myLittleBackup
for SQL Server 2000, 2005 & 2008
Backup/Restore Solution
For Shared SQL Server Hosting

myLittleBackup for SQL Server 2000, 2005 & 2008

Installation Guide

version 1.5

CONTENT

OVERVIEW

myLittleBackup for SQL Server 2000, 2005 & 2008 is a web-based tool specially designed and developed for shared hosting companies. It allows web hosting companies to offer their customers an easy, secure and quick solution to backup/restore SQL Server 2000, 2005 and 2008 databases.

CONFIGURATION REQUIREMENTS

myLittleBackup for SQL Server requires the following configuration:

- Microsoft Windows 2000, 2003, 2008
- Microsoft IIS 5, 6, 7
- Microsoft SQL Server 2000, 2005 & 2008 (successfully tested with Microsoft SQL Server 7.0)
- Microsoft .NET framework 2.0
- A web browser that supports XHTML and CSS2

INSTALLATION AND CONFIGURATION

1. Open the .zip file and extract it to a folder within the scope of your site. Be sure to extract all the sub-folders.
2. Copy your license file (license.config) in the root folder (installation folder) of myLittleBackup
3. Setup your installation folder as a web application (you can do this through your IIS control panel or your web-host control panel). If you do not have the rights to setup your installation folder as a web application, then you must copy the whole content of the /bin folder into the /bin folder of your website.
4. Edit the config.xml and the web.config files (both located at the root of the application) and follow the instructions from the next sections.
5. You can now access myLittleBackup at the following address http://www.yourServer.com/yourInstallFolder/

Note: if you're running myLittleBackup in **Medium Trust**, then you need to add telerik r.a.d. upload to the Global Assembly Cache (GAC):

1. Go to Start -> Settings -> Control Panel -> Administrative Tools -> Microsoft .NET Framework 2.0 Configuration

2. Expand the nodes and locate Assembly cache
3. Right-click on it and select Add:
4. Browse to the myLittleBackup bin directory and ctrl-highlight RadUpload.Net2.dll
5. Click Open

(http://www.telerik.com/help/radupload/v2_NET2/installAddToGlobalAssemblyCache.html)

CONFIG.XML

Config.xml file is divided in several nodes. The nodes that you need to configure are described below.

<sqlservers>

In this node, you will define connection attributes to all your SQL Servers. You can enter as many SQL Servers as it is allowed by your license.
Each SQL Server is defined by its own node:

```
<sqlserver
   address="IP Address | Instance name"
   name="name as it appears in the connection page"
   sysadminconnection="true | false"
   sysadminconnectionid="unique id"
   backupfolder="backup folder full path (can be UNC path)"
   uploadfolder="upload folder full path (can be UNC path)"
   compatibilitylevel="80 | 90 | 100"
   backupcompression="true | false" (with SQL Server 2008 only)
/>
```

address: Enter the instance name or the IP address of the SQL Server.

name: Enter the name of the SQL Server as you want it to appear in the connection page.

sysadminconnection: true if you want to use a SysAdmin connection to perform backup/restore, else false. If you set this value to true, then you must also define the sysadminconnectionid attribute and the SysAdmin connection section in the web.config file.

sysadminconnectionid: a unique integer to indentify the SysAdmin connection used for this SQL Server.

backupfolder: specify the full path to the folder where backup files will be created for <u>this specific SQL Server</u>. Both SQL Server and IIS must be able to access this folder and must have read/write permissions. It means that you need to use UNC path if IIS and SQL Server are not on the same box. Note that ASPNET user must have read permission on this folder.
This value overwrites backup\defaultfolder value.

uploadfolder: specify the full path to the folder where you will upload the backup files you want to use for restore for <u>this specific SQL Server</u>. Both SQL Server and IIS must be able to access this folder and must have read/write permissions. It means that you need to use UNC path if IIS and SQL Server are not on the same box.
This value overwrites restore\defaultfolder value.

compatibilitylevel: 80 for SQL Server 2000, 90 for SQL Server 2005. This attribute is important when you use myLittleBackup to backup a SQL Server 2000 database and to restore it on a SQL Server 2005 database.
This value overwrites backup\dbproperties\compatibilitylevel value.

backupcompression: true if you want to use SQL Server 2008 backup compression feature, else false. Note this parameter is ignored with SQL Server 2000 and 2005.

<u>Examples</u>:
```
<sqlservers>
   <sqlserver
      address=”P27SRV\SQL2K5”
      name=”Prod. SQL Server 2005”
      sysadminconnection=”true”
      sysadminconnectionid=”1”
      backupfolder=”\\SQL2K5\dbbackup\”
      uploadfolder=”\\SQL2K5\dbupload\”
      compatibilitylevel="90"
   />
   <sqlserver
      address="P27SRV/SQL2K"
      name="SQL Server 2000"
      sysadminconnection="false"
      backupfolder="\\P27SRV\dbbackup\"
      uploadfolder="\\P27SRV\dbupload\"
      compatibilitylevel="80"
   />
   <sqlserver
      address="P27SRV\SQL2K8"
      name="Prod. SQL Server 2008"
      sysadminconnection="true"
      sysadminconnectionid="2"
      backupfolder="\\SQL2K8\dbbackup\"
      uploadfolder="\\SQL2K8\dbupload\"
      compatibilitylevel="100"
      backupcompression="true"
   />
   <sqlserver
      address="192.0.0.10"
      name="SQL Server 7.0"
      sysadminconnection="true"
      sysadminconnectionid="3"
      compatibilitylevel="70"
   />
</sqlservers>
```

