Enabling Remote Management of SQL Server Integration Services

Michael@SchmittDotNet.com

www.SchmittDotNet.com

Version 1.0
10/14/2010
Copyright and Disclaimers
This guide is for informational purposes only. THE AUTHOR MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of the author.

© 2010 Michael Schmitt (www.SchmittDotNet.com). All rights reserved.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>10/14/2010</td>
<td>Michael Schmitt</td>
<td>Initial version.</td>
</tr>
</tbody>
</table>

© 2010 Michael Schmitt
Table of Contents

Copyright and Disclaimers ................................................................. 2

History 2

Table of Contents ............................................................................. 3

Scenario ................................................................................................. 4

Tasks 5

Configuring Firewall Rules ................................................................. 5

Configuring DCOM Permissions .......................................................... 15
**Scenario**

We want to manage a remote instance of SQL Server Integration Services using SQL Server Management Studio. Here, *SQL Server 2008 R2 Management Studio* is installed on a notebook running *Windows 7*. The *SQL Server 2008 R2 Integration Services* are installed in a virtual machine running *Windows Server 2008 R2*. (The virtual machine is hosted by Hyper-V.)

Windows Firewall is active on both the notebook and the server. Neither the notebook nor the server is member of a domain. For this scenario, the notebook is called *red* while the server is called *bts2010*. On both machines, there is a local user called *mschmitt* which is member of the local *Administrators* group. The passwords of the user account mschmitt on both machines coincide.

Using SQL Server Management Studio on the notebook, try to connect to the Integration Services on the server.

![Connect to Server](image.png)

The connection will fail. The error message is:

*Cannot connect to bts2010.*

*Failed to retrieve data for this request. (Microsoft.SqlServer.Management.Sdk.Sfc)*

*Connecting to the Integration Services service on the computer "bts2010" failed with the following error: “Access is denied.”*

*This error occurs when the computer has not been configured to allow remote connections through DCOM, or the user does not have permission to access the SQL Server Integration Services service through DCOM.*
Tasks

Tasks that enable remote management of Integration Services include the following:

- Configuring the Windows Firewall on the server machine to allow remote connections through DCOM to the Integration Services service.
- Configuring DCOM permissions on the server machine to allow remote connections through DCOM to the Integration Services service.

Configuring Firewall Rules

The SQL Server Management Studio needs to submit remote procedure calls (RPC) to the Integration Services service running on the server machine. So we have to configure the appropriate firewall rules on the server machine.

On the server machine, open up **Administrative Tools -> Windows Firewall with Advanced Security**. Open up the **Inbound Rules** pane.

In the **Actions** pane, click **New Rule**.
Choose a **Port** rule. Click **Next**.
Choose the **TCP** protocol. Choose **Specific local ports** and enter **135**. Click **Next**.
Choose **Allow the connection**. Click **Next**.
Choose the appropriate firewall profile(s). In this scenario, we only allow computers connected to a **Private** network location. Click **Next**.
Enter **RPC (TCP/135)** as the **Name** of the new firewall rule. Click **Finish**.
We need another inbound firewall rule. So click New Rule once again.

Choose Program. Click Next.
Choose **This program path** and enter

```
%ProgramFiles%\Microsoft SQL Server\100\DTS\Binn\MsDtsSrvr.exe
```

Click **Next**.
Choose **Allow the connection**. Click **Next**.
Choose the appropriate firewall profile(s). In this scenario, we only allow computers connected to a **Private** network location. Click **Next**.
Enter **SQL Server Integration Services service** as the **Name** of the new rule. Click **Finish**.

![New Inbound Rule Wizard](image)

**Configuring DCOM Permissions**

In addition to inbound firewall rules allowing access to the Integration Services service, we also need DCOM permissions to enable access to this service. DCOM permissions are granted to the appropriate users.
Here, our local user mschmitt will be allowed to remotely manage Integration Services. On the server machine, add an appropriate local user (mschmitt in this scenario) to the local user group **Distributed COM Users**.

![Distributed COM Users Properties](image)

If you are logged on as the user who is granted permissions, log off and log on again so that the changes will become effective.
Besides DCOM permissions on computer level, we also have to grant DCOM permissions on application level, namely on the Integration Services DCOM application MsDtsServer100.

On the server machine, open up **Administrative Tools** -> **Component Services**. Expand the **DCOM Config** node.

Right-click the **MsDtsServer100** application and choose **Properties**.
Choose the **Security** tab. In the **Launch and Activation Permissions** region, choose **Customize** and click **Edit**.

![MsDtsServer100 Properties](image-url)

Learn more about [setting these properties](image-url).
Add the user who will remotely manage Integration Services (mschmitt in this scenario) and Allow Local Launch, Remote Launch, Local Activation and Remote Activation for this user. Click OK.
Back in the **Security** tab, in the **Access Permissions** region choose **Customize**. Click **Edit**. Again, add the user who will remotely manage Integration Services (**mschmitt** in this scenario) and **Allow** both **Local Access** and **Remote Access** for this user. Click **OK**.

![Access Permissions](image)

Confirm the changes by clicking **OK** in the MsDtsServer100 **Properties** dialog.

Typically, a restart of the server machine is required for the changes to apply.

*So restart the server machine now!*

As a final test, open up SQL Server Management Studio on the notebook and connect to the Integration Services on the server. Now it should work.
Enabling Remote Management of SQL Server Integration Services

Microsoft SQL Server Management Studio

File Edit View Debug Tools Window Community

New Query

Object Explorer

Connect

bts2010 (Integration Services 10.50.1600 - red\mschmitt)

- Running Packages
- Stored Packages
  - File System
- MSDB
  - Data Collector
    - Generated
    - PerfCountersCollect
    - PerfCountersUpload
    - QueryActivityCollect
    - QueryActivityUpload
    - SqlTraceCollect
    - SqlTraceUpload
    - TSQLQueryCollect
    - TSQLQueryUpload
  - Maintenance Plans