# [MS-ADTS]: Active Directory Technical Specification

**This topic lists the Errata found in the MS-ADTS document 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 V41.0 – 2015/06/30.

| Errata Published* | Description |
|---|---|
| 2015/10/12 | In Section 6.2.2.3.4.5, nTDSConnection Creation, clarified the pseudocode to show when failed DCs are filtered out.<br><br>Changed from:<br><br>`...`<br><br>`CALL CreateConnection(cr, rbh, t, lbh, e.ReplInfo, sched, partialReplicaOkay)`<br>`ENDFOR`<br><br>`RETURN connected`<br>`}`<br>`...`<br>`CreateConnection(IN crossRef cr, IN nTDSDSA rbh, IN interSiteTransport t, IN nTDSDSA lbh, IN REPLINFO ri, IN SCHEDULE sch, INOUT SEQUENCE<GUID> keepConnections)`<br>`{`<br>`LET rsiteGuid be the objectGUID of the site object ancestor of rbh`<br>`LET lsiteGuid be the objectGUID of the site object ancestor of lbh`<br><br>`LET rbhsAll be the result of GetAllBridgeheadDCs(rsiteGuid, cr, t, partialReplicaOkay, FALSE)`<br>`LET rbhsAvail be the result of GetAllBridgeheadDCs(rsiteGuid, cr, t, partialReplicaOkay, detectFailedDCs)`<br>`LET lbhsAll be the result of GetAllBridgeheadDCs(lsiteGuid, cr, t, partialReplicaOkay, FALSE)`<br>`LET lbhsAvail be the result of GetAllBridgeheadDCs(lsiteGuid, cr, t, partialReplicaOkay, detectFailedDCs)`<br>`...`<br><br>Changed to:<br><br>`...`<br><br>`CALL CreateConnection(cr, rbh, t, lbh, e.ReplInfo, sched, detectFailedDCs, partialReplicaOkay, keepConnections)`<br>`ENDFOR`<br><br>`RETURN connected`<br>`}`<br><br>`...` |

| Errata Published* | Description |
|---|---|
| | CreateConnection(IN crossRef cr, IN nTDSDSA rbh,<br>IN interSiteTransport t, IN nTDSDSA lbh, IN REPLINFO ri,<br>IN SCHEDULE sch, IN bool detectFailedDCs, IN bool partialReplicaOkay,<br>INOUT SEQUENCE<GUID> keepConnections)<br>{<br>LET rsiteGuid be the objectGUID of the site object ancestor of rbh<br>LET lsiteGuid be the objectGUID of the site object ancestor of lbh<br><br>LET rbhsAll be the result of GetAllBridgeheadDCs(rsiteGuid, cr,<br>t, partialReplicaOkay, FALSE)<br>LET lbhsAll be the result of GetAllBridgeheadDCs(lsiteGuid, cr,<br>t, partialReplicaOkay, FALSE) |
| 2015/09/28 | In Section 6.2.2.4, Removing Unnecessary Connections, clarified when and how a KCC deletes a connection.<br>Changed from:<br>Given an nTDSConnection object cn, if the DC with the nTDSDSA object dc that is the parent object of cn and the DC with the nTDSDA object referenced by cn!fromServer are in the same site, the KCC on dc deletes cn if all of the following are true:<br>▪ Bit NTDSCONN_OPT_IS_GENERATED is clear in cn!options.<br>▪ No site settings object s exists for the local DC's site, or bit NTDSSETTINGS_OPT_IS_TOPL_CLEANUP_DISABLED is clear in s!options.<br>▪ Another nTDSConnection object cn2 exists such that cn and cn2 have the same parent object, cn!fromServer = cn2!fromServer, and either<br>▪ cn!whenCreated < cn2!whenCreated<br>▪ cn!whenCreated = cn2!whenCreated and cn!objectGUID < cn2!objectGUID<br>▪ Bit NTDSCONN_OPT_RODC_TOPOLOGY is clear in cn!options<br>Given an nTDSConnection object cn, if the DC with the nTDSDSA object dc that is the parent object of cn and the DC with the nTDSDSA object referenced by cn!fromServer are in different sites, a KCC acting as an ISTG in dc's site deletes cn if all of the following are true:<br>▪ Bit NTDSCONN_OPT_IS_GENERATED is clear in cn!options.<br>▪ cn!fromServer references an nTDSDSA object for a DC in a site other than the local DC's site.<br><br>Changed to:<br>Given an nTDSConnection object cn, if the DC with the nTDSDSA object dc that is the parent object of cn and the DC with the nTDSDA object referenced by cn!fromServer are in the same site, the KCC on dc deletes cn if all of the following are true:<br>▪ Bit NTDSCONN_OPT_IS_GENERATED is set in cn!options.<br>▪ No site settings object s exists for the local DC's site, or bit NTDSSETTINGS_OPT_IS_TOPL_CLEANUP_DISABLED is clear in s!options.<br>▪ Another nTDSConnection object cn2 exists such that cn and cn2 have the same parent object, cn!fromServer = cn2!fromServer, and either<br>▪ cn!whenCreated < cn2!whenCreated<br>▪ cn!whenCreated = cn2!whenCreated and cn!objectGUID < cn2!objectGUID<br>▪ Bit NTDSCONN_OPT_RODC_TOPOLOGY is clear in cn!options<br>Given an nTDSConnection object cn, if the DC with the nTDSDSA object dc that is the parent object of cn and the DC with the nTDSDSA object referenced by cn!fromServer are in different sites, a KCC acting as an ISTG in dc's site deletes cn if all of the following are true:<br>▪ Bit NTDSCONN_OPT_IS_GENERATED is set in cn!options.<br>▪ cn!fromServer references an nTDSDSA object for a DC in a site other than the local DC's site. |

