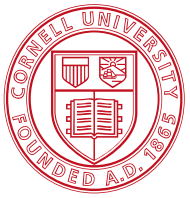


Cornell University
Center for Advanced Computing

MATLAB on the TeraGrid Technical Documentation: How to Build a Parallel MATLAB Cluster and that's not all!

February 2010

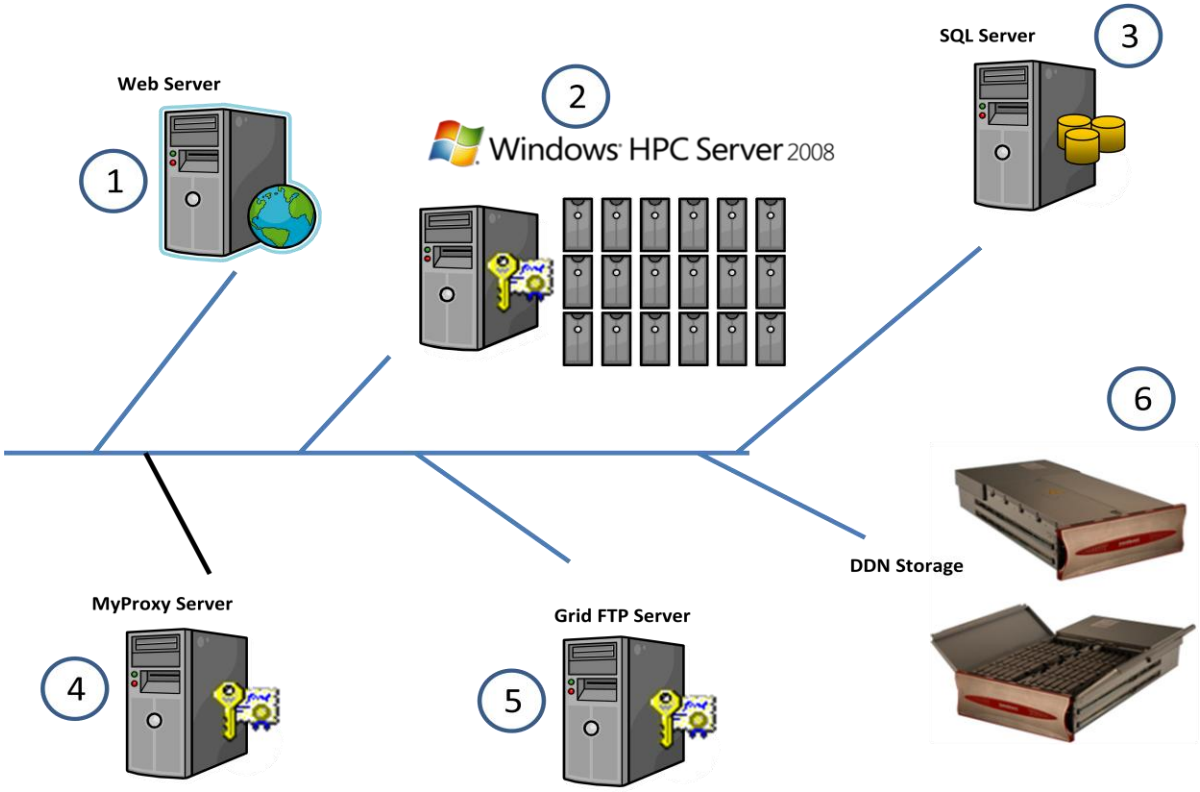
Brenda Lapp, Cornell Center for Advanced Computing – lapp@cac.cornell.edu
Steven Lee, Cornell Center for Advanced Computing – shl1@cac.cornell.edu
Lucia Walle, Cornell Center for Advanced Computing – lwalle@cac.cornell.edu

Abstract

This document provides information about how the Cornell Center for Advanced Computing is running an experimental resource on the TeraGrid. This utility will provide parallel computation and analysis services to interactive desktop users and Science Gateways.

Contents

Architecture of MATLAB on the TeraGrid



We are using Active Directory for single sign on for our environment. All domain users sign on with an account created specifically for them - be it to a Windows server or a Linux server.

1) Web Server:
   a) Current usage and statistics of cluster
   b) Web Service for registering password
   c) Internal Web Page for user list - for Grid FTP
2) Windows HPC Server 2008 Head Node:
   a) Head node (Dell PowerEdge 1950) and compute nodes (four Dell PowerEdge M1000e chassis with 16 PowerEdge M600 blades)
   b) License Server for MATLAB compute nodes
   c) Scripts for copying data to SQL Server about user jobs
   d) Running modified HPC Basic Profile service
3) SQL Server
   a) Lookup Table for users (used from Web Server and Head node)
   b) Data stored from Windows HPC Server 2008 SQL Express Database
4) MyProxy Server
   a) This can be either inside the CAC domain or a trusted server outside of Cornell
5) Grid FTP Server - For moving data, used in simulations
6) DataDirect Network Storage -8 TB of dedicated storage for "MATLAB on the TeraGrid"

Web Server

The web server is running Microsoft Windows Server 2003 R2 Enterprise edition with Service Pack 2 and IIS 6.0. This server could be run on another OS or a more current OS from Microsoft. We are utilizing what we have in place so as not to add more cost to our infrastructure. The current usage and usage for the last 30 days can be found here: http://www.cac.cornell.edu/matlab/status/JobStats.aspx. A web service to register an X.509 certificate with a domain user account is also found on this server. As well as a web page displaying the domain user id with the certificate registered.

