Appendix 3A
Bindery APIs

BinderyAuthenticateWithHandle 2
BinderyCloseAuthenticationHandle 3
BinderyCreateAuthenticationHandle 4
BinderyGetAuthenticationInfo 6
BinderyGetInitialConnection 7
BinderyGetPreferredServer 8
BinderyQualifyConnectionMatch 9
BinderyResolveIdToObject 10
BinderyResolveNameToAddress 11
BinderyResolveObjectToId 13
BinderySetPreferredServer 15
BinderyUnauthenticate 16

BinderyAuthenticateWithHandle

Description	Authenticates a connection using a previously created authentication handle.

Syntax

```
#include "conn.h"
UINT32
BinderyAuthenticateWithHandle(
    VOID                  *authenHandle,
    CONN_HANDLE   connHandle)
```

Input

authenHandle	Authentication handle associated with the information that will be used to authenticate the connection.
connHandle	Connection to be authenticated.

Output

None.

Return values

SUCCESS_CODE
INVALID_CONNECTION
INVALID_AUTHEN_HANDLE
AUTHEN_FAILED

See also

BinderyCreateAuthenticationHandle
BinderyCloseAuthenticationHandle

BinderyCloseAuthenticationHandle

Description

Closes the specified authentication handle.

Syntax

```
#include "conn.h"
UINT32
BinderyCloseAuthenticationHandle(
    VOID    *authenHandle)
```

Input

authenHandle Authentication handle to be closed.

Output

None.

Return values

SUCCESS_CODE
INVALID_AUTHEN_HANDLE

See also

BinderyAuthenticateWithHandle
BinderyCreateAuthenticationHandle

BinderyCreateAuthenticationHandle

Description

Creates an authentication handle that can be used to automatically authenticate connections. *ProcessGroupID* and *processID* identify the scope of the authentication handle.

Syntax

```
#include "conn.h"
UINT32
BinderyCreateAuthenticationHandle(
    UINT32          processGroupID,
    UINT32          processID,
    SPECT_DATA      *objectName,
    SPECT_DATA      *password,
    SPECT_DATA      *domainName,
    VOID            *pAuthenSpecInfo,
    VOID            **authenHandle)
```

Input

processGroupID	ID of process group.
processID	ID of process.
objectName	Name of the object to be authenticated.
password	Clear-text password to be used in authenticating. This value should be stored encrypted. *password* can be specified in local code page or in Unicode.
domainName	Should be set to NULL for Bindery.
pAuthenSpecInfo	Should be set to NULL for Bindery.

Output

authenHandle	Authentication handle that will be passed in when a connection is to be authenticated with the above information.

Return values	SUCCESS_CODE OUT_OF_RESOURCES DUPLICATE_AUTHEN_HANDLE
Remarks	**DUPLICATE_AUTHEN_HANDLE** will be returned if input information already matches a previously returned authentication handle. **OUT_OF_RESOURCES** will be returned if there is not enough space to create a new authentication handle.
See also	BinderyAuthenticateWithHandle

BinderyGetAuthenticationInfo

Description

Returns authentication information associated with an authentication handle.

Syntax

```
#include "conn.h"
UINT32
BinderyGetAuthenInfo(
     VOID               *authenHandle,
     SPECT_DATA         *objectName,
     SPECT_DATA         *domainName,
     VOID               *pAuthenSpecInfo)
```

Input

authenHandle	Authentication handle for which to retrieve information.

Output

objectName	Output buffer to receive the name of the object that will be authenticated with this handle. On input, the *length* field of this structure must specify the number of bytes available in the output buffer to receive the object name. On output, if the buffer is too small then the *length* field will contain the number of bytes of buffer space needed by caller to retrieve the object name.
domainName	Ignored.
pAuthenSpecInfo	Ignored.

Return values

SUCCESS_CODE
INVALID_AUTHEN_HANDLE
MORE_DATA_ERROR

Remarks

MORE_DATA_ERROR will be returned if the buffers described by output parameters *objectName* and *objectType* are too small to receive returned information.