<backup>

In this node you will define where the backup files will be created and the authorization rules.

```
<backup
   defaultfolder="backup folder full path (can be UNC path)"
   maxbackupcount="max number of allowed backup files per database"
>
   <authorizations>
      <authorization dbrole="comma separated list of roles" />
      …
      <authorization dbrole="comma separated list of roles" />
   </authorizations>
</backup>
```

defaultfolder: specify the full path to the folder where backup files will be created. Both SQL Server and IIS must be able to access this folder and must have read/write permissions. It means that you need to use UNC path if IIS and SQL Server are not on the same box. Note that ASPNET user must have read permission on this folder.
This value is used when sqlservers\server\ backupfolder is not defined, otherwiser it is overwritten.

maxbackupcount: specify the maximum number of backup files per database the user can store. When there are more files, the user must use the "Manage backup files" tool to delete old files.

<**authorizations**>: This node contains all the rules that define backup authorization.

<**authorization**>: Each authorization node defines a rule.
dbrole: enter a comma separated list with all the database roles the connected user must be member of in order to be allowed to perform backup.

When authorization rules have been defined, it is possible to let all kind of users to perform backup. Thus, sysadminconnection attribute must have been set to true for the current SQL Server so that the sysadmin connection will be used instead of the current user one.

```
Examples:
<backup
   defaultfolder="\\Server\dbbackup\"
   maxbackupcount="1"
>
   <authorizations>
      <authorization dbrole="db_owner" />
      <authorization dbrole="db_backupoperator" />
      <authorization dbrole="db_datareader, db_datawriter" />
   </authorizations>
</backup>
```

In this sample, members of db_owner and db_backupoperator database roles will be allowed to perform backup. Members of both db_datareader and db_datawriter database roles will also be allowed to perform backup.

<restore>

In this node, you will define where the backup files will be uploaded before restore, authorization rules and several properties to be checked after the restore, including user mapping.

```
<restore
   defaultfolder="upload folder full path (can be UNC path)"
   maxuploadcount="max number of allowed uploaded files per database"
   checkmaxsize="true | false"
   disclaimer="true | false"
>
   <authorizations>
      <authorization dbrole="comma separated list of roles" />
      …
      <authorization dbrole="comma separated list of roles" />
   </authorizations>
   <usermapping
      autofix="true | false"
      mappingmode="0 | 1 | 2 | 3"
      mappinglogin="login value"
      restoreoriginalusers="true | false"
      newuserdbrole=" comma separated list of roles"
   />
   <dbproperties>
      <compatibilitylevel default="80 | 90" />
      <isreadonly default="true | false" />
      <isautocloseon default="true | false" />
      <isautoshrinkon default="true | false" />
      <recoverymodel default="SIMPLE | BULK_LOGGED | FULL" />
   </dbproperties>
   <customsql
      enable="true | false"
      command="customized T-SQL command, %dbname% variable can be used"
   />
</restore>
```

defaultfolder: specify the full path to the folder where you will upload the backup files you want to use for restore. Both SQL Server and IIS must be able to access this folder and must have read/write permissions. It means that you need to use UNC path if IIS and SQL Server are not on the same box.
This value is used when sqlservers\server\ uploadfolder is not defined, otherwiser it is overwritten.

maxuploadcount: specify the maximum number of uploaded files per database the user can store. When there are more files, the user must use the "Manage backup files" tool to delete old files.

checkmaxsize: when set to true, it is not possible to restore a backup file which would exceed database file size limit. For instance, if your plan offers a 100 Mb database, you must set this attribute to true so that your customers are not able to restore from their backup files if the restored database should exceed this limit.

disclaimer: when set to true, a disclaimer is displayed before starting the restore process. Disclaimer text can be set by editing the disclaimer.txt file in the installation folder.

<**authorizations**>: This node contains all the rules that define restore authorization.

<**authorization**>: Each authorization node defines a rule.
dbrole: enter a comma separated list with all the database roles the connected user must be member of in order to be allowed to perform restore.

<usermapping>: This node contains information about how user mapping will be perform if ever the restored database contains orphan users.

autofix: true if you want to auto-fix orphan users when possible, else false.

mappingmode: numeric value which defined the mapping mode to be used:

- **0**: no mapping.
- **1**: orphan users are mapped to sa. (not recommended)
- **2**: orphan users are mapped to the current user login. (recommended)
- **3**: orphan users are mapped to the login specified by the mappinglogin attribute

mappinglogin: the login to be used to map orphan users when mappingmode is set to 3

restoreoriginalusers: when set to true, original users are restored after the the restore process. This option avoids that users defined by the webhoster disappear because of a restore process.

newuserdbrole: enter a comma separated list with the database roles that will be assigned to new users (i.e. the users that did not exist in the original database).

