.

1.4 Relationship to Other Protocols

The Directory Services Setup Remote Protocol is dependent upon Microsoft remote procedure call (RPC) (Remote Procedure Call Protocol Extensions, as specified in [MS-RPCE]), which is used to communicate between computers on a network.

This protocol depends on the **Server Message Block (SMB)** Protocol, as specified in [MS-SMB], and TCP/IP protocols for sending messages on the wire.

No protocols depend on this protocol.

1.5 Prerequisites/Preconditions

This protocol is a remote procedure call (RPC)-based protocol and therefore has the prerequisites, as specified in [MS-RPCE], common to all RPC interfaces.

It is assumed that the client has the name of a computer that runs an implementation of this protocol before the protocol is invoked. How a client does this is outside the scope of this specification.

Security configuration for RPC usage is specified in section 5.1.

1.6 Applicability Statement

This protocol can be used to perform the following functions:

- Obtain the configuration information of the domain to which a computer is joined. The information includes the domain name and domain **globally unique identifier (GUID)**. This protocol can be used to query a domain controller to determine if it is a **Primary Domain Controller (PDC)** (or PDC emulator) or a **read-only domain controller**.
- Query the progress of the promotion or demotion of a domain controller.
- Retrieve the upgrade status of a domain controller. This information is only applicable for the upgrade of a legacy domain controller to a version of Windows that is able to host **Active Directory (AD)**.
- Retrieve the **domain membership role** type for the computer.

1.7 Versioning and Capability Negotiation

- **Supported Transports:** This protocol uses only remote procedure calls (RPC). The protocol supports the Server Message Block (SMB) transport. For more information, see section 2.1.
- **Protocol Version:** This protocol interface has a single version number of 0.0. An RPC client determines if a method is supported by attempting to invoke the method; if the method is not supported, the RPC server MUST return an "Opnum out of range" error<1> as specified in [C706] and [MS-RPCE].
- **Security and Authentication Methods:** Authentication and security are provided as specified in [MS-SMB] and [MS-RPCE]. Anonymous access can be allowed for some operations, as specified in **DsRolerGetPrimaryDomainInformation (Opnum 0) (section 3.2.5.1)**.

1.8 Vendor-Extensible Fields

This protocol does not define any vendor-extensible fields within the protocol itself.

1.9 Standards Assignments

Parameter	Value	Reference
Named pipe	\PIPE\lsarpc	Section 2.1
RPC Interface **UUID** for Directory Services Setup Remote Protocol	3919286a-b10c-11d0-9ba8-00c04fd92ef5	Section 2.1

No public standard assignments have been received for this protocol. All values used in these extensions are in private ranges specified in section 2.1.

2 Messages

The following sections specify how Directory Services Setup Remote Protocol messages are encapsulated on the wire, and common Directory Services Setup Remote Protocol data types.

2.1 Transport

This protocol MUST use the following remote procedure call (RPC) protocol sequence: RPC over Server Message Block (SMB) (ncacn_np), as specified in [MS-RPCE].

This protocol uses the following **well-known endpoints**. These **endpoints** are pipe names for RPC over SMB, as specified in [MS-RPCE]:

- \PIPE\lsarpc

A server MUST listen on RPC over the above-named pipe. A client MUST only attempt to connect to this protocol via RPC over the above-named pipe.<2>

For authentication and authorization services, both the requestor and responder of this protocol MUST use the SMB transport to communicate the identity of the requestor, as specified in [MS-SMB] section 3.2.4.2.3.

The requestor MUST NOT use the RPC-provided security-support-provider mechanisms (for authentication, authorization, confidentiality, or tamper-resistance services).

This protocol MUST use this universally unique identifier (UUID) interface (3919286a-b10c-11d0-9ba8-00c04fd92ef5). The interface version number is 0.0.

2.2 Common Data Types

In addition to remote procedure call (RPC) base types, the sections that follow use the definition of globally unique identifier (GUID), as specified in [MS-DTYP] Appendix A.