Building the Head Node

We used Windows Deployment Services (WDS) to install the base operating system (our unattend file is very simple and sets the time zone, license string, etc.) and then installed Windows features. A screenshot of the installed features appears in Appendix 1. We then configured the head node with additional software (ActivePerl, Dell OpenManage) and then installed the HPC Pack Installation Disc to create the head node. You could choose to use the Windows HPC Server 2008 Installation Disc to install the OS.

Installing MATLAB on the Head Node

Choose the appropriate wizard (for the current or a previous version) at http://www.mathworks.com/support/product/DM/installation/ver_current/ to generate documents which will assist with the MATLAB installation. There will be at least four documents; two are applicable to installing MATLAB on the head and cluster nodes (also referred to as worker nodes) and the remaining are applicable to the client installation (the number of files is determined by what client operating systems you selected in the wizard). Here are a few notes made during our installation of MATLAB Distributed Computing Server 4.1 (R2009a) on Windows HPC Server 2008 *(the documentation was written for Windows Compute Cluster Server 2003 (CCS) not Windows HPC Server 2008)*:

- Start the installation from an administrative command prompt.
- Unpack the installer in the same folder as the toolboxes or the toolboxes will not be available during installation.
- Stage 1, Step 2, Item 1 – if not using an installation DVD, map the drive to the folder that has the setup.exe; the installation will not work if a UNC path is specified.
- Stage 1, Step 2, Item 4 – select Custom installation otherwise you will get a message about MATLAB needing to be installed.
- Stage 1, Step 2, Item 5 – after clicking next, there will be a list of options on the following screen. if your license manager runs on a different server, verify everything except license is selected; if you decide the license manager needs be installed on the head node, you can add it later *[we ran into some license check-out issues with our existing license manager that were resolved by installing the license manager for MATLAB on the head node].*
    - We recently had the opportunity to build another small cluster. Some of the following comments may be obvious to those that who have installed the MATLAB license manager before, but for others, it may save time:

    - All licenses have to be generated for the physical address of the head node (even the distributed computing engine for the worker nodes).
    - Create your license.dat file as indicated in Stage 1, Step 1, including the lines that begin with "#." You may also need to include any shorter license "strings" for the MATLAB Distributed Computing Engine licenses you got for the worker nodes in the same file. It does not matter if they are added at the top of the bottom of the license.dat file, but include the lines that begin with "#."
- Stage 1, Step 3 and later have to wait until a compute node is built.

Installing and Configuring HPC Pack

Use a Microsoft HPC Pack 2008 installation CD to create the head node. Choose "Create a new HPC cluster by creating a head node" for the installation type. A wizard will start to walk you through the basic configuration.

The Network Topology is the first choice you'll have to make. The CAC head node is on both public (Enterprise) and private (Private) networks so we selected "Compute nodes isolated on a private network". We also turned off the firewall. The compute nodes are in 10-space (Private). We found that the head node had to be physically located on a different subnet from the compute notes, i.e., if the compute nodes were located at 10.84.3.x, the head node could not communicate with the compute nodes if it was configured at 128.84.3.x. This may change with future releases of Windows HPC Server.

Failover Clustering and Backing up the Head Node

If you decide that you need failover clustering, please see http://technet.microsoft.com/en-us/library/cc719001(WS.10).aspx for a step-by-step guide. We have not implemented failover clustering; the head node cannot serve as a Windows Communications Foundation (WCF) broker node which ours does for another purpose. To backup the head node, use the guidelines at http://technet.microsoft.com/en-us/library/ee247401(WS.10).aspx.

Resources that we found helpful include:

Windows HPC Home Page	http://www.microsoft.com/hpc/en/us/default.aspx
Windows HPC Server 2008	http://technet.microsoft.com/en-us/library/cc510343(WS.10).aspx
Windows Deployment Services Getting Started Guide	http://technet.microsoft.com/en-us/library/cc771670(WS.10).aspx
Overview and Requirements for Windows HPC Server 2008 in a Failover Cluster	http://technet.microsoft.com/en-us/library/cc719013(WS.10).aspx
Customizing Cluster Setup Instructions: Installing Version 4.1 (R2009a)	http://www.mathworks.com/support/product/DM/installation/ver_current/setupwiz.html

Building Compute Nodes

We used Windows Deployment Services (WDS) to install the base Windows Server HPC 2008 Edition operating system on one compute node. Next we installed Windows features (a screenshot of the installed features appears in Appendix 2). Then we installed the software needed (ActivePerl, Dell OpenManage, Microsoft HPC Pack, MATLAB).

Installing MATLAB on the Compute Nodes

- Start the installation from an administrative command prompt .
- Unpack the installer in the same folder as the toolboxes or the toolboxes will not be available during installation.
- Stage 1, Step 2, Item 1 – if not using an installation DVD, map the drive to the folder where setup.exe resides; the installation will not work with a UNC path.
- Stage 1, Step 2, Item 4 – select Custom installation otherwise you will get a message about MATLAB needing to be installed.
- Stage 1, Step 2, Item 5 – after clicking next, there will be a list of options on the following screen. If your license manager runs on a different server verify everything except license is selected. If you decide the license manager needs be installed on the head node, you can add it later *[we ran into some license check-out issues with our existing license manager that were resolved by installing the license manager for MATLAB on the head node].*
- Stage 1, Step 3 must be done from an administrative command prompt.
    - If output.txt reads "License Manager Error -18," it is likely that either the physical address for the MATLAB Distributed Computing Engine was not generated for the head node or that they were not appended to the license.dat file. If the physical address does not match that of the head node, The Mathworks will need to generate new licenses. If the information needs to be appended to license.dat, append it and then go to Stage 1, Step 3 and stop the license manager. Reread the license file and then restart the license manager.
- Stage 2, some of the steps must be done from an administrative command prompt.
- Stage 2, Item 3 – skipped; we are not using a Windows firewall (or any other firewall).
- Stage 2, Item 4a – skipped; MDCS is not on a network share.
- Stage 2, Item 5 – you only have to complete this step on the node you take an image from.