<dbproperties>: This nodes contains the database properties to check and the values to use.

compatibilitylevel: 80 for SQL Server 2000, 90 for SQL Server 2005. This attribute is important when you use myLittleBackup to backup a SQL Server 2000 database and to restore it on a SQL Server 2005 database. **This value is used when sqlservers\server\ compatibilitylevel is not defined, otherwiser it is overwritten.**

isreadonly: true if you want to set the database to readonly, else false (highly recommended)

isautocloseon: true if you want the database to auto-close, else false (highly recommended)

isautoshrinkon: true if you want the database to auto-shrink, else false.

recoverymodel: the recovery model to be applied to the database after the restore. Can be SIMPLE, BULK_LOGGED or FULL. If this value is not specified, then the recovery model used before the restore will be applied.

<customsql>: This node offers the possibility to launch a custom T-SQL command just after the restore process.

enable: true if the T-SQL command specified must be executed after the restore process, else false.

command: the T-SQL command to be executed just after the restore. This command can use the *%dbname%* variable.

Example:

```
<restore
   defaultfolder=”\\Server\dbupload\”
   maxuploadcount=”1”
   checkmaxsize=”true”
>
   <authorizations>
      <authorization dbrole=”db_owner” />
      <authorization dbrole=”db_datareader, db_datawriter,  db_ddladmin” />
   </authorizations>
   <usermapping
      autofix=”true”
      mappingmode=”3”
      mappinglogin=”orphanuser”
   />
   <dbproperties>
      <compatibilitylevel default=”90” />
      <isreadonly default=”false” />
      <isautocloseon default=”false” />
      <isautoshrinkon default=”false” />
      <recoverymodel default=”SIMPLE” />
   </dbproperties>
   <customsql
```

```
      enable="true"
      command="EXEC sp_myproc %dbname%"
   />
</restore>
```

<check>

In this node, you will define authorization rules to use the Check database tool.

```
<check>
   <authorizations>
      <authorization dbrole="comma separated list of roles" />
      …
      <authorization dbrole="comma separated list of roles" />
   </authorizations>
</check>
```

<**authorizations**>: This node contains all the rules that define authorization to use the Check database tool.
<**authorization**>: Each authorization node defines a rule.

dbrole: enter a comma separated list with all the database roles the connected user must be member of in order to be allowed to use the Check database tool.

```
Example:
<check>
   <authorizations>
      <authorization dbrole="db_owner" />
      <authorization dbrole="db_datareader, db_datawriter,  db_ddladmin" />
   </authorizations>
</check>
```

<log>

In this node, you will define if you want to write logs.

```
<log
   enable="true | false"
   folder="log folder full path (can be UNC path)"
/>
```

enable: true if you want to write log files, else false.

folder: specify the full path to the folder where log files will be written when enable has been set to true. IIS must be able to access this folder and must have read/write permissions. myLittleBackup writes one log file per day.

```
Example:
<log
   enable="true"
   folder="d:\myLittleBackup\log\"
/>
```

WEB.CONFIG

If you choose to SysAdmin connection to perform backup/restore (highly recommended), then you will need to add a few lines to the web.config file: in the <**appSettings**> node, you will need to add 2 lines per SQL Server.

```
   <add key="SysAdminLogin_xx" value="login" />
   <add key="SysAdminPassword_xx" value="password" />
```

SysAdminLogin_*xx*: ***xx*** is the numeric value defined in the **sysadminconnectionid** attribute (see config.xml / sqlservers paragraph). Set this value to the sysadmin login you want to use.

SysAdminPassword_*xx*: ***xx*** is the numeric value defined in the **sysadminconnectionid** attribute (see config.xml / sqlservers paragraph). Set this value to the password of the sysadmin login previously defined.

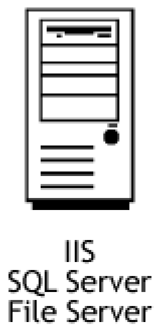
```
Examples:
<appSettings>
   <add key="Version" value="1.5" />
   <add key="Release" value="0039" />
   <add key="Build" value="3162" />
   <!-- Sysadmin connection value //-->
   <add key="SysAdminLogin_1" value="sa" />
   <add key="SysAdminPassword_1" value="x23yPOx" />
   <add key="SysAdminLogin_3" value="sa" />
   <add key="SysAdminPassword_3" value="po85FC8Dgc" />
</appSettings>
```

Note: If you chose to use SysAdmin connection, the SysAdmin connection is checked each time a user logs in myLittleBackup. If this check fails, then an error message appears.

4 TYPICAL CONFIGURATIONS

We recommend using configuration #4 when possible.

Config. #1: IIS, SQL Server, Storage on the same machine



config. #1

This is the easiest environment you can have. IIS and SQL Server are located on the same machine. Backup files will be stored and uploaded on this same machine.

- You do not need to use UNC to define backup\defaultfolder and restore\defaultfolder attributes.
- You do not need to use impersonation.
- **You must give read/write permissions to ASPNET user for the backup and upload folders**.
- Your SQL Server services can log on as Local System Account.