| Errata Published* | Description |
| --- | --- |
| 2015/08/17 | In Section 6.2.2.3.4.4, Spanning Tree Computation, modified the pseudo code for BridgeheadDCFailed so that the detectFailedDCs field is not a default value but a switch for detection.<br><br>Changed from:<br><br>/***** BridgeheadDCFailed *****/<br>/* Determine whether a given DC is known to be in a failed state.<br>* IN: objectGUID - objectGUID of the DC's nTDSDSA object.<br>* IN: detectFailedDCs - TRUE if and only failed DC detection is<br>* enabled.<br>* RETURNS: TRUE if and only if the DC should be considered to be in a<br>* failed state.<br>*/<br>BridgeheadDCFailed(IN GUID objectGUID, IN bool detectFailedDCs) : bool<br>{<br>IF bit NTDSSETTINGS_OPT_IS_TOPL_DETECT_STALE_DISABLED is set in<br>the options attribute of the site settings object for the local<br>DC's site<br>RETURN FALSE<br>ELSEIF a tuple z exists in the kCCFailedLinks or<br>kCCFailedConnections variables such that z.UUIDDsa =<br>objectGUID, z.FailureCount > 1, and the current time -<br>z.TimeFirstFailure > 2 hours<br>RETURN TRUE<br>ELSE<br>RETURN detectFailedDCs<br>ENDIF<br>}<br><br>Changed to:<br><br>/***** BridgeheadDCFailed *****/<br>/* Determine whether a given DC is known to be in a failed state.<br>* IN: objectGUID - objectGUID of the DC's nTDSDSA object.<br>* IN: detectFailedDCs - TRUE if and only if failed DC detection is<br>* enabled.<br>* RETURNS: TRUE if and only if the DC should be considered to be in a<br>* failed state.<br>*/<br>BridgeheadDCFailed(IN GUID objectGUID, IN bool detectFailedDCs) : bool<br>{<br>IF detectFailedDCs is FALSE<br>RETURN FALSE<br>ENDIF<br><br>IF bit NTDSSETTINGS_OPT_IS_TOPL_DETECT_STALE_DISABLED is set in |

| Errata Published* | Description |
|---|---|
| | the options attribute of the site settings object for the local DC's site<br>RETURN FALSE<br>ENDIF<br><br>IF a tuple z exists in the kCCFailedLinks or kCCFailedConnections variables such that z.UUIDDsa = objectGUID, z.FailureCount > 1, and the current time - z.TimeFirstFailure > 2 hours<br>RETURN TRUE<br>ENDIF<br><br>RETURN FALSE<br>} |
| | In Section 6.2.2.5, Connection Translation, revised the value assignment for uuidDsa during KCC connection translation from a 'GUID based DNS name' to a 'GUID'.<br><br>Changed from:<br><br>If s and the local DC's nTDSDSA object are in the same site, cn!transportType has no value, or the RDN of cn!transportType is CN=IP:<br><br>▪ Bit DRS_MAIL_REP in t.replicaFlags is clear.<br><br>▪ t.uuidTransport = NULL GUID.<br><br>▪ t.uuidDsa = The GUID-based DNS name of s.<br><br>Otherwise:<br><br>▪ Bit DRS_MAIL_REP in t.replicaFlags is set.<br><br>▪ If x is the object with dsname cn!transportType, t.uuidTransport = x!objectGUID.<br><br>▪ Let a be the attribute identified by x!transportAddressAttribute. If a is the dNSHostName attribute, t.uuidDsa = the GUID-based DNS name of s. Otherwise, t.uuidDsa = (s!parent)!a.<br><br>Finally, the KCC calls IDL_DRSReplicaAdd to add a tuple u to n!repsFrom for each IDL_DRSGetNCChanges server "implied" by the nTDSConnection object children of the local DC's nTDSDSA object if such a u does not already exist. For each such nTDSConnection cn, a tuple u is implied if all of the following are true: |