Even though you will be able to view the compute node from the Cluster Manager, do not apply a node template yet. The node templates will be applied after imaging the compute nodes.

Add m files and if using .MEX files install VS C++ Express and SDK before taking an image.

Sysprep Compute Node to Take Image

Before taking an image of the compute node you just built, sysprep the compute node. We used information in Brian Desmond's and Elan Shudnow's blogs to create our unattend file. We also created a

command file to delete the unattend file on completion and issued the sysprep command as described in Brian Desmond's blog.

Image Node (Take Image of sysprepped Compute Node)

We used Windows Deployment Services (WDS) to take an image of the compute node we just sysprepped.

When the image is taken, open Windows Deployment Services and find the image file (*.wim). Right-click and choose Properties. Check the box to "allow image to install in unattended mode" and navigate to your unattend.xml file and click the "Select File" button. If you miss this step, some of the problems you may encounter are the wrong time zone and you will need to enter your product key again to activate.

Resources we used:

Windows HPC Home Page	http://www.microsoft.com/hpc/en/us/default.aspx
Windows HPC Server 2008	http://technet.microsoft.com/en-us/library/cc510343(WS.10).aspx
Windows Deployment Services	http://technet.microsoft.com/en-us/library/cc772106(WS.10).aspx
Windows Deployment Services: Getting Started Guide	http://technet.microsoft.com/en-us/library/cc771670(WS.10).aspx
How to Sysprep in Windows 2008	http://briandesmond.com/blog/how-to-sysprep-in-windows-2008/
Unattended Server 2008 Base Image Creating using WSIM/Sysprep	http://www.shudnow.net/2008/05/05/unattended-server-2008-base-image-creation-using-wsimsysprep/
How Prestaged Devices Work	http://technet.microsoft.com/en-us/library/cc772219.aspx
Prestaging Client Computers	http://technet.microsoft.com/en-us/library/cc770832(WS.10).aspx
Deploying Vista – Part 18: Managing Windows Deployment Services *(used this document for provisioning the nodes in Active Directory)*	http://www.windowsnetworking.com/articles_tutorials/Deploying-Vista-Part18.html

Image Nodes (Push to Other Compute Nodes)

Provision the compute nodes in Active Directory using WDSUTIL. If necessary, add DHCP reservations for the nodes.

PXE-boot the remaining compute nodes and push the image to them using WDS. You could choose to use other methods to image the compute nodes including the imaging tools built into HPC.

Resources we used:

How Prestaged Devices Work	http://technet.microsoft.com/en-us/library/cc772219.aspx
Prestaging Client Computers	http://technet.microsoft.com/en-us/library/cc770832(WS.10).aspx
Deploying Vista – Part 18: Managing Windows Deployment Services *(used this document for provisioning the nodes in Active Directory)*	http://www.windowsnetworking.com/articles_tutorials/Deploying-Vista-Part18.html
Use Netsh to Insert a List of Reservations in a DHCP Scope from an Input File	http://cwashington.netreach.net/depo/view.asp?Index=983
Creating DHCP reservations at the command line	http://www.networkplumbing.org/2008/07/creating-dhcp-reservations-at-command.html

Post Image Tasks

- Partition the reminder of the hard disk. We typically assign the letter T to the partition and name it temp.
- Set the size and move the pagefile to the T partition.
- All of the nodes should be visible from Node Management on HPC Cluster Manager. The node state should be unknown and the node template column blank.
    - If a compute node comes up with an incorrect name, **note the name as shown, and then** right-click on the node in Cluster Manager. Choose edit and change the computer name (this has to be done before assigning a node template). Go to the compute node and rename the computer there. The renaming order does not matter.
    - Once all of the nodes have their correct name, apply the Node Template you plan to use.
- Run diagnostics – the diagnostic tests we typically run are: All Services Running, DNS Name Resolution, Domain Connectivity, Internode Connectivity, Job Submission Test, MPI Ping-Pong: Quick Check and MPI Ping-Pong, and Lightweight Throughput. Note, if your diagnostics fail with an error similar to "unauthorized user," go to the Configuration tab in HPC Cluster Manager, click on "Provide installation credentials," and input your credentials. This should resolve your problem.

Resources we used:

DiskPart Command-Line Options	http://technet.microsoft.com/en-us/library/cc766465(WS.10).aspx
A Description of the Diskpart Command-Line Utility	http://support.microsoft.com/kb/300415
Server Core Tips and Tricks Vol. 4	http://blogs.technet.com/server_core/archive/2006/12/18/tips-and-tricks-vol-4.aspx
Server Core - Move that Swap/Page File – WMI Error	http://blog.mpecsinc.ca/2008/04/server-core-move-that-swappage-file-wmi.html

Code - 0x80041017	

Setting up the HPC Basic Profile Service with X.509 Certificates

On the Windows HPC Server 2008 Developer Resources web page there are five HPC Basic Profile Web Service videos. These videos describe what is available to the developers and also provide some insight on setup and operation. You also will need the Microsoft HPC Pack 2008 SDK for the Sample code. We used the projects HPCBPServiceV1 and HPCBPSerivceV2 for proof of concept.