Your config.xml file will look like

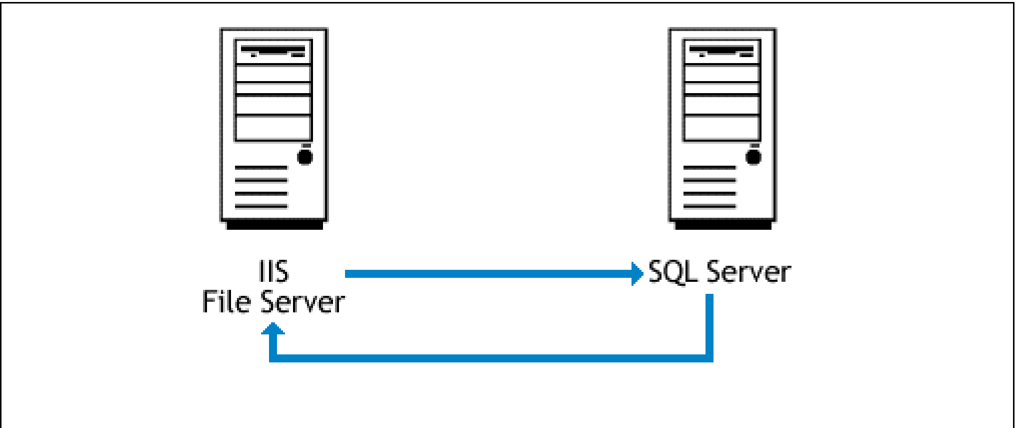
```
<myLittleBackup>
   <sqlservers>
      <sqlserver
         address="SrvName\InstanceName"
         name="SQL2005"
         sysadminconnection="true"
         sysadminconnectionid="1"
```

```
          />
    </sqlservers>
    <backup
       defaultfolder="c:\dbBackup\"
       maxbackupcount="1"
    >
       <authorizations>
          <authorization dbrole="db_owner" />
          <authorization dbrole="db_backupoperator" />
          <authorization dbrole="db_datareader, db_datawriter" />
       </authorizations>
    </backup>

    <restore
       defaultfolder="c:\dbUpload\"
       maxuploadcount="1"
       checkmaxsize="true"
    >
       <authorizations>
          <authorization dbrole="db_owner" />
          <authorization dbrole="db_datareader,db_datawriter,db_ddladmin" />
       </authorizations>
       <usermapping
          autofix="true"
          mappingmode="2"
       />
       <dbproperties>
          <compatibilitylevel default="90" />
          <isreadonly default="false" />
          <isautocloseon default="false" />
          <isautoshrinkon default="false" />
       </dbproperties>
    </restore>
    <check>
       <authorizations>
          <authorization dbrole="db_owner" />
          <authorization dbrole="db_datareader, db_datawriter" />
       </authorizations>
    </check>
    <log
       enable="true"
```

```
      folder="c:\dbLog\"
   />
   [ … snip … ]
</myLittleBackup>
```

Config. #2: IIS and Storage on machine A, SQL Server on machine B



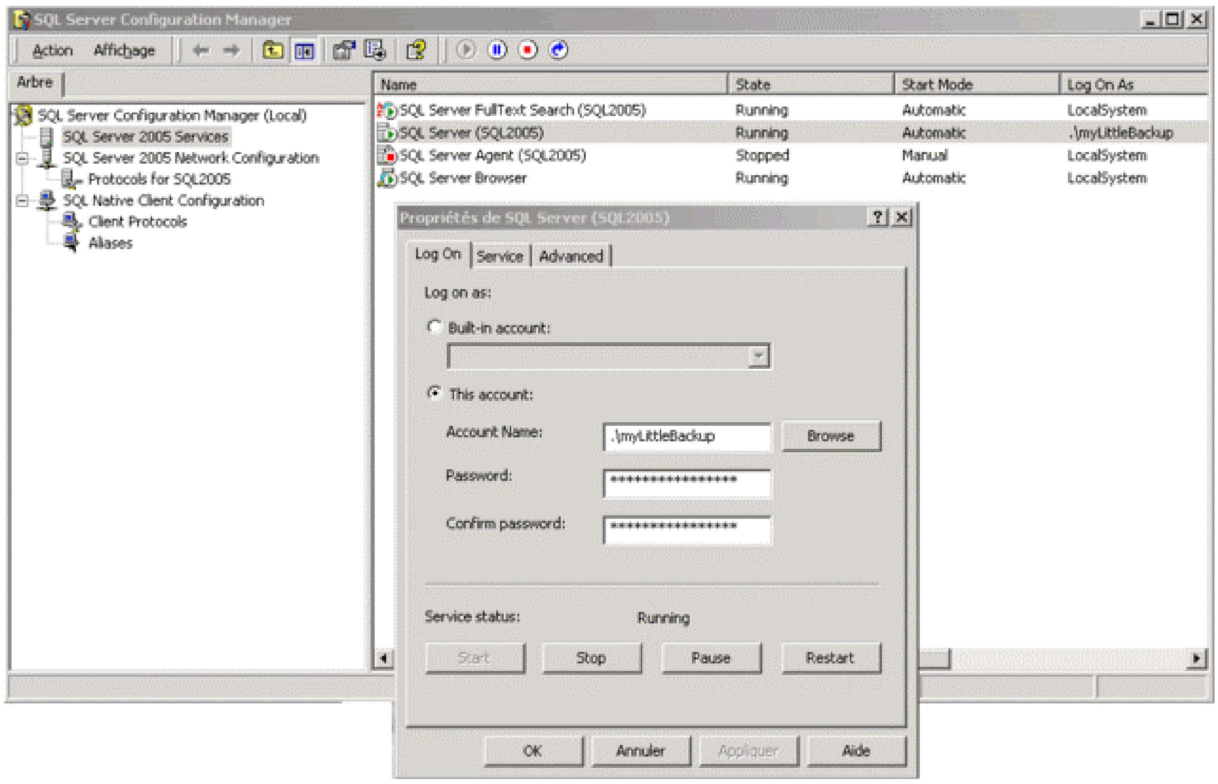
config. #2

- As you have 2 different boxes, **you need to use UNC** to define backup\defaultfolder and restore\defaultfolder attributes.
- As IIS and File Server are located on the same machine, you do not need to use impersonation.
- **You must give read/write permissions to ASPNET user for the backup and upload folders.**
- **You must allow SQL Server to access the backup and upload folders**.