See also

BinderyCreateAuthenticationHandle
BinderyCloseAuthenticationHandle

BinderyGetInitialConnection

Description

Resolves supplied name to a transport address.

Syntax

```
UINT32
BinderyGetInitialConnection(
     UINT32            processGroupId,
     UINT32            processId,
     UINT32            transportType,
     CONN_HANDLE   *connHandle )
```

Input

processGroupId	Process group ID.
processId	Process ID.
reqTranType	Preferred or required transport type.
sessSvcType	Type of session required (such as NCP).

Output

connHandle	Handle to the established connection.

Return values

SUCCESS_CODE	Success
INVALID_PARAMETER	
RESOLVE_SVC_FAILED	Unable to resolve name

BinderyGetPreferredServer

Description

Returns the preferred server for the specified scope. The preferred server set in NET.CFG will be returned if the preferred server is not specified for the requested scope. *ProcessGroupID* and *processID* are used to specify the scope of the preferred server.

Syntax

```
#include "name_svc.h"
UINT32
BinderyGetPreferredServer(
    UINT32          processGroupID,
    UINT32          processID,
    SPECT_DATA      *servername)
```

Input

processGroupID	ID for process group.
processID	ID for process.

Output

servername	Points to the buffer to receive the null-terminated preferred server for the specified scope.

Return values

SUCCESS_CODE
MORE_DATA_ERROR

Remarks

MORE_DATA_ERROR is returned if the caller's output buffer is too small to receive preferred server name.

See also

BinderySetPreferredServer

BinderyQualifyConnectionMatch

Description

Called by ConnMan so that when it needs to open a connection, it knows which field of the connection entry to try and match an existing connection with.

Syntax

```
UINT32
BinderyQualifyConnectionMatch(
    SPECT_DATA     *serverName,
    UINT32         *connEntryId,
    SPECT_DATA     *qualifiedServerName)
```

Input

serverName	Name of server to fully qualify.

Output

connEntryId	Contains the server name field ID.
qualifiedServerName	*serverName* string copied into this structure unmodified.

Return values

SUCCESS_CODE	Success
INVALID_PARAMETER	

BinderyResolveIdToObject

Description

Resolves the object ID on the given connection to its object name and object type.

Syntax

```
UINT32
BinderyResolveIdToObject(
    CONN_HANDLE   reqConnId,
    UINT32        objectId,
    VOID              *reqNSSpec,
    SPECT_DATA        *repObjectName,
    SPECT_DATA        *repObjectType)
```

Input

reqConnId	Connection handle where the object ID exists.
objectId	ID of object to qualify.
reqNSSpec	Ignored by Bindery.

Output

reqObjectName	Pointer to object to which name ID was resolved.
reqObjectType	Pointer to the type of service to which object ID was resolved.

Return values

SUCCESS_CODE
INVALID_CONNECTION
INVALID_PARAMETER
RESOLVE_SVC_FAILED

BinderyResolveNameToAddress

Description

Resolves a given NetWare name to a transport address. *processGroupID* and *processID* specify the preferred server connection to use if *reqConnHandle* is NULL.

Syntax

```
#include "name_svc.h"
UINT32
BinderyResolveNameToAddress(
    CONN_HANDLE   connHandle,
    SPECT_DATA        *objectName,
    SPECT_DATA        *objectType,
    UINT32            transportType,
    VOID              *nameSvcSpec,
    UINT32            repSessionSvcID,
    TRAN_ADDR_TYPE    *repTranAddr,
    UINT32            *repTranAddrCount )
```

Input

connHandle	Connection handle to resolve name with. Cannot be NULL.
objectName	NetWare Bindery name to resolve to a transport address.
objectType	Type of NetWare Bindery name being resolved. Currently the only support type is NCP_SERVER.
transportType	The preferred or required transport type. Must be one of the following: TRAN_TYPE_IPX TRAN_TYPE_IP TRAN_TYPE_WILD
nameSvcSpec	Point to name service-specific information. Should be NULL for Bindery.