Initially, we set up the Basic Profile Service to use WS-Security User name Token by following the instructions from "HPC Server Basic Profile Web Service Configurations Guide." Get a server certificate for SSL communication, since you are sending username and passwords. There are links provided for help on this if needed. The HPC Basic Profile service comes with the HPC distribution. You do, however, have to run the HPC Power Shell as an administrator, then execute the script hpcbpws.ps1.

On our first attempt, we get the following message:

PS C:\Program Files\Microsoft HPC Pack\Bin> hpcbpws.ps1 install
File C:\Program Files\Microsoft HPC Pack\Bin\hpcbpws.ps1 cannot be loaded because the execution of scripts is disabled on this system. Please see "get-help about_signing" for more details.
At line:1 char:12+ hpcbpws.ps1 <<<< install

Then we issued the following commands:

- Get-Help about_signing
- Get-ExecutionPolicy
- Set-ExecutionPolicy

For more information on PowerShell scripts, visit the web page "Running Windows PowerShell Scripts" listed in the Resources Section.

After completing the steps above, the HPCBasicProfile Web Service worked with Username and Password authentication.

To change this to work with x.509 certificates, follow these steps. Code modifications are necessary for multiple files in the HPCBPServiceV1 sample project. For testing purposes the C# client modifications are included as well. Five files need to be modified to get the Service and the Client talking using x.509 certificates. These five files are in three of the projects HPCBasicProfileCCS, HPCBasicProfileClient, HPCBasicProfileHost. The five files are: HPCBasicProfileService.cs, HPCBasicProfileHost.exe.config, MainModule.cs, HPCBasicProfileClient, and HPCBasicProfile.xml. The code changes are listed in Appendix 3. We also created a SQL Lookup Table to match the DN from the x.509 certificate to the user account and associated password of the user. We have a web front end to gather the data and put it into the database. The HPCBasicProfile Service queries the lookup table with the DN from the client certificate. If the certificate is valid and we get a match from the lookup table, the job goes through.

Once all the code changes are in place, we log into the head node and set the SSL connection to enable client certificates. The following is an example scenario. When we enter the "netsh http show sslcert" command, we see that Negotiate Client Certificate property is disabled. Basically, we delete this certificate and create one that has the Negotiate Client Certificate enabled. The "Certificate Hash" is important to keep the same.

```
C:\Program Files\Microsoft HPC Pack\Bin>netsh http show sslcert

SSL Certificate bindings:
-------------------------
    IP:port                 : 0.0.0.0:443
    Certificate Hash        : 8f846e55b9123e9a794025149f6dd0aca49729f2
    Application ID          : {4dc3e181-e14b-4a21-b022-59fc669b0914}
    Certificate Store Name  : MY
    Verify Client Certificate Revocation    : Enabled
    Verify Revocation Using Cached Client Certificate Only    : Disabled
    Usage Check    : Enabled
    Revocation Freshness Time : 0
    URL Retrieval Timeout   : 0
    Ctl Identifier          : (null)
    Ctl Store Name          : (null)
    DS Mapper Usage    : Disabled
    Negotiate Client Certificate    : Disabled

C:\Program Files\Microsoft HPC Pack\Bin>netsh http delete sslcert ipport=0.0.0.0:443

SSL Certificate successfully deleted


C:\Program Files\Microsoft HPC Pack\Bin>netsh http add sslcert ipport=0.0.0.0:443
certhash=8f846e55b9123e9a794025149f6dd0aca49729f2 appid={00112233-4455-6677-8899-
AABBCCDDEEFF} clientcertnegotiation=enable

SSL Certificate successfully added


C:\Program Files\Microsoft HPC Pack\Bin>netsh http show sslcert

SSL Certificate bindings:
-------------------------

    IP:port                 : 0.0.0.0:443
    Certificate Hash        : 8f846e55b9123e9a794025149f6dd0aca49729f2
    Application ID          : {00112233-4455-6677-8899-aabbccddeeff}
    Certificate Store Name  : (null)
    Verify Client Certificate Revocation    : Enabled
    Verify Revocation Using Cached Client Certificate Only    : Disabled
    Usage Check    : Enabled
    Revocation Freshness Time : 0
    URL Retrieval Timeout   : 0
    Ctl Identifier          : (null)
    Ctl Store Name          : (null)
    DS Mapper Usage    : Disabled
    Negotiate Client Certificate    : Enabled
```

This completes the Head Node server side of the setup.

Resources:

Windows HPC Server 2008 Developer Resources	http://www.microsoft.com/hpc/en/us/developer-resources.aspx
HPC Server Basic Profile Web Service Operations Guide	http://technet.microsoft.com/en-us/library/cc972837(WS.10).aspx
Creating Clients that Connect to the HPC Basic Profile Web Service (download document)	http://go.microsoft.com/fwlink/?LinkId=128113
Netsh command for HTTP	http://msdn.microsoft.com/en-us/library/cc307236(VS.85).aspx
Running Windows PowerShell Scripts	http://www.microsoft.com/technet/scriptcenter/topics/winpsh/manual/run.mspx

SQL Server

We are currently running Microsoft Windows Server 2003 Enterprise Edition with SP2, running Microsoft SQL Server 2005 with the latest service pack as well. The purpose of this server is to hold the lookup table for matching a domain user with an X.509 certificate. Again, since we already run SQL Server we chose to utilize one that is already built and in our domain, a different mechanism could replace this. We wrote two stored procedures one to enter the data into the table, run as a user with only those permissions. The second stored procedure gets the data and is called from the HPC Basic Profile Service to get the users credentials as needed to run the users job, this is set up to run as a separate user with only the permissions needed to do this task. The domain user name to certificate DN mapping is made available via a web page for the gridftp server to generate the grid-mapfile it needs.