SQL Server needs to access the file server. The default set of NT credentials used by MSSQLSERVER is the Localsystem account. The Localsystem account has no access to shares on the network as it isn't an authenticated network account. Therefore **SQL Server running under this account cannot backup/restore to/from a normal network share**. You will then need to use another account to run SQL Server.

Step 1. Create a new local user account on machine B (SQL Server)

Step 2. Launch your SQL Server Configuration manager and change the account used by SQL Server from the local system account to this newly created account. (see below)


SQL Server Configuration Manager
Action
Affichage
Arbre
SQL Server Configuration Manager (Local)
SQL Server 2005 Services
SQL Server 2005 Network Configuration
Protocols for SQL2005
SQL Native Client Configuration
Client Protocols
Aliases
Propriétés de SQL Server (SQL2005)
Log On
Service
Advanced
Browse
Start
Stop
Pause
Restart
OK
Annuler
Appliquer
Aide

Name	State	Start Mode	Log On As
SQL Server FullText Search (SQL2005)	Running	Automatic	LocalSystem
SQL Server (SQL2005)	Running	Automatic	.\myLittleBackup
SQL Server Agent (SQL2005)	Stopped	Manual	LocalSystem
SQL Server Browser	Running	Automatic	LocalSystem

Log on as:
Built-in account: [✓ This account:]
Account Name: .\myLittleBackup
Password: ****************
Confirm password: ****************
Service status: Running

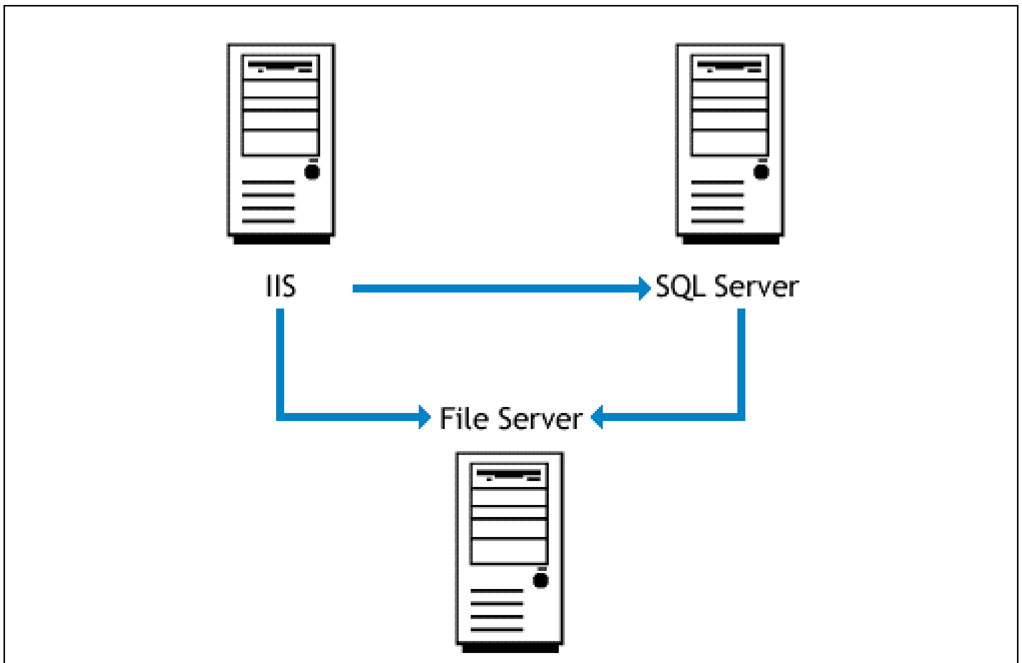
Step 3. Create a new local user account on machine A (IIS, File Server) with exactly the same name and password than the one created on machine B.
Step 4. Share the backup and upload folders on machine A.
Step 5. Give read/write permissions to this newly created user on the backup and upload folders.

Another solution consists to enable null session shares. Please follow the instructions from this Microsoft technote if you prefer this solution: http://support.microsoft.com/kb/289655/en-us.

Finally your config.xml file will look like

```
<myLittleBackup>
   <sqlservers>
      <sqlserver
         address="MachineB\InstanceName"
         name="SQL2005"
         sysadminconnection="true"
         sysadminconnectionid="1"
         />
   </sqlservers>
   <backup
      defaultfolder="\\MachineA\dbBackup\"
      maxbackupcount="1"
   >
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_backupoperator" />
         <authorization dbrole="db_datareader, db_datawriter" />
      </authorizations>
   </backup>
   <restore
      defaultfolder="\\MachineA\dbUpload\"
      maxuploadcount="1"
      checkmaxsize="true"
   >
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_datareader,db_datawriter,db_ddladmin" />
      </authorizations>
      <usermapping
         autofix="true"
         mappingmode="2"
      />
      <dbproperties>
         <compatibilitylevel default="90" />
         <isreadonly default="false" />
         <isautocloseon default="false" />
         <isautoshrinkon default="false" />
      </dbproperties>
   </restore>
   <check>
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_datareader, db_datawriter" />
      </authorizations>
   </check>
   <log
      enable="true"
      folder="c:\dbLog\"
   />
   [ … snip … ]
</myLittleBackup>
```

Config. #3: IIS on machine A, SQL Server on machine B, Storage on machine C



config. #3

This environment requires more configurations.

