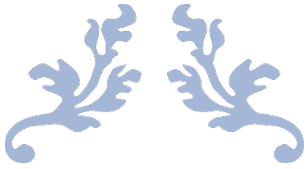

CSC8106:SYSTEM EVALUATION

A1 Report

NOVEMBER 26, 2017

AREEB ALSHOSHAN

Exercise A1

A. & B. Description of the built system and the scripts.

Briefing:

The built system is a system that use the features of Simple Networks Management Protocol (SNMP) to compute the measurement of the CPU user load percentage for every 5 seconds over a given time period. Where, the original targeted operating system is (17dcompv008) that is installed in Newcastle university and it is actually a virtual Linux operating system. However, after gathering a good amount of CPU load % for more than three hours by store it in a text file, the system uses this information to find the average CPU load over the time period and compute the confidence interval. The following paragraphs will describe the system step by step by describing the shell scripts.

1.Computing the CPU user Load for an interval time:

To computer the CPU user load over a period of time, first we need to find the right scalar objects to compute the CPU user load and according to ORACLE docs it can be compute as following:

- The needed scalar objects from SNMP are ssCpuRawUser, ssCpuRawNice, ssCpuRawSystem and ssCpuRawIdle.
- To compute the CPU user load % between two polls use the following steps:

  CPU user load % =

  $$\left(100 * \left(\text{ssCpuRawUser at t1} - \text{ssCpuRawUser at t0}\right)\right) \div ((\text{ssCpuRawUser} + \text{ssCpuRawNice} + \text{ssCpuRawSystem} + \text{ssCpuRawIdle})\ at\ t1 - (\text{ssCpuRawUser} + \text{ssCpuRawNice} + \text{ssCpuRawSystem} + \text{ssCpuRawIdle})\ at\ t0)$$

- The previous equation applied in my shell script after retrieving the needed scalar objects using *snmpwalk* with its corresponding object then

take the last part of the retrieved information which is the needed filed that contains the number of ticks, as following:

```
#!/bin/bash

touch cpuAvglast.txt
cpuLoad=0.0

function snmpData {

 cpuUserT0=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawUser | awk
'{print $NF}')
 cpuNiceT0=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawNice | awk
'{print $NF}')
 cpuSysT0=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawSystem | awk
'{print $NF}')
 cpuIdleT0=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawIdle | awk
'{print $NF}')

 cpuAllT0=$(( cpuUserT0 + cpuNiceT0 + cpuSysT0 + cpuIdleT0 ))


  sleep 5s

 cpuUserT1=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawUser | awk
'{print $NF}')
 cpuNiceT1=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawNice | awk
'{print $NF}')
 cpuSysT1=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawSystem | awk
'{print $NF}')
 cpuIdleT1=$(snmpwalk -v 2c -c allInfo 17dcompv008 ssCpuRawIdle | awk
'{print $NF}')

 cpuAllT1=$(( cpuUserT1 + cpuNiceT1 + cpuSysT1 + cpuIdleT1 ))


CpuUserAll=$((cpuUserT1-cpuUserT0))
CpuAll=$((cpuAllT1-cpuAllT0))


cpuLoad=`bc<<<"scale=20 ; 100*$CpuUserAll/$CpuAll"`



}
```

2.Store the CPU user load results in a text file:

after computing the CPU user load the result stored into a text file each time the above function invoked in the for loop, where the for loop keep work to satisfy the requirement for entering the time period as a parameter and the shell code snippet is:

```
let count=$1/5

for ((i=count; i>0; i--)) #For loop
do
#invoke the function that bring the snmpwalk data
snmpData
echo "$cpuLoad" >> "cpuAvglast.txt"
echo "The CPU user average $cpuLoad ,the loop $i"
done
```

3.Display the average CPU user load over the time period :

Using the generated text file that store the CPU user load every 5 seconds for a period of time, I wrote a shell scrips using the shell scripts which published in the Blackboard to find the average CPU load over the time period as folloing:

```
#This loop well calcualte the Mean (average)=
summation=0.0
count=0
while read line #Read every line and assign to varable $line....
do
    summation=`bc<<<"scale=20 ; $summation + $line"`
        let count=count+1
done < cpuAvglast.txt

echo "The counter is : $count"
mean=`bc<<<"scale=20;$summation / $count"`
echo "The Average : $mean"
```

4.Display the confidence interval:

The confidence interval is range of values that suggest that the resulted values will lie in. Before we compute the confidence interval we need to compute the variance and the standard deviation using the following equation:

- Variance : $\overline{E[(X - E[X])^2]} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{E[X]})^2$

and the shell script I coded to fulfill this requirement is:

```
#This loop calcualtes the variance
total=0.0
varP=0.0
while read line #Read every line and assign to varable $line....
do
varP=`bc<<<"scale=20 ; ($line - $mean)^2"`
total=`bc<<<"scale=20 ; ($varP + $total)"`
done < cpuAvglast.txt

let count=count-1

variance=$(bc<<<"scale=20;($total/$count)")
echo "The variance : $variance"
```

and the standard deviation is $\sqrt[2]{variance}$ and the shell code to find it is:

```
sDev=$(bc<<<"scale=20;sqrt($variance)")
echo "The standard Deviation :$sDev"
```

Finally, the equation to compute the confidence interval is: $[\overline{E[X]} - \frac{c_p.sd}{\sqrt{N}}, \overline{E[X]} + \frac{c_p.sd}{\sqrt{N}}]$

Where c_p is a constant depend on the required percent of the confidence interval as following $\begin{matrix} p = 95\% \rightarrow c_p = 1.65 \\ p = 99\% \rightarrow c_p = 1.96 \end{matrix}$ and for my shell script I chose C_p=1.96 and the shell code to the confidence interval is:

```
#finding the confidence interval
let count=count+1
minVal=`bc<<<"scale=20 ; $mean-(1.96*$sDev)/sqrt($count)"`
maxVal=`bc<<<"scale=20 ; $mean+(1.96*$sDev)/sqrt($count)"`

echo "The confidence interval : [$minVal , $maxVal]"
```