We also store job usage data on this server so we can track usage by user. A powershell script runs on the Head Node and puts the data from the SQL Express Db used by Windows HPC Server 2008, into a db here.

MyProxy Server

MyProxy server is only needed to allow users without TG certificates to use the system. If all users already have certificates issued by a trusted CA, MyProxy server is **not** required.

MyProxy server runs MyProxy server compiled from globus 4.2.1. It is configured to run as a certificate authority (CA). When a user logs on using “myproxy-logon” command from a client with a user name and password, myproxy server:

- authenticates against the Windows Active Directory via Kerberos, and
- obtains user information from Active Directory via LDAP to generate a user certificate valid for 12 hours.

The relevant parameters in myproxy-server.config are:

```
pam "sufficient"
ca_ldap_server "ldap://<FQDN of Active Directory server>"
ca_ldap_uid_attribute "sAMAccountName"
ca_ldap_searchbase "<base>"
ca_ldap_connect_dn "<user name for LDAP binding"
ca_ldap_connect_passphrase "<password>"
```

Please note: ca_ldap_connect_dn and ca_ldap_connect_passphrase are only needed if your Active Directory requires them for binding. And there is a bug in myproxy-server code in globus-4.2.1 that truncates the last character of the passphrase that causes the binding to fail.

GridFTP Server

The gridftp server mounts the 8 TB storage via NFS and makes the mounted file system accessible via gridftp. User certificate-to-uid mapping is provided by /etc/grid-security/grid-mapfile. A script updates the grid-mapfile using the information pulled from the web server/SQL database.

DDN Storage

The DDN storage is attached to a Dell file server via IB. The file server serves the file system to the Windows HPC cluster via CIFS and the GridFTP server via NFS.

Alternatively, you can directly attach the storage to the GridFTP server and run samba on the GridFTP server for better performance. We did not go this route to save the cost of additional IB infrastructure.

Reporting

You can increase the number of days the job history is displayed in HPC Cluster Manager by going to the Options menu, choosing Job Scheduler Configuration, and selecting the Job History tab.

If you need to report on cluster usage, click on Charts and Reports in HPC Cluster Manager (the only document we found on reporting was prerelease documentation which can no longer be found on-line; we were unable to find additional documentation).

Monitoring and Troubleshooting

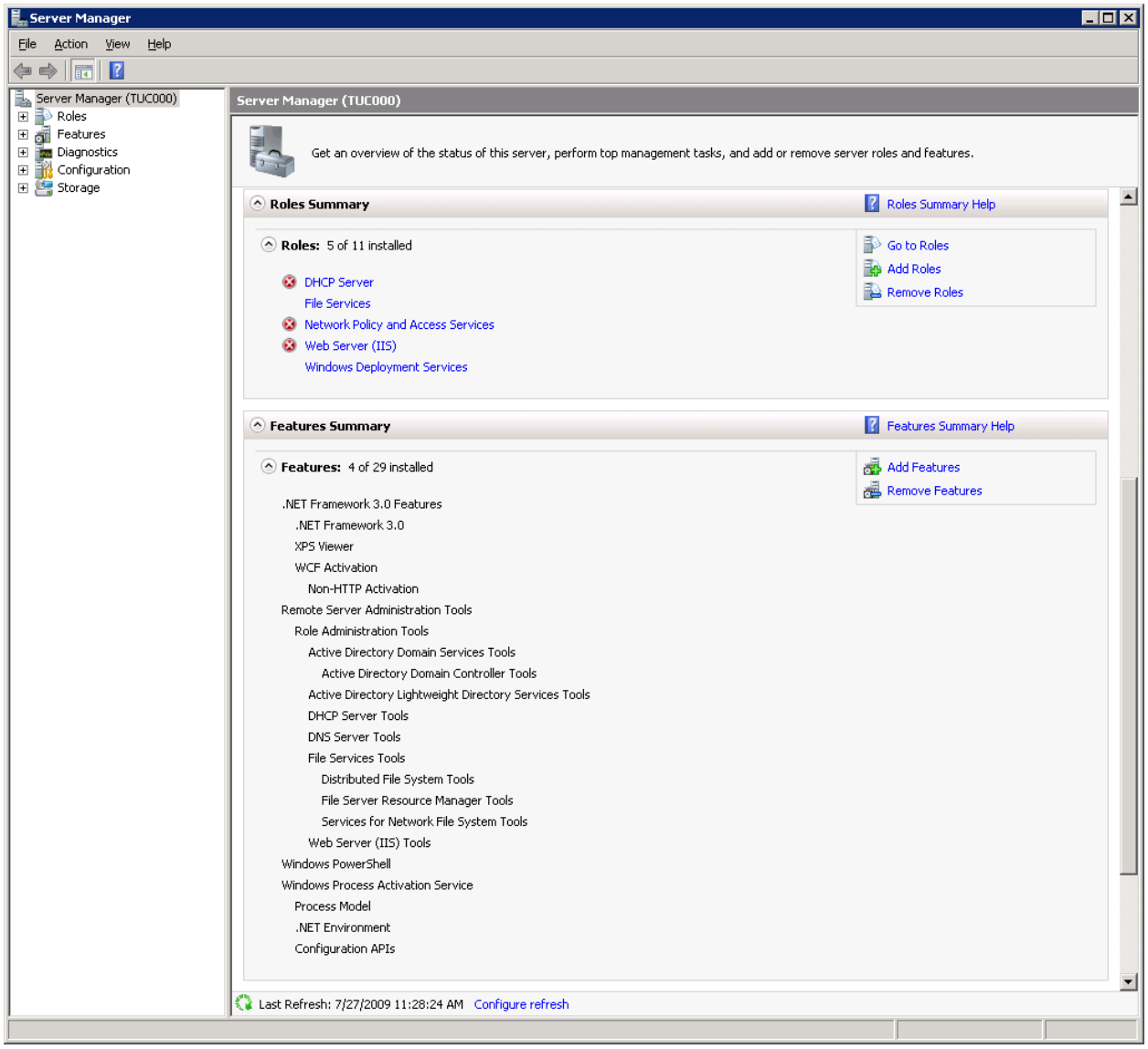
- Open Cluster Manager and look for irregularities; offline nodes, failed jobs, etc., and investigate to correct potential problems.
- If on the heat map, all nodes appear grey and running jobs and cores in use are zero, see http://technet.microsoft.com/en-us/library/dd729211(WS.10).aspx.
- If your screen is blank when you logon to any of the nodes, see http://support.microsoft.com/kb/970879.
- In addition to the documentation, see http://social.microsoft.com/Forums/en-US/category/windowshpc for issues others using HPC Server 2008 have encountered.