- As you have 3 different boxes, **you need to use UNC** to define backup\defaultfolder and restore\defaultfolder attributes.
- As IIS and File Server are located on 2 different machines, **you need to use impersonation**.
- **You must allow SQL Server to access the backup and upload folders**.

First of all, **we must check that IIS (located on machine A) will be able to access machine C and that the default asp.net user**

(ASPNET) will be able to read/write on the backup and upload folders. A lot of web-hosting companies have already configured IIS and their file servers for this behavior.

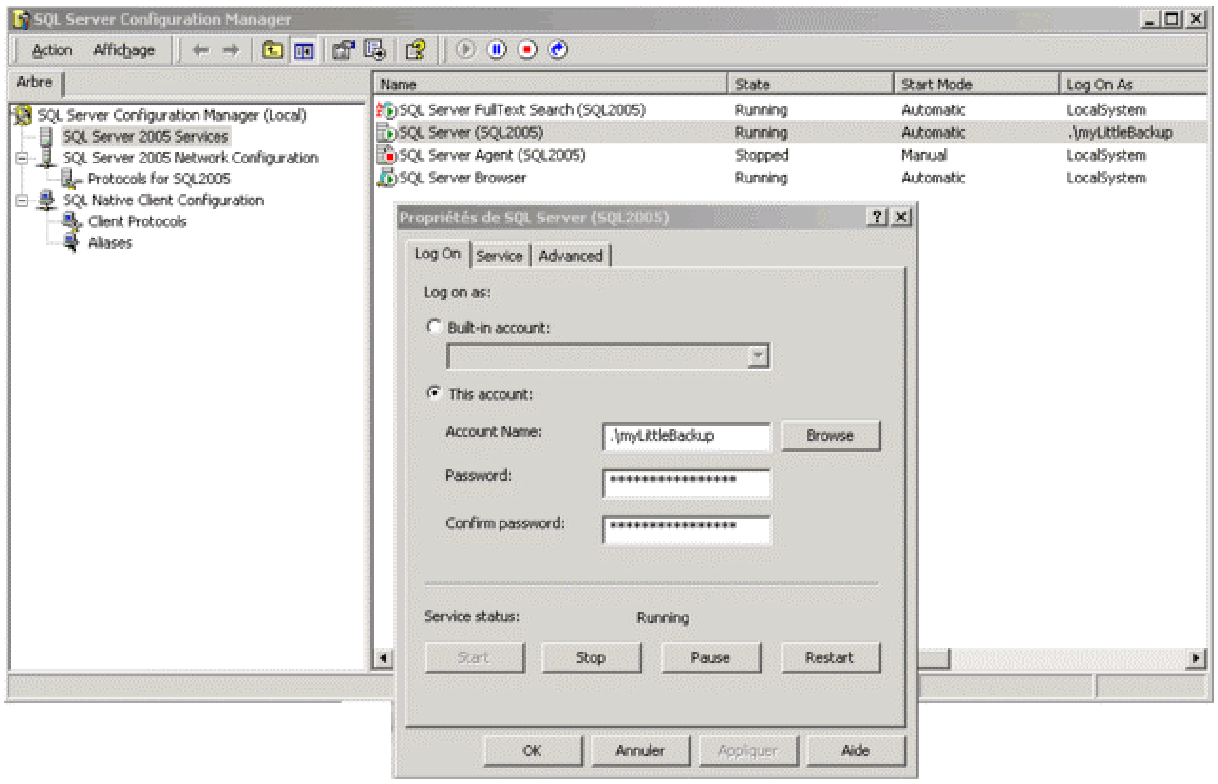
If you do not already have this behavior, then you need to use impersonation. You need to add this line to the web.config file

```
<identity impersonate="true" userName="UserName" password="Password" />
```

Step 1. Create a new local user account on machine A (IIS)
Step 2. Create a local user account on machine C (File server) with exactly the same name and password than the one created on machine A (IIS).
Step 3. If you are using Windows XP or Windows .NET Server 2003 you don't need do anything. If you have Windows 2000, go to Local Security Setting and browse to "User Rights assignment" and locate "Act as part of Operating System" policy. Double click and add the newly created account. You also need to set permission for impersonated user for full control on C:\winnt\Microsoft.NET\Framework\v2.0.50727\Temporary ASP.NET Files.
Step 4. Share the backup and upload folders on machine C.
Step 5. Give read/write permissions to this newly created user on the backup and upload folders.
Step 6. You may need to restart IIS services.

SQL Server needs to access the file server. The default set of NT credentials used by MSSQLSERVER is the Localsystem account. The Localsystem account has no access to shares on the network as it isn't an authenticated network account. Therefore **SQL Server running under this account cannot backup/restore to/from a normal network share**. You will then need to use another account to run SQL Server.

