Appendix 13A
Authentication Multiplexor API

AUTHAuthenticate 2

AUTHAuthenticateWithHandle 4

AUTHCloseAuthenticationHandle 5

AUTHCreateAuthenticationHandle 6

AUTHEnumerateSvc 8

AUTHRegisterSvc 10

AUTHScanAuthenticationHandles 11

AUTHUnauthenticate 13

AUTHUnregisterSvc 14

AUTHAuthenticate

Description Authenticates a connection without first creating an authentication handle.

Syntax

```
#include "conn.h"
UINT32
AUTHAuthenticate(
    UINT32          processGroupID,
    UINT32          processID,
    CONN_HANDLE     connHandle,
    UINT32          authenSvcID,
    SPECT_DATA      *username,
    SPECT_DATA      *password,
    SPECT_DATA      *domainName,
    VOID            *pAuthenSpecInfo );
```

Input

processGroupID	ID of process group.
processID	ID of process.
connHandle	Connection that is to be authenticated.
authenSvcID	Unique ID of authentication module to use to authenticate with.
username	The name of the user that is being authenticated. Username can be specified in local code page or in Unicode.
password	The clear text password to be used in authenticating. This value should be stored in an encrypted fashion. Password can be specified in local code page or in Unicode.
domainName	Domain the user exists in. For bindery this can be NULL or a servername, for NDS this is a treename, and for PNW this is a workgroup name.

pAuthenSpecInfo	Each authentication type needs information that is specific to that type of authentication. For example, for Bindery authentication the object type is needed. This parameter allows information specific to the authentication type to be passed in. The first DWORD of this pointer should contain the number of bytes to follow of specific authentication information.

Output None.

Return values

SUCCESS_CODE
INVALID_CONNECTION
AUTHEN_FAILED

Remarks Input parameters *processGroupID* and *processID* identify the scope of the authentication.

See also **AUTHAuthenticateWithHandle**

AUTHAuthenticateWithHandle

Description

Authenticate a connection using a previously created authentication handle.

Syntax

```
#include "conn.h"

UINT32
AUTHAuthenticateWithHandle(
    AUTH_HANDLE    authenHandle,
    CONN_HANDLE    connHandle );
```

Input

authenHandle The authentication handle associated with the information that will be used to authenticate this connection.

connHandle Connection that is to be authenticated.

Output

None.

Return values

SUCCESS_CODE
INVALID_CONNECTION
INVALID_AUTHEN_HANDLE
AUTHEN_FAILED

Remarks

See also

AUTHCreateAuthenticationHandle
AUTHFreeAuthenticationHandle

AUTHCloseAuthenticationHandle

Description This service allows a caller to close the specified authentication handle.

Syntax

```
#include "conn.h"

UINT32
AUTHCloseAuthenticationHandle(
    AUTH_HANDLE    authenHandle );
```

Input *authenHandle* Authentication handle to be closed.

Output None.

Return values

SUCCESS_CODE
INVALID_AUTHEN_HANDLE

Remarks

See also **AUTHCreateAuthenticationHandle**
AUTHAuthenticateWithHandle

AUTHCreateAuthenticationHandle

Description

Allows a caller to create an authentication handle that can be used to automatically authenticate connections using the given information. This interface is designed to allow multiple authentication handles to be created, which will allow the client to keep authentication information for multiple entities such as NDS trees.

Syntax

```
#include "conn.h"

UINT32
AUTHCreateAuthenticationHandle(
    UINT32          processGroupID,
    UINT32          processID,
    UINT32          authenSvcID,
    SPECT_DATA  *username,
    SPECT_DATA  *password,
    SPECT_DATA  *domainName,
    VOID            *pAuthenSpecInfo,
    AUTH_HANDLE     *authenHandle );
```

Input

processGroupID ID of process group.

processID ID of process.

authenSvcID Unique ID of authentication service module to use.

username The name of the user to be authenticated with this handle. Username can be specified in local code page or in Unicode.

password The clear text password to be used in authenticating with the returned handle. This value should be stored in an encrypted fashion. Password can be specified in local code page or in Unicode.

domainName Domain the user exists in. For bindery this can be NULL or a servername, for NDS this is a treename, and for PNW this is a workgroup name.

pAuthenSpecInfo Each authentication type needs information that is specific to that type of authentication. For example

		for BINDERY authentication the object type is needed. This parameter allows authentication type specific information to be passed in. The first DWORD of this pointer should contain the number of bytes to follow of specific authen info.
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

Remarks

Input parameters *processGroupID* and *processID* identify the scope of the authentication handle.

A return code of **OUT_OF_RESOURCES** will be returned by an authentication service module if it does not have the space to create an authentication handle with the supplied information.

See also

AUTHAuthenticateWithHandle
AUTHFreeAuthenticationHandle

AUTHEnumerateSvc

Description

Lists the currently registered authentication service modules providing authentication service support. Besides the unique ID of the authentication service module, this call also returns a copy of the authentication service module's description block which provides additional information describing the authentication service module.