- Resources we used:

Windows HPC Server 2008 Command Reference	http://technet.microsoft.com/en-us/library/cc972841(WS.10).aspx
PowerShell Cmdlet Equivalents for HPC Command-Line Commands	http://technet.microsoft.com/en-us/library/dd540131(WS.10).aspx

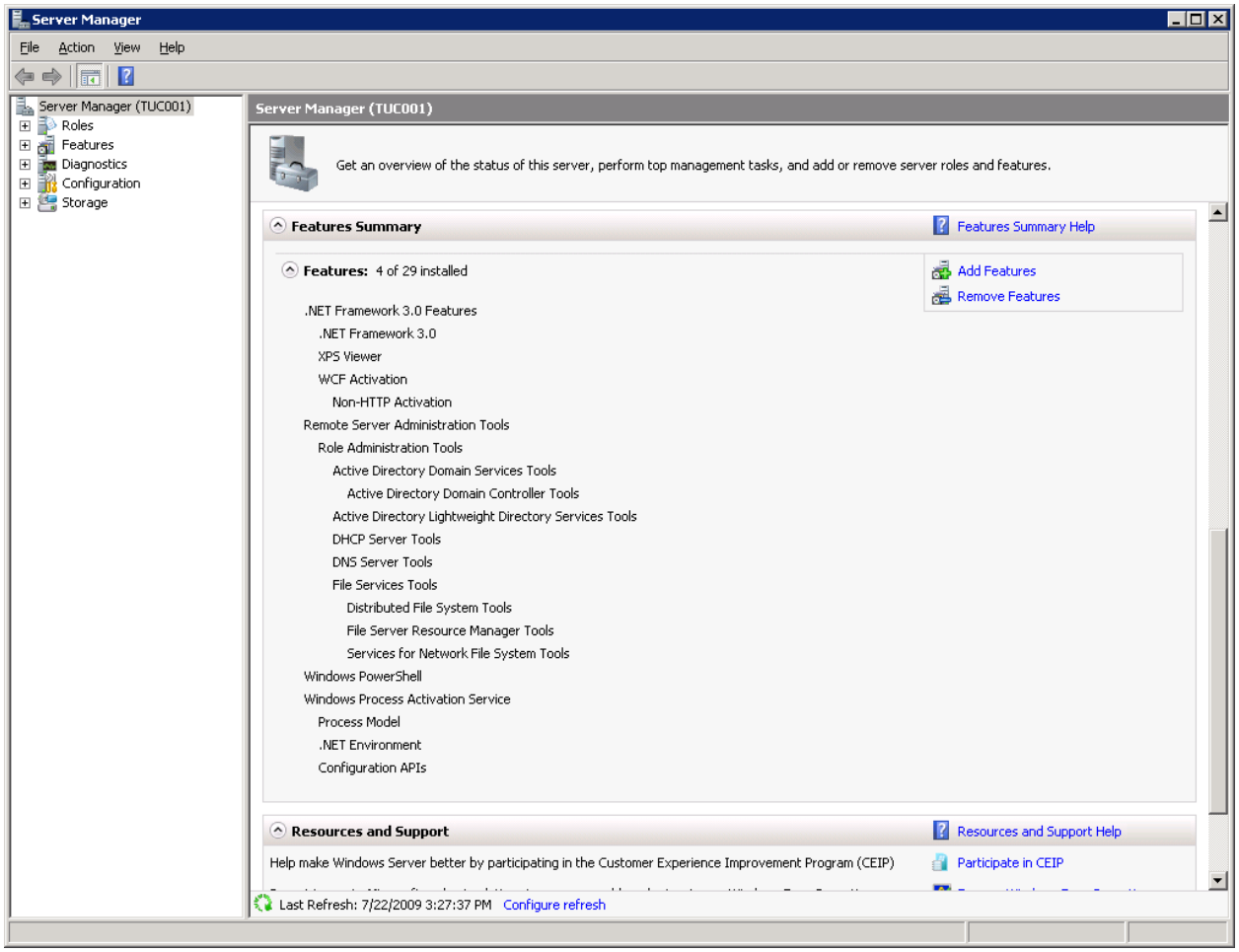
Appendix 1

Roles and Features on head node (all features were added with installation of HPC Pack).



