[MS-NNS]: .NET NegotiateStream Protocol

This topic lists the Errata found in [MS-NNS] since it was last published. Since this topic is updated frequently, we recommend that you subscribe to these RSS or Atom feeds to receive update notifications.

Errata are subject to the same terms as the Open Specifications documentation referenced.

RSS
Atom

Errata below are for Protocol Document Version V6.0 – 2016/07/14.

Errata Published*	Description
2016/07/18	In Section 2.2.2, Data Message, changed from: This section defines the structure of the data exchange messages. These messages are used to transfer application-specific data after the handshake phase is complete. The .NET NegotiateStream Protocol only frames application data using the format noted in the following table if the negotiation of security services during the handshake phase resulted in both the client and server agreeing to sign or encrypt and sign the data to be transferred. Thus, if the negotiated security context in the handshake phase results in a context that **does not support** message confidentiality or integrity, then the data transferred is not framed, and does not follow the format specified in this section (that is, application-supplied data is written directly to the underlying TCP stream). Changed to (change in **bold**): This section defines the structure of the data exchange messages. These messages are used to transfer application-specific data after the handshake phase is complete. The .NET NegotiateStream Protocol only frames application data using the format noted in the following table if the negotiation of security services during the handshake phase resulted in both the client and server agreeing to sign or encrypt and sign the data to be transferred. Thus, if the negotiated security context in the handshake phase results in a context that **supports neither** message confidentiality nor integrity, then the data transferred is not framed, and does not follow the format specified in this section (that is, application-supplied data is written directly to the underlying TCP stream). In Section 3.1.4.1, Application Invocation of the .NET NegotiateStream Protocol, changed from: … If the function returns any major_status other than GSS_S_COMPLETE, the implementation MUST notify the application of the failure without sending anything over the Underlying TCP Connection. Otherwise, the implementation MUST store the returned credential handle as the Client Credentials, and MUST set the Stream State to CreatingSecurityToken. The implementation MUST pass the Client Credentials to the GSS_Init_sec_context function ([RFC2743] section 2.2.1). The input_context_handle parameter MUST be GSS_C_NO_CONTEXT. The targ_name parameter MUST be the Target Name. The mech_type parameter MUST be the same as that passed to GSS_Acquire_cred. The deleg_req_flag MUST be true if and only if Allowed Impersonation Level is Delegation. The conf_req_flag MUST be true if and only if the Required Protection Level is EncryptAndSign. The integ_req_flag MUST be true if and only if the Required Protection Level is Sign or EncryptAndSign. The mutual_req_flag, replay_det_req_flag, and sequence_req_flag MUST be true. The anon_req_flag MUST be false. The chan_bindings parameter MUST be the Channel Binding Token. The input_token MUST be NULL, and the lifetime_req MUST be 0. Changed to: … If the function returns any major_status other than GSS_S_COMPLETE, the implementation MUST notify the application of the failure without sending anything over the Underlying TCP

Errata Published*	Description
	Connection. Otherwise, the implementation MUST store the returned credential handle as the Client Credentials, and MUST set the Stream State to CreatingSecurityToken. The implementation MUST pass the Client Credentials to the GSS_Init_sec_context function ([RFC2743] section 2.2.1). The input_context_handle parameter MUST be GSS_C_NO_CONTEXT. The targ_name parameter MUST be the Target Name. The mech_type parameter MUST be the same as that passed to GSS_Acquire_cred. The deleg_req_flag MUST be true if and only if Allowed Impersonation Level is Delegation. The conf_req_flag MUST be true if and only if the Required Protection Level is EncryptAndSign. The integ_req_flag MUST be true if and only if the Required Protection Level is Sign or EncryptAndSign. The mutual_req_flag, replay_det_req_flag, and sequence_req_flag MUST be true. The anon_req_flag MUST be false. The chan_bindings parameter MUST be the Channel Binding Token. The input_token MUST be NULL, and the lifetime_req MUST be 0. If the conf_avail return value is true, the integ_avail return value MUST also be true, and the Negotiated Protection Level is EncryptAndSign. If the conf_avail return value is false and the integ_avail return value is true, the Negotiated Protection Level is Sign. Otherwise, the Negotiated Protection Level is None.

*Date format: YYYY/MM/DD