Additional data types that follow are defined in the **Microsoft Interface Definition Language (MIDL)** (as specified in section 6) for this RPC interface.

2.2.1 DSROLER_PRIMARY_DOMAIN_INFO_BASIC

The **DSROLER_PRIMARY_DOMAIN_INFO_BASIC** structure contains basic information, including the role of the computer, domain name, and globally unique identifier (GUID) of the domain.

```
typedef struct _DSROLER_PRIMARY_DOMAIN_INFO_BASIC {
  DSROLE_MACHINE_ROLE MachineRole;
  unsigned __int32 Flags;
  [unique, string] wchar_t* DomainNameFlat;
  [unique, string] wchar_t* DomainNameDns;
  [unique, string] wchar_t* DomainForestName;
  GUID DomainGuid;
} DSROLER_PRIMARY_DOMAIN_INFO_BASIC,
 *PDSROLER_PRIMARY_DOMAIN_INFO_BASIC;
```

MachineRole: The current role of the computer, expressed as a **DSROLE_MACHINE_ROLE** data type.

Flags: The value that indicates the state of the directory service and validity of the information contained in the **DomainGuid** member. The value of this parameter MUST be zero or a combination of one or more individual flags in the following table. The combination is the result of a bitwise OR of the flags that apply to the computer for which information is being retrieved. All undefined bits MUST be 0.

Value	Meaning
DSROLE_PRIMARY_DS_RUNNING 0x00000001	The directory service is running on this computer. If this flag is not set, the directory service is not running on this computer.
DSROLE_PRIMARY_DS_MIXED_MODE 0x00000002	The directory service is running in **mixed mode**. This flag is valid only if the DSROLE_PRIMARY_DS_RUNNING flag is set and the DSROLE_PRIMARY_DS_READONLY flag is not set.
DSROLE_PRIMARY_DS_READONLY 0x00000008	The computer holds a read-only copy of the **directory**. This flag is valid only if the DSROLE_PRIMARY_DS_RUNNING flag is set and the DSROLE_PRIMARY_DS_MIXED_MODE flag is not set.
DSROLE_PRIMARY_DOMAIN_GUID_PRESENT 0x01000000	The **DomainGuid** member contains a valid domain globally unique identifier (GUID). If this bit is not set, the value in DomainGuid member is undefined.

DomainNameFlat: The **NetBIOS name** of the domain to which the computer is joined.

DomainNameDns: The domain name of the computer. This member MUST be NULL if the **MachineRole** member is **DsRole_RoleStandaloneWorkstation** or **DsRole_RoleStandaloneServer** and MUST NOT be NULL otherwise.

DomainForestName: The name of the **forest** to which the computer belongs. This member MUST be NULL, if the computer is a stand-alone workstation or server.

DomainGuid: The universally unique identifier (UUID) of the domain to which the computer belongs. The value of this member is valid only if the DSROLE_PRIMARY_DOMAIN_GUID_PRESENT flag is set.

2.2.2 DSROLE_MACHINE_ROLE

The **DSROLE_MACHINE_ROLE** enumeration specifies the current role of the computer.

```
typedef  enum _DSROLE_MACHINE_ROLE
{
  DsRole_RoleStandaloneWorkstation = 0,
  DsRole_RoleMemberWorkstation = 1,
  DsRole_RoleStandaloneServer = 2,
  DsRole_RoleMemberServer = 3,
  DsRole_RoleBackupDomainController = 4,
  DsRole_RolePrimaryDomainController = 5
} DSROLE_MACHINE_ROLE;
```

DsRole_RoleStandaloneWorkstation: The computer is a stand-alone workstation.

DsRole_RoleMemberWorkstation: The computer is a workstation that is joined to a domain.

DsRole_RoleStandaloneServer: The computer is a stand-alone server.

DsRole_RoleMemberServer: The computer is a server that is joined to a domain.

