Scalable Node Monitoring

Alex Drotar – University of Colorado
Erin Quinn – Fairleigh Dickinson University
Landon Sutherland – New Mexico State University

Mentors: Mike Mason – HPC3, Jon Bringhurst – HPC3
Project Description

- Build a high performance computer
- Create a tool to monitor node applications in Component Based Tool Framework (CBTF) using code from Lightweight Data Metric Service (LDMS)
Importance

• Need a scalable, parallel tool to monitor nodes on clusters
• New LDMS plugins need to be able to be easily added to tool
Our Cluster

- Built from the ground up
  - 8 nodes
  - Running CentOS 6.2
  - 8 Cores each
  - 16Gb RAM each
  - WareWulf-provisioned
CBTF

• CBTF stands for “Component Based Tool Framework”
• Scalable
• Adjusts to different topologies automatically
• Uses MRNet (Multicast/Reduction Network) mechanism for information transport
CBTF

• CBTF is flexible and general enough to be used for any tool that needs to do a task on many nodes

• Components are reusable and “EASILY” added to a new tool
Three Levels of CBTF

• Frontend Node
  – Interacts with user

• Filter Nodes
  – Filters or concatenates information from backend nodes

• Backend Nodes
  – Where the actual work of the tool is done
How CBTF Works

• 3 Main Files
  – Tool File – loads files, MRNet setup, interacts with CBTF network
  – Component (Plugin) File – Components are defined and given instructions
  – XML File – sets up connections between components and directs data streams
LDMS

• LDMS stands for “Lightweight Data Metric Service”
• Tool used for monitoring nodes
  – Information from /proc/
    • Vmstat, meminfo ...
• Created an application layer
• Created Dynamic Libraries
Why Change LDMS?

• Fat tree implementation
  – CBTF uses MRNet
Implement LDMS Code Using CBTF
Ltool

- Ltool is the name of the tool we derived from LDMS
- Dynamically linked
- Includes the following components:
  - Vmstat
  - Meminfo
  - Procinterrupts
  - ...and more
Expected Results

CPU Trend

CPU Usage (%) vs. Time

0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100

0 20 40 60 80 100 120

LA-UR-12-23629
LDMS to CBTF

- Allows new LDMS plugins to be simply transferred into our tool
How Does It Work?

- Ltool command is run on the frontend node
- Ltool collects information from the backend nodes
- Backend nodes send information to the filter nodes
- Filter nodes concatenate information and send to a database on the front end node
CBTF – Ltool

• Frontend Node
  – Allows user to select plugin to run

• 2 Filter Nodes
  1) Collects X amount of messages from each node
  2) Adds the information to the database

• Backend Node
  – Runs LDMS application
  – Runs on a specified interval
Database

• MySQL
  – Each plugin has its own table
    • Allows to query by plugin
    • Simplifies future additions
  – Information connected by MySQL dynamic script
    • Allows to be easily moved to new machines / users
      – No predatabase information needed
Ltool Overhead (Problem Size = 100)

- No CBTF
- Meminfo
- Procstatutil
- Linear (No CBTF)
- Linear (Meminfo)
- Linear (Procstatutil)

Wall Clock Time (seconds)

Trial

0  2  4  6  8  10  12
Results

• LDMS was successfully implemented into a CBTF tool, and that the overhead involved with running the tool is relatively low.
Conclusions

• Ltool is a useful tool when it comes to monitoring nodes on a cluster because the overhead involved with running the tool is not particularly high and it will automatically scale to any size cluster.
Future Work

• Ltool can be tested to see if it can run consistently for numerous days
• Create triggers for code to run with Ltool to allow multiple components to run at different times
• Multiple components executing at the same time
• Use MRNet to filter more data
Any Questions?
Thank You

- Dane Gardner
- Our Mentors: Mike Mason, Jon Bringhurst
- TJ Machado
- Jim Brandt
- Ann Gentile