Step 1. Create a new local user account on machine B (SQL Server)
Step 2. Launch your SQL Server Configuration manager and change the account used by SQL Server from the local system account to this newly created account. (see below)


SQL Server Configuration Manager
Action Affichage
Arbre
SQL Server Configuration Manager (Local)
SQL Server 2005 Services
SQL Server 2005 Network Configuration
Protocols for SQL2005
SQL Native Client Configuration
Client Protocols
Aliases

Name	State	Start Mode	Log On As
SQL Server FullText Search (SQL2005)	Running	Automatic	LocalSystem
SQL Server (SQL2005)	Running	Automatic	.\myLittleBackup
SQL Server Agent (SQL2005)	Stopped	Manual	LocalSystem
SQL Server Browser	Running	Automatic	LocalSystem

Propriétés de SQL Server (SQL2005)

Log On | Service | Advanced

Log on as:

( ) Built-in account:

(•) This account:

Account Name: .\myLittleBackup

Password: *****************

Confirm password: *****************

Service status: Running

Step 3. Create a new local user account on machine C (File Server) with exactly the same name and password than the one created on machine B (SQL Server).

Step 4. Give read/write permissions to this newly created user on the backup and upload folders.

Another solution consists to enable null session shares. Please follow the instructions from this Microsoft technote if you prefer this solution: http://support.microsoft.com/kb/289655/en-us.

Finally your config.xml file will look like

```
<myLittleBackup>
   <sqlservers>
      <sqlserver
         address="MachineB\InstanceName"
         name="SQL2005"
         sysadminconnection="true"
         sysadminconnectionid="1"
         />
   </sqlservers>
```

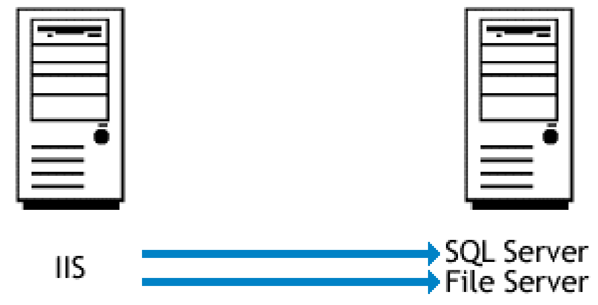
```
<backup
   backupfolder="\\MachineC\dbBackup\"
   maxbackupcount="1"
>
   <authorizations>
      <authorization dbrole="db_owner" />
      <authorization dbrole="db_backupoperator" />
      <authorization dbrole="db_datareader, db_datawriter" />
   </authorizations>
</backup>

<restore
   uploadfolder="\\MachineC\dbUpload\"
   maxuploadcount="1"
   checkmaxsize="true"
>
   <authorizations>
      <authorization dbrole="db_owner" />
      <authorization dbrole="db_datareader,db_datawriter,db_ddladmin" />
   </authorizations>
   <usermapping
      autofix="true"
      mappingmode="2"
   />
   <dbproperties>
      <compatibilitylevel default="90" />
      <isreadonly default="false" />
      <isautocloseon default="false" />
      <isautoshrinkon default="false" />
   </dbproperties>
</restore>
<check>
   <authorizations>
      <authorization dbrole="db_owner" />
      <authorization dbrole="db_datareader, db_datawriter" />
   </authorizations>
</check>
<log
   enable="true"
   folder="c:\dbLog\"
/>
```

```
   [ … snip … ]
</myLittleBackup>
```

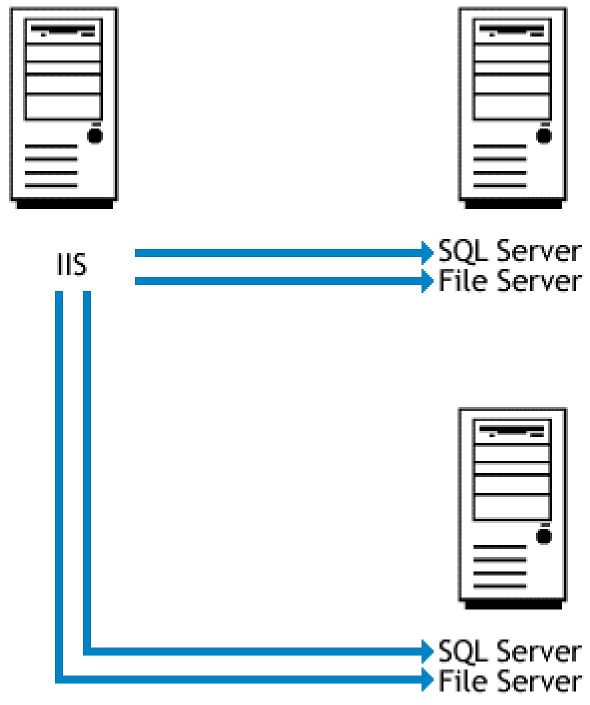
Config. #4: IIS on machine A, SQL Server and Storage on machine B

This is the configuration we recommend.



config. #4

Configuration will behave the same when you use several SQL Servers.



config. #4bis

- ➡ As you have 2 different boxes, **you need to use UNC** to define backup\defaultfolder and restore\defaultfolder attributes.
- ➡ As IIS and File Server are located on 2 different machines, **you need to use impersonation.**
- ➡ SQL Server and File Server are on the same machine, thus your SQL Server services can log on as Local System Account. There is no need to enable Null Session Share.