DsRole_RoleBackupDomainController: The computer is a server that is a **backup domain controller** or a read-only domain controller.<3>

DsRole_RolePrimaryDomainController: The computer is a server that is the primary domain controller emulator.

2.2.3 DSROLE_OPERATION_STATE_INFO

The **DSROLE_OPERATION_STATE_INFO** structure contains the status of a pending domain controller role-change operation, if any, for the computer.

```
typedef struct _DSROLE_OPERATION_STATE_INFO {
  DSROLE_OPERATION_STATE OperationState;
} DSROLE_OPERATION_STATE_INFO,
 *PDSROLE_OPERATION_STATE_INFO;
```

OperationState: The role-change status of the computer domain membership, as specified by a **DSROLE_OPERATION_STATE** enumeration.

2.2.4 DSROLE_OPERATION_STATE

The **DSROLE_OPERATION_STATE** enumeration specifies values that determine if a domain controller promotion or demotion operation is currently being performed on a computer.<4>

```
typedef  enum _DSROLE_OPERATION_STATE
{
  DsRoleOperationIdle = 0,
  DsRoleOperationActive = 1,
  DsRoleOperationNeedReboot = 2
} DSROLE_OPERATION_STATE;
```

DsRoleOperationIdle: No promotion or demotion operation is currently being performed on the computer.

DsRoleOperationActive: A promotion or demotion operation is in progress.

DsRoleOperationNeedReboot: A promotion or demotion operation has been performed. The computer must be restarted to function in the new role.

2.2.5 DSROLE_UPGRADE_STATUS_INFO

The **DSROLE_UPGRADE_STATUS_INFO** structure contains information about the status of a pending **operating system upgrade**, if any, for the computer. This structure is intended to store only the status of operating system upgrade of a **legacy domain controller**.

```
typedef struct _DSROLE_UPGRADE_STATUS_INFO {
```

```
  unsigned __int32 OperationState;
  DSROLE_SERVER_STATE PreviousServerState;
} DSROLE_UPGRADE_STATUS_INFO,
 *PDSROLE_UPGRADE_STATUS_INFO;
```

OperationState: The current status of the upgrade. Valid values are shown in the following table:<5>

Value	Meaning
0	No upgrade is currently in progress.
DSROLE_UPGRADE_IN_PROGRESS 0x00000004	An upgrade is currently in progress.

PreviousServerState: The role of the computer prior to the upgrade. The value of this member is valid only if an upgrade is in progress (that is, if the **OperationState** member is set to DSROLE_UPGRADE_IN_PROGRESS).

2.2.6 DSROLE_SERVER_STATE

The **DSROLE_SERVER_STATE** enumeration specifies the role of the computer prior to the upgrade.

```
typedef  enum _DSROLE_SERVER_STATE
{
  DsRoleServerUnknown = 0,
  DsRoleServerPrimary = 1,
  DsRoleServerBackup = 2
} DSROLE_SERVER_STATE,
 *PDSROLE_SERVER_STATE;
```

DsRoleServerUnknown: The previous role of the computer is unknown.

DsRoleServerPrimary: The previous role of the computer was primary domain controller in a **legacy domain**.

DsRoleServerBackup: The previous role of the computer was backup domain controller in a legacy domain.

2.2.7 DSROLE_PRIMARY_DOMAIN_INFO_LEVEL

The **DSROLE_PRIMARY_DOMAIN_INFO_LEVEL** enumeration defines the information level that the client requests.

```
typedef  enum _DSROLE_PRIMARY_DOMAIN_INFO_LEVEL
{
  DsRolePrimaryDomainInfoBasic = 1,
  DsRoleUpgradeStatus = 2,
  DsRoleOperationState = 3
} DSROLE_PRIMARY_DOMAIN_INFO_LEVEL;
```

DsRolePrimaryDomainInfoBasic: Request for information about the domain to which the computer belongs.

DsRoleUpgradeStatus: Request for computer operating system upgrade status.

DsRoleOperationState: Request for computer operation state.

