CMPE 283 – Virtualization

Group project: this project is implemented by group effort.

Due date: December 2nd (Monday), 2013

Project Overview

The theme is a large-scale statistics gathering and analysis tool in scalable virtualized environments

The goal of this project is
1. Practice with virtualized environment, managing and testing VM.
2. Apply open source tools like Scribe
3. Understanding the need of gathering and analyze for large data
4. Visualize the collected data using the tools like Vaadin UI or Chart4J or google charts

Problem description
Develop large-scale statistics gathering and analysis in scalable virtualized environments. Specific areas of interest include health models for multi-tier applications, VM and host performance, and detection of anomalies.

1. Framework
Build a framework for large scale metrics data collection and analysis and display of the collected and analyzed data
The types of operations that the framework will have are:
1. Performing collection of system data (metrics, logs) to identify workloads on system elements (jobs, hosts, guests etc) as you have done in project 1 and store them in noSQL Database such as Cassandra or Mongo DB. Data rate can reach 100,000 per minutes.
Monitor Hosts and collect the following metrics (Per Guest Level CPU, Memory, Threads, I/O, Locks, Site ID, Domain Cookie, VMotion and VFT messages)
2. Applying higher-level processing (average per 5 mins intervals, etc) on collected system data to generate abstract views of the system. You can store them in Relational DB such as MySQL, Postgress, BerkeleyDB.
3. Presenting and/or visualizing the outcomes of the above steps in a simple manner.

4. Experiment security options in ESXi such as installing security option for ESXi and experiment with various security options, collect statistics, analyze performance measurements, and report the results.

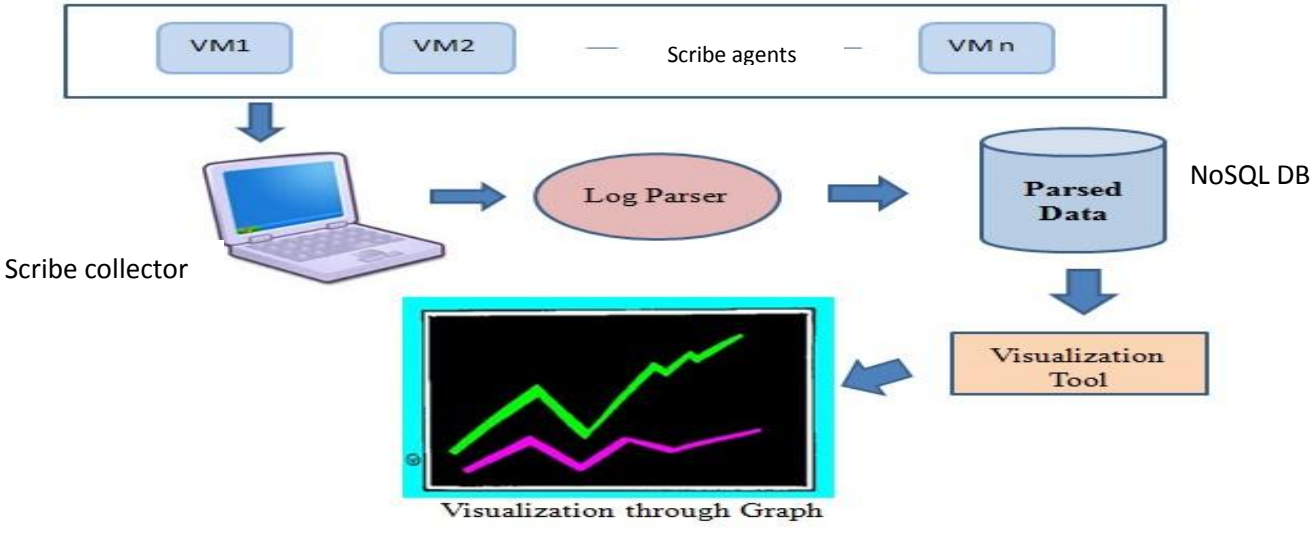
2. Design Components
The system will have at the minimum the following components:
- One or more agent processes sitting at the ESXi kernel level that collect and send monitoring data to the collector
- In order to measure health of application across multiple tiers, access to packets is required.
- Collector Process that will Poll, Process and Archive collected data
- Analysis module to process the collected data
- A Supervisor module that can configure/ manage/ control the various agents
- A simple visualization Tool for above data

3. Configurations

- List of agents and locations
- Environment and configurations for Collectors, port numbers for agents, locations to store data, Polling intervals, etc Startup and Shutdown
- Agents and Collector can be started independently
- Start and stop scripts can also be used to manage them independently

Simulate work load - There needs to be a way to stimulate activity. This could involve some free apps that are designed to stress-test a real CPU, like Prime95 or Folding@Home to keep a vCPU busy. The students could start up one or more instances of these apps and try to force VMware to auto-migrate to another Host with more CPU resources. (Most of these are all platform: Windows, Linux, Mac.)

4. Process and Data Flow



1. Agents will collect the configured metrics and commit as local record tables on each host, every 5 seconds
2. Pollers poll the agents and get record tables back via suitable Transport (TCP/ UDP).
3. After collection is successful, data records will be archived and the engines will be notified.
4. These engines will then analyze and reduce data, and create second set of tables that has analyzed values
5. Hourly Analyzed Record Rollup Process will kick in every hour for rolling up data into 24 sets
6. Daily Analyzed Records Rollup. 24 hour Process will kick in for rolling up data from the 24 sets into daily sets
7. Purge and Cleanup
8. Visualization widgets will run on top of the collected and analyzed data. E.g. Per VM and System wide Histograms, Heat Maps etc, all with events coalesced by time, allowing superimpositions. Latency of Tiered applications co-hosted with per VM Guest/ Network statistics etc

5. Other Related Work

There are other systems and frameworks that collect logs and perform analysis.

1. SPLUNK treats each log entry as an event and does not attempt to correlate.
2. SALSA tries to derive control and data flow and presents state machine views.
3. Prime95:http://en.wikipedia.org/wiki/Prime95 Folding@home (Stanford): http://folding.stanford.edu/English/Guide

6. Deliverables

- Complete source code
- Screenshots
- Test scripts
- Project report (Why? and Why not?)
    - Introduction (goals, objectives, needs, …)
    - Background (context of project)
    - Requirements (functional and nonfunctional)
    - Design (architecture, components, key workflows, …)
    - Implementation (environment, tools, approaches, scripting language and tools uses, screenshots, who did what, …)
    - Assumptions: What are the assumptions and why?
    - Limitations: What are limitations and why?
    - Future Work: how this work can be extended?
    - Individual Contribution: Who did what?
    - Installation and Execution manual: What and how to execute your codes? Show screenshots.
    - System Testing Results: What/How did you test? Describe all your test methodologies, test cases and show screenshots.
    - Testing (unit, integration, who did what, …)
    - Screenshots with annotations
    - Conclusion (lessons learned, challenges …)
    - References

7. Submission

- **On-line submission**: Submit all codes and report as one zip file. Submissions shall be made via email to **sjsucmpesubmission@gmail.com** with subject line "CMPE283 Project 2 submission team # Smith, Jones, Sing, (your last names)". **Do not send it to the group mailing**. DO NOT WAIT UNTIL LATE ON THE DUE DATE, as email server lags or delays may result in a late submission. If you are concerned about this, submit your assignment early. Shall include image captures of your program working.
- **Demo and presentations are on/after due date.**
- **No hardcopy submission is required.**