| Errata Published* | Description |
|---|---|
| | …<br><br>Changed to:<br><br>If s and the local DC's nTDSDSA object are in the same site, cn!transportType has no value, or the RDN of cn!transportType is CN=IP:<br><br>▪ Bit DRS_MAIL_REP in t.replicaFlags is clear.<br><br>▪ t.uuidTransport = NULL GUID.<br><br>▪ t.uuidDsa = s!objectGUID.<br><br>Otherwise:<br><br>▪ Bit DRS_MAIL_REP in t.replicaFlags is set.<br><br>▪ If x is the object with dsname cn!transportType, t.uuidTransport = x!objectGUID.<br><br>▪ Let a be the attribute identified by x!transportAddressAttribute. If a is the dNSHostName attribute, t.uuidDsa = s!objectGUID. Otherwise, t.uuidDsa = (s!parent)!objectGUID.<br><br>Finally, the KCC calls IDL_DRSReplicaAdd to add a tuple u to n!repsFrom for each IDL_DRSGetNCChanges server "implied" by the nTDSConnection object children of the local DC's nTDSDSA object if such a u does not already exist. For each such nTDSConnection cn, a tuple u is implied if all of the following are true:<br><br>… |
| 2015/08/03 | In Section 7.6.2.4, Performing an LDAP Unbind Against a Directory Server, corrected the label of the input parameter from 'TaskInputLdapMessage' to 'TaskInputRequestMessage'.<br>Changed from:<br>4. Invoke the Performing an LDAP Operation Against a Directory Server (section 7.6.2.5) task with the following parameters: TaskInputConnectionInfo is set to the TaskInputConnectionInfo that was passed to this task and TaskInputLdapMessage is set to ldapRequest.<br>Changed to:<br>4. Invoke the Performing an LDAP Operation Against a Directory Server (section 7.6.2.5) task with the following parameters: TaskInputConnectionInfo is set to the TaskInputConnectionInfo that was passed to this task and TaskInputRequestMessage is set to ldapRequest. |
| 2015/08/03 | In two sections related to the SPN uniqueness checking logic, updated the text to account for the availability of Windows Server 2012 R2 with [MSKB-3070083]. |

<table>
<tr><th>Errata Published*</th><th>Description</th></tr>
<tr><td></td><td>
In Section 3.1.5.1.3, Uniqueness Constraints, changed from:
<ul>
<li>In AD DS, if the DC functional level is DS_BEHAVIOR_WIN2012R2 or greater, then the new attribute value must be unique within the entire forest. If the DC is not a GC, then the DC should issue an LDAP search against a GC to determine uniqueness. The following additional considerations for uniqueness checking are relevant for Windows Server 2016 Technical Preview:
<ul>
<li>…</li>
<li>Neither userPrincipalName nor servicePrincipalName uniqueness is checked if the DoNotVerifyUPNAndOrSPNUniqueness character of the dsHeuristics attribute is set to "3".</li>
</ul></li>
</ul>
Changed to:
<ul>
<li>In AD DS, if the DC functional level is DS_BEHAVIOR_WIN2012R2 or greater, then the new attribute value must be unique within the entire forest. If the DC is not a GC, then the DC should issue an LDAP search against a GC to determine uniqueness. The following additional considerations for uniqueness checking are relevant for Windows Server 2012 R2 with [MSKB-3070083] and Windows Server 2016 Technical Preview:
<ul>
<li>…</li>
<li>Neither userPrincipalName nor servicePrincipalName uniqueness is checked if the DoNotVerifyUPNAndOrSPNUniqueness character of the dsHeuristics attribute is set to "3".</li>
<li>userPrincipalName and servicePrincipalName uniqueness is checked if the DoNotVerifyUPNAndOrSPNUniqueness character of the dsHeuristics attribute is set to any value other than "1", "2", or "3".</li>
</ul></li>
</ul>
In Section 6.1.1.2.4.1.2, dSHeuristics, changed from:
<table>
<tr><th>Character number</th><th>Character name</th><th>Description</th></tr>
<tr><td>21</td><td>DoNotVerifyUPNAndOrSPNUniqueness</td><td>In AD LDS, if this character is anything other than "0", AD LDS will not check values of userPrincipalName for uniqueness. See section 3.1.1.5.2.2.In AD LDS, this heuristic applies to Windows Server 2003, Windows Server 2008, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, and Windows Server 2016 Technical Preview.In AD DS, if this character is "1", "2" or "3", AD DS will not check values of userPrincipalName or servicePrincipalName for uniqueness. See section 3.1.1.5.1.3.In AD DS, this heuristic applies to Windows Server 2016 Technical Preview.</td></tr>
</table>
Changed to:
</td></tr>
</table>

| Errata Published* | Description |
|---|---|

| Character number | Character name | Description |
|---|---|---|
| 21 | DoNotVerifyUPNAndOrSPNUniqueness | In AD LDS, if this character is anything other than "0", AD LDS will not check values of userPrincipalName for uniqueness. See section 3.1.1.5.2.2.In AD LDS, this heuristic applies to Windows Server 2003, Windows Server 2008, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, and Windows Server 2016 Technical Preview.In AD DS, if this character is "1", "2" or "3", AD DS will not check values of userPrincipalName or servicePrincipalName for uniqueness. See section 3.1.1.5.1.3.In AD DS, this heuristic applies to Windows Server 2012 R2 with [MSKB-3070083] and Windows Server 2016 Technical Preview. |

*Date format: YYYY/MM/DD