2.2.8 DSROLER_PRIMARY_DOMAIN_INFORMATION

The **DSROLER_PRIMARY_DOMAIN_INFORMATION** union contains one of three types of information about a computer.

```
typedef
[switch_type(DSROLE_PRIMARY_DOMAIN_INFO_LEVEL)]
union _DSROLER_PRIMARY_DOMAIN_INFORMATION {
  [case(DsRolePrimaryDomainInfoBasic)]
    DSROLER_PRIMARY_DOMAIN_INFO_BASIC DomainInfoBasic;
  [case(DsRoleUpgradeStatus)]
    DSROLE_UPGRADE_STATUS_INFO UpgradStatusInfo;
  [case(DsRoleOperationState)]
    DSROLE_OPERATION_STATE_INFO OperationStateInfo;
} DSROLER_PRIMARY_DOMAIN_INFORMATION,
 *PDSROLER_PRIMARY_DOMAIN_INFORMATION;
```

DomainInfoBasic: Basic information about a computer. For more information, see **DSROLER_PRIMARY_DOMAIN_INFO_BASIC (section 2.2.1)**.

UpgradStatusInfo: Information about the upgrade of the computer. For more information, see **DSROLE_UPGRADE_STATUS_INFO (section 2.2.5)**.

OperationStateInfo: Role-change status of the computer. For more information, see **DSROLE_OPERATION_STATE_INFO (section 2.2.3)**.

3 Protocol Details

The following sections specify details of the Directory Services Setup Remote Protocol, including abstract data models, timers, and message processing rules.

3.1 Client Details

The following sections specify client details of the Directory Services Setup Remote Protocol, including abstract data models, timers, and message processing rules.

3.1.1 Abstract Data Model

No abstract data model is used.

3.1.2 Timers

No protocol timers are required other than those internal ones used in remote procedure call (RPC) to implement resiliency to network outages, as specified in [MS-RPCE].

3.1.3 Initialization

The client creates a remote procedure call (RPC) association (or binding) to the server RPC before an RPC method is called. The client MAY create a separate association for each method invocation, or it MAY reuse an association for multiple invocations.<6>

3.1.4 Higher-Layer Triggered Events

There is a one-to-one correspondence between higher-layer triggered events and methods defined in section 3.1.5. When a higher layer requests a particular action, the associated method is passed to the remote procedure call (RPC) with all values as specified by the upper layer.

3.1.5 Message Processing Events and Sequencing Rules

The client SHOULD ignore errors returned from the remote procedure call (RPC) server and notify the application invoker of the error received in the higher layer.<7> Otherwise, no special message processing is required on the client beyond the processing required in the underlying RPC protocol.

3.1.6 Timer Events

No protocol timer events are required on the client other than the events maintained in the underlying **RPC transport**.

3.1.7 Other Local Events

No additional local events are used on the client other than the events maintained in the underlying RPC transport.

3.2 Server Details

The following sections specify server details of the Directory Services Setup Remote Protocol, including abstract data models, timers, and initialization.

3.2.1 Abstract Data Model

The following information is maintained by the server to respond to client queries.

The computer maintains abstract variables that contain the identity of the directory service domain and forest to which it belongs, if any, as follows:

- NetBIOSDomainName: The name of the domain or non-domain workgroup, as known by NetBIOS name, to which the computer belongs.
- DNSDomainName: The fully qualified Domain Name System (DNS) name of the domain to which the computer belongs. This abstract has value only for computers that are joined to a domain; otherwise, it is NULL.
- ForestName: The fully qualified Domain Name System (DNS) name of the forest to which the computer belongs. This variable has value only for computers that are joined to a domain; otherwise, it is NULL.
- DomainGUID: The universally unique identifier (UUID), as specified in [MS-DTYP], that identifies the domain to which the computer belongs. This variable has type GUID, as specified in [MS-DTYP],<8> and has value only for computers that are joined to a directory service domain; otherwise, the value is NULL.