Syntax

```
#include "conn.h"
UINT32
AUTHEnumerateAuthenticationSvc(
    UINT32                  *enumHandle,
    UINT32                  *authenSvcID,
    AUTH_SVC_DESC_BLOCK     *authenSvcDescBlk );
```

Input

enumHandle Handle to be used to retrieve the next authentication service module that has registered authentication service support. This value should initially be set to zero. The output of this function will be the next handle to use on subsequent calls to this function.

Output

enumHandle Handle to use on the next iteration to find the next authentication service module that has registered authentication service support.

authenSvcID Pointer to receive the unique ID of the next authentication service module that has registered authentication service support.

authenSvcDescBlk
Pointer to data structure to receive a copy of the description block that this authentication service module has registered with ConnMan. This parameter can be set to NULL if this information is not needed by caller.

Return values

SUCCESS_CODE
INVALID_PARAMETER
NO_MORE_ENTRIES

Remarks

This service will return INVALID_PARAMETER if enumHandle is invalid. If no more authentication service modules have registered authentication service support, the error NO_MORE_ENTRIES is returned.

See also

AUTHRegisterAuthenticationSvc
AUTHUnregisterAuthenticationSvc

AUTHRegisterSvc

Description

Allows authentication service modules to register their authentication service API support.

Syntax

```
#include "conn.h"
UINT32
AUTHRegisterSvc(
    UINT32                      authenSvcID,
    AUTH_SVC_API_SET_TYPE       *authenApiSet,
    AUTH_SVC_DESC_BLOCK         *authenSvcDescBlk );
```

Input

authenSvcID	Unique ID assigned to an authentication service module.
authenApiSet	Pointer to array of functions that an authentication service module must implement.
authenSvcDescBlk	Pointer to authentication service module's description block which provides additional information describing this authentication service module.

Output

None.

Return values

SUCCESS_CODE
AUTHEN_SVC_ALREADY_REGISTERED

Remarks

See also

AUTHUnregisterAuthenticationSvc

AUTHScanAuthenticationHandles

Description

Allows caller to discover which authentication handles are available to authenticate a connection with. Other values returned along with the authentication handle are the authentication type, user name, and other information specific to the authentication type.

Syntax

```
#include "conn.h"

UINT32
AUTHScanAuthenticationHandles(
    UINT32          processGroupID,
    UINT32          processID,
    UINT32          *scanHandle,
    AUTH_HANDLE     *authenHandle,
    UINT32          *authenSvcID,
    SPECT_DATA      *username,
    SPECT_DATA      *domainName,
    VOID            *pAuthenSpecInfo );
```

Input

processGroupID ID of process group.

processID ID of process.

scanHandle Handle to be used to retrieve the next authentication handle. This value should initially be set to zero. The output of this service will be the next handle to use on subsequent calls to this function.

Output

authenHandle Authentication handle.

authenSvcID Unique ID of authentication service module that created this authentication ID.

username Output buffer to receive the name of the user that will be authenticated with this handle. On input the *length* field of this structure must specify the number of bytes available in buffer to receive username. 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 username.

	domainName	Output buffer to receive the name of the domain the user exists in. On input the *length* field of this structure must specify the number of bytes available in buffer to receive domainName. 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 domainName.
	pAuthenSpecInfo	Each authentication type needs information that is specific to that type of authentication. For example, BINDERY authentication requires an object type. This parameter returns authentication type specific information. The first DWORD of this pointer should contain the number of bytes of buffer space available to store returned information into.

Return values

SUCCESS_CODE
INVALID_PARAMETER
MORE_DATA_ERROR
NO_MORE_ENTRIES

Remarks

Input parameters *processGroupID* and *processID* are used to specify the scope of the authentication handle being scanned for.

This service will return **INVALID_PARAMETER** if *scanHandle* is invalid. If no more authentication handles are available (they have all been scanned) then **NO_MORE_ENTRIES** is returned to caller.

An error of **MORE_DATA_ERROR** is returned if buffers described by *username*, *domainName*, and *pAuthenSpecInfo* are too small to receive returned information.

See also

AUTHCreateAuthenticationHandle
AUTHFreeAuthenticationHandle

AUTHUnauthenticate

Description Unauthenticates a connection.

Syntax

```
#include "conn.h"

UINT32
AUTHUnauthenticate(
    CONN_HANDLE    connHandle);
```

Input *connHandle* Connection to be unauthenticated.

Output None.

Return values

SUCCESS_CODE
INVALID_CONNECTION

Remarks

See also **ConnAuthenticate**
ConnAuthenticateWithHandle

AUTHUnregisterSvc

Description Allows an authentication service module to unregister its authentication service API support.

Syntax

```
#include "conn.h"
UINT32
AUTHUnregisterAuthenticationSvc(
    UINT32      authenSvcID );
```

Input

authenSvcID Unique ID assigned to authentication service module that is being unregistered.

Output None.

Return values

SUCCESS_CODE
AUTHEN_SVC_NOT_REGISTERED

Remarks

See also AUTHRegisterAuthenticationSvc