Output

repSessionSvcID	ID of the session protocol on which the resolved name is valid.
repTranAddr	Points to array of TRAN_ADDR_TYPE entries to be filled in with the transport addresses of the

resolved name.

repTranAddrCount
The actual number of TRAN_ADDR_TYPE entries being returned to caller. On input, specifies the number of transport address entries available to receive from the name service provider.

Return values

SUCCESS_CODE
INVALID_PARAMETER
RESOLVE_NAME_FAILED
MORE_DATA_ERROR

Remarks

It is possible for a name to be resolved to multiple transport addresses of different transport types. The caller must then decide which transport address to use, since this will determine which transport is used for communication with the server.

See also

BinderyResolveObjectToId

BinderyResolveObjectToId

Description

Resolves a given NetWare object name to an object ID and transport address(es). *processGroupID* and *processID* specify the preferred server connection to use if *reqConnHandle* is NULL.

Syntax

```
#include "name_svc.h"
UINT32
BinderyResolveObjectToId(
    CONN_HANDLE   connHandle,
    SPECT_DATA         objectName,
    SPECT_DATA         objectType,
    UINT32                 transportType,
    VOID                    *nameSvcSpec,
    UINT32                 *repObjectID,
    UINT32                 *repSessionSvcID,
    TRAN_ADDR_TYPE  *repTranAddr,
    UINT32                 *repTranAddrCount )
```

Input

connHandle	Connection handle for which to resolve name. Cannot be NULL.
objectName	NetWare object name to resolve to an ID or transport address.
objectType	Type of NetWare object name being resolved. Currently the only support types are: USER GROUP QUEUE NCP_SERVER
transportType	Preferred or required transport type. Must be one of the following: TRAN_TYPE_IPX TRAN_TYPE_IP TRAN_TYPE_WILD
nameSvcSpec	Should be set to NULL for Bindery.

Output

repObjectID	Object ID of object name on resolved address.
repSessionSvcID	ID of session protocol that object ID is valid on.

repTranAddr	Points to array of TRAN_ADDR_TYPE entries to be filled in with the transport addresses of the resolved object.
repTranAddrCount	Contains the actual number of TRAN_ADDR_TYPE entries being returned to caller. On input, specifies the number of transport address entries available to receive from the name service provider.

Return values

SUCCESS_CODE
INVALID_PARAMETER
RESOLVE_NAME_FAILED
MORE_DATA_ERROR

Remarks

It is possible for a name to be resolved to multiple transport addresses of different transport types. The caller must then decide which transport address to use, since this will determine which transport is used for communication with the server.

See also

BinderyResolveNameToAddress

BinderySetPreferredServer

Description	Sets the preferred server for the specified scope. If a preferred server is already specified for the indicated scope then this information will overwrite the previous setting.

Syntax

```
#include "name_svc.h"
UINT32
BinderySetPreferredServer(
    UINT32          processGroupID,
    UINT32          processID,
    SPECT_DATA      *servername )
```

Input

processGroupID	ID for process group.
processID	ID for process.
servername	Name of preferred server to set for specified scope.

Output

None.

Return values

SUCCESS_CODE
INVALID_PARAMETER
OUT_OF_RESOURCES

Remarks

INVALID_PARAMETER is returned if the name being set is too big for the name service provider it's being set for.

OUT_OF_RESOURCES is returned if the name service provider does not have enough memory to store the preferred server for the specified scope.

See also

BinderyGetPreferredServer

BinderyUnauthenticate

Description	Unauthenticates a connection.

Syntax

```
#include "conn.h"
UINT32
BinderyUnauthenticate(
    CONN_HANDLE   connHandle)
```

Input	*connHandle*	Connection to be unauthenticated.
Output	None.	

Return values

SUCCESS_CODE
INVALID_CONNECTION

See also BinderyAuthenticateWithHandle