The computer maintains information about its role and status in the domain, as follows:

- ComputerRole: An abstract variable of type **DSROLE_MACHINE_ROLE** that describes the current domain membership role of the machine.
- DirectoryServiceStatus: The status of the directory service running on the server. The type of this variable is unsigned __int32 whose value maps to the values of the **Flag** member of **DSROLER_PRIMARY_DOMAIN_INFO_BASIC**.
- ComputerOperationState: The status of the current ComputerRole change operation. The type of this variable is **DSROLE_OPERATION_STATE** enumeration.
- ComputerUpgrade: A Boolean abstract variable that keeps track of the status of the current upgrade operation. It is true only if the computer is in the process of upgrading a legacy domain controller to a domain controller which supports the Active Directory protocols specified in [MS-ADTS] and [MS-DRSR].
- PreviousServerState: The security role of the computer in the network prior to the most recent upgrade that has been initiated. This variable is set only if ComputerUpgrade is TRUE. The type of this variable is **DSROLE_SERVER_STATE** enumeration.

3.2.2 Timers

No protocol timer events are required on the server other than the timers required in the underlying RPC transport, as specified in [MS-RPCE].

3.2.3 Initialization

The server MUST listen on the well-known endpoint defined for this remote procedure call (RPC) interface. For more information, see section 2.1.

The server initializes the abstract state variables with values appropriate to its current role, upgrade status, and other properties.<9>

3.2.4 Higher-Layer Triggered Events

No higher-layer triggered events are used.

3.2.5 Message Processing Events and Sequencing Rules

For authenticated remote procedure call (RPC) over Server Message Block (SMB), the details of method authentication are specific to the underlying RPC implementation, as specified in [C706] section 13, [MS-RPCE] section 5, and [MS-SMB] section 5.

Opnums 1 through 11 are not used across the network. These opnums are reserved and MUST NOT be reused by non-Microsoft implementations.<10>

Methods in RPC Opnum Order

Method	Description
DsRolerGetPrimaryDomainInformation	The **DsRolerGetPrimaryDomainInformation** method returns the requested information about the current configuration or state of the computer on which the server is running. Opnum: 0

All methods MUST NOT throw exceptions.

3.2.5.1 DsRolerGetPrimaryDomainInformation (Opnum 0)

The **DsRolerGetPrimaryDomainInformation (Opnum 0)** method returns the requested information about the current configuration or state of the computer on which the server is running.

```
HRESULT DsRolerGetPrimaryDomainInformation(
  [in] handle_t hBinding,
  [in] DSROLE_PRIMARY_DOMAIN_INFO_LEVEL InfoLevel,
  [out, switch_is(InfoLevel)] PDSROLER_PRIMARY_DOMAIN_INFORMATION* DomainInfo
);
```

hBinding: As specified in [C706].

InfoLevel: The type of data requested by the client. For possible values in this enumeration, see section 2.2.7.

DomainInfo: The requested information that the server provides to the client. The value of the *InfoLevel* parameter indicates the type of information that is requested; information will be returned in the corresponding member of the **DSROLER_PRIMARY_DOMAIN_INFORMATION** union.

Return Values: The method returns 0 if successful; if failed, it returns a nonzero error code. The values transmitted in this field are implementation specific. All nonzero values MUST be treated the same for protocol purposes.<11>

This method obtains the identity and authorization information about the client from the underlying remote procedure call (RPC) runtime. Servers that implement this method SHOULD impose an authorization policy decision before performing the function.<12>

The server determines the appropriate response to the request by examining the *InfoLevel* parameter, setting the appropriate fields in the *DomainInfo* parameter, and sending the response to the caller.

The following describes which fields are used and what they contain for each *InfoLevel* value.