Appendix 2

Features on compute nodes.



Appendix 3

Code Changes for Server Side HPCBPServiceV1

In the Project HPCBasicProfileCCS the File HPCBasicProfileService.cs

Comment out:

```
line 117     [OperationBehavior(Impersonation = ImpersonationOption.Required)]
line 150     if (!this.GetWSUserCredentials(ref userName, ref password))
line 151     {
line 152     NotAuthorizedFaultHelper.Throw(LocalizedStrings.CannotRetrieveUserName);
line 153     }
line 158     if (!String.IsNullOrEmpty(jsdlUserName) && (0 !=
             String.Compare(userName, jsdlUserName,
             StringComparison.InvariantCultureIgnoreCase)))
```

```
line 159    {
line 160    NotAuthorizedFaultHelper.Throw(LocalizedStrings.InvalidUserName);
line 161    }
line 201    [OperationBehavior(Impersonation = ImpersonationOption.Required)]
line 295    [OperationBehavior(Impersonation = ImpersonationOption.Required)]
line 431    [OperationBehavior(Impersonation = ImpersonationOption.Required)]
Line 524    [OperationBehavior(Impersonation = ImpersonationOption.Required)]
```

Insert the following code lines:

```
line 29      using System.Data.SqlClient;
line 30      using System.Data;
line 31      using System.DirectoryServices;
line 140
    ServiceSecurityContext ssc = OperationContext.Current.ServiceSecurityContext;
    string [] sName = ssc.PrimaryIdentity.Name.Split(';');
    SqlConnection con;
    con = new
    SqlConnection(ConfigurationManager.ConnectionStrings["lookup"].ConnectionStri
    ng);

    string cmdTxt = "SPGetData";
    SqlCommand GetInfo = new SqlCommand(cmdTxt, con);
    GetInfo.CommandType = CommandType.StoredProcedure;

    SqlParameter DNParam = GetInfo.Parameters.Add("@DN", SqlDbType.VarChar);

    DNParam.Value = sName[0];
    con.Open();
    try
    {
        SqlDataReader rdr = GetInfo.ExecuteReader();
        if (rdr.HasRows)
        {
             while (rdr.Read())
             {
                 userName = (string)rdr["UserID"];
                 password = (string)rdr["PW"];
             }
        }
        else
        {
             NotAuthorizedFaultHelper.Throw("User not Registered with CAC");
        }
        rdr.Close();
    }
    catch (System.Exception ex)
        {
             Console.WriteLine("Getting Error:  {0}", ex);
        }
    con.Close();
line 204     After // Submit job
    if (String.IsNullOrEmpty(userName) || String.IsNullOrEmpty(password))
        {
            NotAuthorizedFaultHelper.Throw("User Name or Password is not valid");
```

```
    }
try
{
    //check credentials before submitting to queue
    //User ID or password not valid - possibly expired password
    bool isValid = false;
    string adPath = "LDAP://" + Environment.UserDomainName + "/rootDSE";
    DirectoryEntry adRoot = new DirectoryEntry(adPath, userName,
password,AuthenticationTypes.ReadonlyServer);
    try
        {
            object o = adRoot.Properties["defaultNamingContext"];
            isValid = true;
        }
    catch
    {
        isValid = false;
    }
    if (isValid)
    {
        int retresult = cluster.QueueJob(ccsJob, userName, password, true,
0);
        Console.WriteLine("QueueJob returned: " + retresult.ToString());
    }
    else
    {
        NotAuthorizedFaultHelper.Throw("Credentials not valid, please contact
CAC");
    }
}
catch (Exception ex)
    {
        Console.WriteLine("Exception: " + ex.ToString());
    }
```

In the Project HPCBasicProfileHost the File HPCBasicProfileHost.exe.config

Modify lines 4 and 5 to reflect your head node:

```
    <add key="baseAddress" value="https://localhost/HPCBasicProfile" />
    <add key="clusterName" value="CCSHeadNode" />
```

Replace lines 42 through 44:

```
    <security mode="TransportWithMessageCredential">
    <transport clientCredentialType="None" proxyCredentialType="None"
realm="" />
    <message clientCredentialType="UserName" />
```

With:

```
  <security mode="Transport">
   <transport clientCredentialType="Certificate" proxyCredentialType="None"
realm="" />
```

Replace line 52:

```
          <serviceAuthorization impersonateCallerForAllOperations="true" />
```

With:

```
          <serviceMetadata httpsGetEnabled="true" httpsGetUrl=""/>
```

Replace line 57 and 58:

```
   <messageLogging logEntireMessage="true" logMalformedMessages="false"
    logMessagesAtServiceLevel="true" logMessagesAtTransportLevel="true" />
```

With:

```
   <messageLogging logEntireMessage="true" logMalformedMessages="true"
logMessagesAtServiceLevel="true" />
```