First of all, **we must check that IIS (located on machine A) will be able to access machine B and that the default asp.net user (ASPNET) will be able to read/write on the backup and upload folders**. A lot of web-hosting companies have already configured IIS and their file servers for this behavior.

If you do not already have this behavior, then you need to use impersonation. You need to add this line to the web.config file

```
<identity impersonate="true" userName="UserName" password="Password" />
```

Step 1. Create a new local user account on machine A (IIS)
Step 2. Create a local user account on machine B (File server) with exactly the same name and password than the one created on machine A (IIS).
Step 3. If you are using Windows XP or Windows .NET Server 2003 you don’t need do anything. If you have Windows 2000, go to Local Security Setting and browse to “User Rights assignment” and locate “Act as part of Operating System” policy. Double click and add the newly created account. You also need to set permission for impersonated user for full control on C:\winnt\Microsoft.NET\Framework\v2.0.50727\Temporary ASP.NET Files.
Step 4. Share the backup and upload folders on machine B.
Step 5. Give read/write permissions to this newly created user on the backup and upload folders.
Step 6. You may need to restart IIS services.

Of course you need to repeat this operation for each file servers (config. #4bis).

As we can have several SQL Servers/File Servers, we will define backupfolder and uploadfolder attributes for each server using UNC.

Finally your config.xml file will look like

```
<myLittleBackup>
   <sqlservers>
      <sqlserver
         address="MachineB1\InstanceName"
         name="SQL2005"
         sysadminconnection="true"
         sysadminconnectionid="1"
         backupfolder="\\MachineB1\dbBackup"
         uploadfolder="\\MachineB1\dbUpload"
      />
      <sqlserver
         address="MachineB2\InstanceName"
         name="SQL2000"
         sysadminconnection="true"
         sysadminconnectionid="2"
         backupfolder="\\MachineB2\dbBackup"
         uploadfolder="\\MachineB2\dbUpload"
         compatibilitylevel="80"
         />
   </sqlservers>
   <backup
      maxbackupcount="1"
   >
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_backupoperator" />
         <authorization dbrole="db_datareader, db_datawriter" />
      </authorizations>
   </backup>

   <restore
      maxuploadcount="1"
      checkmaxsize="true"
   >
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_datareader,db_datawriter,db_ddladmin" />
      </authorizations>
      <usermapping
         autofix="true"
         mappingmode="2"
      />
      <dbproperties>
         <compatibilitylevel default="90" />
         <isreadonly default="false" />
         <isautocloseon default="false" />
         <isautoshrinkon default="false" />
      </dbproperties>
   </restore>
   <check>
      <authorizations>
         <authorization dbrole="db_owner" />
         <authorization dbrole="db_datareader, db_datawriter" />
      </authorizations>
   </check>
   <log
      enable="true"
      folder="c:\dbLog\"
   />
   [ … snip … ]
</myLittleBackup>
```

ADDING A NEW INTERFACE LANGUAGE

myLittleBackup for SQL Server 2000 and 2005 can support multi-language. Please follow the instructions below to localize myLittleBackup in your own language.

1. Edit the config.xml file located at the root folder of myLittleAdmin. Add a new language node to the <languages> node with your language culture and name. For instance:

```
<languages>
   <language culture="en-US" name="English" />
   <language culture="fr-FR" name="French" />
   <language culture="ja-JP" name="Japanese" />
   ...
</languages>
```

2. Create a new folder in the **/bin** folder of myLittleAdmin. Name it with the culture name of your language. For instance, if you want to add a Japanese translation, then you create a new **ja-JP** folder.
3. Copy files **strings.resources** and **compileRes.bat** located in the **/bin** folder of myLittleAdmin in your newly created folder, and rename **strings.resources** file into **strings.RessourceName.resources**.
   For instance : **strings.ja-JP.resources**
4. Edit this resource file with a resource editor application. You can download and use the free Resourcer for .NET by Lutz Roeder: http://www.aisto.com/roeder/DotNet/
5. Now you must translate all the entries from your resource file in your own language.
6. At last, you just need to compile the resource file in a DLL. To do that, you need the **Assembly Linker** application to be installed on your computer. You can find this application in the **Framework SDK** (you can download it here: http://msdn.microsoft.com/netframework/downloads/framework1_1/)
7. Edit the **compileRes.bat** file and change the line to (this sample is for ja-JP culture)

```
C:\WINNT\Microsoft.NET\Framework\v1.1.4322\al.exe
/embed:strings.ja-JP.resources,strings.ja-JP.resources
/out:strings.resources.dll /c:ja-JP
```

8. Launch the **compileRes.bat** batch file by double-clicking on it. You should now have a **strings.resources.dll** located in your language folder.
9. Launch myLittleBackup and choose your language in the **Preferences** section

ENABLING/DISABLING FEATURES

Each feature of myLittleBackup can be enabled/disabled. Please follow the instructions below:

1. Feature settings are located in the **xml/profile/default.xml** file. Edit the **xml/profile/default.xml**.
2. Set to **true** or **false** the feature you want to enable or disable.
3. Relaunch myLittleBackup

```
<profile id="default">
   <tools>
      <backup display="true" />
      <restore display="true" />
      <check display="true" />
      <manage display="true" />
   </tools>
</profile>
```

Note: If you also want to remove a link to a tool from the left navigation sidebar, then you need to edit file **xml/navbardata.xml** and delete the specified node.