DsRolePrimaryDomainInfoBasic

When the *InfoLevel* is **DsRolePrimaryDomainInfoBasic**, the server MUST use the **DomainInfoBasic** field of the *DomainInfo* parameter, whose type is **DSROLER_PRIMARY_DOMAIN_INFO_BASIC**. The result MUST be constructed in the following manner:

1. Determine the role of the server and set the **MachineRole** field of **DomainInfoBasic** according to the ComputerRole state element. If the server ComputerRole state element indicates that it is not a stand-alone computer, set the **DomainNameFlat**, **DomainNameDNS**, **ForestName**, and **DomainGUID** fields of the DomainInfoBasic structure according to the NetBIOSDomainName, DNSDomainName, ForestName, and DomainGUID state information. If the DomainGUID state element is non-empty, the DSROLE_PRIMARY_DOMAIN_GUID_PRESENT bit must be set in the **Flags** member of DomainInfoBasic.

2. If the server is a stand-alone computer, set the **DomainNameFlat** field of DomainInfoBasic according to NetBIOSDomainName state information; and then set the other fields to NULL.

3. If the server is a domain controller and the directory service is enabled, set the **Flags** member of the DomainInfoBasic structure as follows:

    1. Set the DSROLE_PRIMARY_DS_RUNNING bit.

    2. If the domain is in mixed mode, set the DSROLE_PRIMARY_DS_MIXED_MODE bit.

    3. If the server is a read-only domain controller, set the DSROLE_PRIMARY_DS_READONLY bit.<13>

DsRoleUpgradeStatus

When InfoLevel is DsRoleUpgradeStatus, the server sets the requested information into the **UpgradeStatusInfo** field of the *DomainInfo* parameter, whose type is DSROLE_UPGRADE_STATUS_INFO. The result MUST be constructed in the following manner:

1. Set the OperationState to DSROLE_UPGRADE_IN_PROGRESS if the ComputerUpgrade state element is TRUE.

2. Set the PreviousServerState to reflect the role of the server prior to the upgrade, according to the PreviousServerState state element, if the UpgradeStatusInfo member's DSROLE_UPGRADE_IN_PROGRESS bit is set. If that bit is not set, set PreviousServerState to 0.

DsRoleOperationState

When InfoLevel is DsRoleOperationState, the server MUST return the result in the **OperationStateInfo** field of the *DomainInfo* parameter, whose type is DSROLE_OPERATION_STATE_INFO. The result MUST be constructed by setting the **OperationState** member of the OperationStateInfo structure according to the value of the ComputerOperationState state element.

3.2.6 Timer Events

No timer events are required on the server other than the events maintained in the underlying RPC transport.

3.2.7 Other Local Events

No additional local events are used on the server other than the events maintained in the underlying RPC transport.

4 Protocol Examples

The following is an example of a **DsRolerGetPrimaryDomainInformation** remote procedure call (RPC) method.

Assume the server is a workstation computer joined to a domain called MyDomainName.com.

The client calls the **DsRolerGetPrimaryDomainInformation** RPC method on the server with InfoLevel equal to 1.

The server returns with code 0x00000000; and with the **DomainInfoBasic** field of DomainInfo structure, the following values are in fields of **DomainInfoBasic**.

```
MachineRole = 1
Flags = 0x01000000
DomainNameFlat = "MyDomainName"
DomainNameDns = "MyDomainName.com"
DomainForestName = "MyDomainName.com"
DomainGuid = { 0x5585777b, 0xe549, 0x43b6,
{ 0xa8, 0x42, 0x2, 0xbe, 0xd, 0xd6, 0xab, 0x14 } };
```

5 Security

The following sections specify security considerations for implementers of the Directory Services Setup Remote Protocol.