Replace the <system.diagnostics> section with the following: modify c:\tmp to your configuration:

```
<system.diagnostics>
    <sources>
      <source name="System.ServiceModel" switchValue="Verbose">
        <listeners>
          <clear />
          <add type="System.Diagnostics.DefaultTraceListener" name="Default"
              traceOutputOptions="None" />
          <add name="ServiceModel Listener" traceOutputOptions="None" />
        </listeners>
      </source>
      <source name="System.ServiceModel.MessageLogging">
        <listeners>
          <clear />
          <add type="System.Diagnostics.DefaultTraceListener" name="Default"
              traceOutputOptions="None" />
          <add name="MessageLogging Listener" traceOutputOptions="None"/>
        </listeners>
      </source>
    </sources>
    <sharedListeners>
      <add initializeData="C:\tmp\HPCSvcTraceLog.svclog"
              type="System.Diagnostics.XmlWriterTraceListener, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
              name="ServiceModel Listener"
              traceOutputOptions="LogicalOperationStack, DateTime, Timestamp,
ProcessId, ThreadId, Callstack" />
      <add initializeData="C:\tmp\HPCSvcMessageLog.svclog"
              type="System.Diagnostics.XmlWriterTraceListener, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
             name="MessageLogging Listener"
             traceOutputOptions="LogicalOperationStack, DateTime, Timestamp,
ProcessId, ThreadId, Callstack" />
    </sharedListeners>
  </system.diagnostics>
```

Add connectionStrings:

```
  <connectionStrings>
    <!-- this connectionstring element must remain first but others may be
added -->
    <add name="lookup" connectionString="Data Source=MySqlServer; Initial
Catalog=MyLookupTable; Integrated Security=True; Encrypt=True;
TrustServerCertificate=True"/>
  </connectionStrings>
```

Code changes for Client Side HPCBPServiceV1

In the Project HPCBasicProfileClient the File MainModule.cs

Add the following lines of code:

line 27:

```
    using System.Security.Cryptography.X509Certificates;
```

line 210:

```
  X509Store myStore = new X509Store("My", StoreLocation.CurrentUser);
  myStore.Open(OpenFlags.ReadOnly | OpenFlags.OpenExistingOnly);
  X509Certificate2Collection selectedCerts =

  X509Certificate2UI.SelectFromCollection(myStore.Certificates,
     "Certificate selector",
     "Choose your caGrid credential to use.",
     X509SelectionFlag.SingleSelection);
  X509Certificate2 selectedCert = null;
  if (selectedCerts.Count > 0)
  { // if user hits cancel 0 certs are returned.
    selectedCert = selectedCerts[0];
  }

  proxy.ClientCredentials.ClientCertificate.Certificate = selectedCert;
```

In the Project HPCBasicProfileClient the File HPCBasicProfileClient.exe.config

Modify the following lines of code:

```
line 7 and 14 to reflect where the service resides:
          address="https://localhost/HPCBasicProfile"
```

replace lines 23 through 25:

```
<security mode="TransportWithMessageCredential">
<transport clientCredentialType="None" proxyCredentialType="None" realm="" />
<message clientCredentialType="UserName" />
```

With:

```
<security mode="Transport">
<transport clientCredentialType="Certificate" proxyCredentialType="None"
realm="" />
```

Replace line 39:

```
  </system.serviceModel>
```

With:

```
  <diagnostics>
    <messageLogging logEntireMessage="true" logMalformedMessages="true"
      logMessagesAtServiceLevel="true" logMessagesAtTransportLevel="true" />
  </diagnostics>
  </system.serviceModel>
  <system.diagnostics>
    <sources>
```

```
      <source name="System.ServiceModel" switchValue="Verbose">
        <listeners>
          <clear />
          <add type="System.Diagnostics.DefaultTraceListener" name="Default"
              traceOutputOptions="None" />
          <add name="ServiceModel Listener" traceOutputOptions="None" />
        </listeners>
      </source>
      <source name="System.ServiceModel.MessageLogging">
        <listeners>
          <clear />
          <add type="System.Diagnostics.DefaultTraceListener" name="Default"
              traceOutputOptions="None" />
          <add name="MessageLogging Listener" traceOutputOptions="None"/>
        </listeners>
      </source>
    </sources>
    <sharedListeners>
      <add initializeData="C:\tmp\HPCClientTraceLog.svclog"
              type="System.Diagnostics.XmlWriterTraceListener, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
              name="ServiceModel Listener"
              traceOutputOptions="LogicalOperationStack, DateTime, Timestamp,
ProcessId, ThreadId, Callstack" />
      <add initializeData="C:\tmp\HPCClientMessageLog.svclog"
              type="System.Diagnostics.XmlWriterTraceListener, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
             name="MessageLogging Listener"
             traceOutputOptions="LogicalOperationStack, DateTime, Timestamp,
ProcessId, ThreadId, Callstack" />
    </sharedListeners>
  </system.diagnostics>
```

In the Project HPCBasicProfileClient the File HPCBasicProfile.xml

Remove the following lines of code:

lines 23 through 25:

```
      <jsdl:CandidateHosts>
        <jsdl:HostName></jsdl:HostName>
      </jsdl:CandidateHosts>
```

Acknowledgement:
MATLAB on the TeraGrid is based upon work supported by the National Science Foundation under Grant No. 0844032. and grants from Dell, Microsoft Corporation, and The MathWorks. The TeraGrid is an NSF-sponsored open scientific discovery infrastructure that unites people, resources, and services to enable discovery in U.S. science and engineering. For more information, visit www. cac.cornell.edu/matlab. The authors also wish to acknowledge Marty Humphrey of the University of Virginia who provided valuable assistance with this technology.