GVP Troubleshooting Guide

Collecting Data

12/19/2025

Collecting Data

Windows

This section describes the PerfMon counters, which enables you to check CPU and memory use. The PerfMon counters are detailed below:

- Process counters (`pwcallmgr.exe`, `ccpccxml.exe`, `ssg.exe`, `java.exe`, `resourcemgr.exe`, `CTIConnector.exe`):
  - % Processor Time
  - Working Set
  - Private Bytes
  - Handle Count
  - Thread Count
  - Virtual Bytes
- Memory counters:
  - Available Kilobytes
  - Committed Bytes
- Processor counters:
  - % Processor Time (choose _Total from Select Instances From List)
- LogicalDisk Counters (choose _Total from Select Instances From list)
  - Avg Disk Bytes/Read
  - Avg Disk Bytes/Write
  - Avg Disk Queue Length
- CPU time for the executable program. (This is a column in the Task Manager.)

Make sure the PerfMon data is written as a `.csv` file, not binary. The collection interval should be 15 seconds.

Linux

You can use the following commands to collect data:

- `top`—to monitor CPU and memory usage per process, and to monitor high level system CPU and memory usage (installed with Linux: `procps-3.2.3-8.9.i386.rpm` on RHEL)

- sar—to monitor detail system resource usage (installed with Linux: `sysstat-5.0.5-16.rhel4.i386.rpm` on RHEL)
- gvpfd—to monitor per process file descriptor usage

```
gvpfd source
$owner = "all";
$duration = shift;
$repeat = shift;
$logfile = shift;
$pattern = "/usr/local/phoneweb/logs/logProcess.txt";
sub trim($)
{
  my $string = shift;
  $string =~ s/^\s+//;
  $string =~ s/\s+$//;
  return $string;
}
sub tabify($)
{
  my $string = shift;
  $string =~ s/\s+//g;
  return $string;
}
print "Starting vgfd\n”;
  print "Process Owner: $owner \n”;
  print "Interval: $duration seconds \n”;
  print "Repeat by: $repeat times \n”;
  print "Log file: $logfile \n”;
  open (LOGFILE, ">$logfile");
print LOGFILE
"TIME\tNumFD\tPID\tUSER\tPR\tNI\tCPU\tTIME+\tMEM\tVIRT\tRES\tSHR\tS\tCOMMAND
  for ($i=0; $i<$repeat; $i++){
  $ts = ‘date +"%D %T"‘; chop($ts);
my @list;
if ($owner eq "all"){
  @list=’ps -A | grep -v PID | grep -f $pattern’;
}else{
   @list=’ps -u$owner | grep -v PID | grep -f $pattern’;
}
$index=0;
while ($list[$index]){
  $list[$index]=trim($list[$index]);
  @token = split(/\s+/,$list[$index]);
  $pid = $token[0];
  $pname = $token[3];
  $numFD = ‘find /proc/$pid/fd -not -type d | wc -l’; $numFD=trim($numFD);
  $top = ‘top n 1 b | grep $pid | grep $pname’; $top=tabify(trim($top));
  print LOGFILE "$ts\t$numFD\t$top\n”;
  $index++;
  }
 sleep $duration;
 }
 close (LOGFILE);
 exit 0;
```

To initiate all three process, you can use the following script:

gvpmon source

```
echo "Starting gvpmon script"
N=$1
```

```
(( D=1+$2/$N))
rm -f ~pw/logs/sar.dat
nohup /usr/lib/sa/sadc $N $D /var/log/sar.dat >/dev/null &
nohup /usr/bin/top -d $N -n $D -b > /var/log/top.log &
nohup ./vgfd $N $D /var/log/fd.log >/dev/null &
```

It takes two arguments:

- arg 1—interval between each snapshot captured in seconds
- arg 2—number of snapshot to capture

The preceding scripts/programs require a super user privilege to execute. After creating the two preceding scripts, make sure that you add executable permission on them prior to running, by issuing: `chmod a+x {script's filename}`