5.1 Security Considerations for Implementers

Information returned by this protocol can reveal more than is appropriate for anonymous users, thus resulting in an information leak. Implementers should therefore determine whether to allow access to anonymous users.<14>

5.2 Index of Security Parameters

Security Parameter	Section
Remote procedure call (RPC) authentication.	Section 3.2.5
Allow anonymous users and non-administrative users to retrieve information using the **DsRolerGetPrimaryDomainInformation** RPC method.	Section 3.2.5.1

6 Appendix A: Full IDL

```
import "ms-dtyp.idl";

[
    uuid(3919286a-b10c-11d0-9ba8-00c04fd92ef5),
    version(0.0),
    pointer_default(unique)
]
interface dssetup
{
typedef enum _DSROLE_MACHINE_ROLE {
    DsRole_RoleStandaloneWorkstation,
    DsRole_RoleMemberWorkstation,
    DsRole_RoleStandaloneServer,
    DsRole_RoleMemberServer,
    DsRole_RoleBackupDomainController,
    DsRole_RolePrimaryDomainController
} DSROLE_MACHINE_ROLE;
typedef enum _DSROLE_SERVER_STATE {

    DsRoleServerUnknown = 0,
    DsRoleServerPrimary,
    DsRoleServerBackup
} DSROLE_SERVER_STATE, *PDSROLE_SERVER_STATE;
typedef enum _DSROLE_PRIMARY_DOMAIN_INFO_LEVEL {
    DsRolePrimaryDomainInfoBasic = 1,
    DsRoleUpgradeStatus,
    DsRoleOperationState
} DSROLE_PRIMARY_DOMAIN_INFO_LEVEL;
typedef struct _DSROLE_UPGRADE_STATUS_INFO {
    unsigned __int32 OperationState;
    DSROLE_SERVER_STATE PreviousServerState;
} DSROLE_UPGRADE_STATUS_INFO, *PDSROLE_UPGRADE_STATUS_INFO;
typedef enum _DSROLE_OPERATION_STATE {
    DsRoleOperationIdle = 0,
    DsRoleOperationActive,
    DsRoleOperationNeedReboot
} DSROLE_OPERATION_STATE;
typedef struct _DSROLE_OPERATION_STATE_INFO {
    DSROLE_OPERATION_STATE OperationState;
} DSROLE_OPERATION_STATE_INFO, *PDSROLE_OPERATION_STATE_INFO;

typedef struct _DSROLER_PRIMARY_DOMAIN_INFO_BASIC {
    DSROLE_MACHINE_ROLE MachineRole;
    unsigned __int32 Flags;
    [ unique, string ] wchar_t *DomainNameFlat;
    [ unique, string ] wchar_t *DomainNameDns;
    [ unique, string ] wchar_t *DomainForestName;
    GUID DomainGuid;
} DSROLER_PRIMARY_DOMAIN_INFO_BASIC,
 *PDSROLER_PRIMARY_DOMAIN_INFO_BASIC;
typedef [switch_type(DSROLE_PRIMARY_DOMAIN_INFO_LEVEL)] union
_DSROLER_PRIMARY_DOMAIN_INFORMATION {
    [case(DsRolePrimaryDomainInfoBasic)]
DSROLER_PRIMARY_DOMAIN_INFO_BASIC   DomainInfoBasic;
    [case(DsRoleUpgradeStatus)]
```

```
DSROLE_UPGRADE_STATUS_INFO UpgradStatusInfo;
    [case(DsRoleOperationState)]
DSROLE_OPERATION_STATE_INFO OperationStateInfo;
} DSROLER_PRIMARY_DOMAIN_INFORMATION,
*PDSROLER_PRIMARY_DOMAIN_INFORMATION;

DWORD
DsRolerGetPrimaryDomainInformation(
    [in] handle_t hBinding,
    [in] DSROLE_PRIMARY_DOMAIN_INFO_LEVEL InfoLevel,
    [out, switch_is( InfoLevel )]
PDSROLER_PRIMARY_DOMAIN_INFORMATION *DomainInfo );

/*The following methods are part of the dssetup
interface in Windows 2000, Windows XP RTM,
and Windows XP SP1 , they are not part of
this interface in Windows XP SP2 or later
service packs, Windows Server 2003 and Windows Vista.
These methods do not expose client server protocol.*/

void Opnum1NotUsedOnWire(void);
void Opnum2NotUsedOnWire(void);
void Opnum3NotUsedOnWire(void);
void Opnum4NotUsedOnWire(void);
void Opnum5NotUsedOnWire(void);
void Opnum6NotUsedOnWire(void);
void Opnum7NotUsedOnWire(void);
void Opnum8NotUsedOnWire(void);
void Opnum9NotUsedOnWire(void);
void Opnum10NotUsedOnWire(void);
void Opnum11NotUsedOnWire(void);
}
```

7 Appendix B: Windows Behavior

The information in this specification is applicable to the following versions of Windows:

- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista
- Windows Server 2008

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Windows behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that Windows does not follow the prescription.

<1> Section 1.7: Windows RPC protocol returns PC_S_PROCNUM_OUT_OF_RANGE to notify the client that an RPC method is out of range, as specified in [MS-RPCE].

<2> Section 2.1: Windows servers listen on all protocols bound to RPC. Windows clients attempt only to connect via RPC over the above-named pipe.

<3> Section 2.2.2: Read-only domain controllers are supported only on Windows Server 2008.

<4> Section 2.2.4: In the Windows implementation, after a promotion or demotion operation that requires a reboot has occurred and prior to that reboot being performed, the RPC interface used by this protocol may be unavailable or it may reject connections with authentication errors.

<5> Section 2.2.5: DSROLE_UPGRADE_IN_PROGRESS is only set for an operating system upgrade from a Windows NT 4.0 domain controller to a later release of Windows Server. In the Windows implementation, a Windows 2000 or later computer returns this under the following conditions: (1) it was previously a Windows NT 4.0 domain controller, (2) the operating system upgrade to Windows 2000 or later has completed, and (3) it has not yet transitioned to being a Windows 2000 or later domain controller.

<6> Section 3.1.3: This protocol configures the RPC runtime to perform a strict **NDR** data consistency check at target level 6.0 for Windows Vista and Windows Server 2008, and target level 5.0 for earlier releases of Windows, as specified in [MS-RPCE] section 3.

<7> Section 3.1.5: The Windows implementation ignores errors and passes them back to the invoker.

<8> Section 3.2.1: A Windows **Active Directory domain** has a domain GUID, and a Windows NT 4.0 **domain** does not have a domain GUID. Computers running Windows 2000 may be members of a Windows NT 4.0 domain.

<9> Section 3.2.3: This protocol configures the RPC runtime to perform a strict NDR data consistency check at target level 6.0 for Windows Vista and Windows Server 2008, and target level 5.0 for earlier releases of Windows, as specified in [MS-RPCE], section 3.

Starting from Windows Vista, this protocol configures the RPC runtime to reject a NULL unique or full pointer with non-zero conformant value, as specified in [MS-RPCE] section 3.

This protocol configures the RPC runtime via the strict_context_handle attribute to reject use of context handles created by a method of a different RPC interface than this one, as specified in [MS-RPCE] section 3.

<10> Section 3.2.5: Gaps in the opnum numbering sequence apply to Windows as follows:

Opnum	Description
1-11	Only used locally by Windows, never remotely.

<11> Section 3.2.5.1: This field may take on any Windows error code value, as specified in [MS-ERREF].

<12> Section 3.2.5.1: Windows domain controllers allow any authenticated or unauthenticated connection to invoke **DsRolerGetPrimaryDomainInformation**. Computers running the Windows operating system that are not domain controllers require the connection not to be anonymous.

<13> Section 3.2.5.1: Currently, read-only domain controllers are supported only on Windows Server 2008. The domain hosted by a read-only domain controller must be in **native mode**. Read-only domain controllers are not supported on Windows 2000 Server or Windows Server 2003.

<14> Section 5.1: An anonymous user can access **DsRolerGetPrimaryDomainInformation** on a domain controller on Windows Server 2008, Windows Server 2003, and Windows 2000, but not on a non-domain controller computer.

8 Index

T

V

W