CallManager Server: Use PsList to Troubleshoot a Memory Leak Problem

Document ID: 66967

Contents

Introduction

This document describes how to use the PsList utility in order to debug a memory leak problem on a Cisco CallManager server. Windows operating systems support the Performance Monitor utility that administrators use to view detailed information about process CPU and memory usage. However, it does not capture memory utilization for processes that started after the Performance Monitor utility starts. The PsList serves to fill this gap.

Note: PsList is a free utility. You can download it from SysInternals ⬀ .

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Windows Operating Systems
- Cisco CallManager
- Windows Performance Monitor

Components Used

The information in this document is based on these software and hardware versions:

- Cisco CallManager

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Background

The default behavior of PsList is to show CPU−oriented information for all the processes that currently run on the local system. The information listed for each process includes the time the process executes, the amount of time the process executes in kernel and user modes, and the amount of physical memory that the OS assigns the process. Command−line switches allow you to view memory−oriented process information, thread statistics, or all three types of data.

Usage

This is the format of the utility:

```
pslist [-?] [-d] [-m] [-x][-t][-s [n] [-r n]][\\computer [-u username] [-p password]] [name
```

This list provides a detailed description of each parameter:

- **−?** Displays the supported options and the units of measurement used for output values.
- **−d** This switch has PsList **show** statistics for all active threads on the system, and groups threads with their own process.
- **−m** This switch has PsList **show memory**−oriented information for each process, rather than the default of CPU−oriented information.
- **−x** With this switch, PsList shows CPU, memory, and thread information for each of the processes specified.
- **−t** Shows the tree of processes.
- **−s [n]** Causes PsList to run in a mode similar to a task manager update mode. Press the **Escape** key to optionally specify the number of seconds it runs and abort this mode.
- **−r n** Task manager mode refresh rate in seconds (default is 1).
- **name** Instead of listing all the running processes in the system, this parameter narrows PsList's scan to those processes that begin with the name process. For example, **pslist exp** displays statistics for all the processes that start with "exp". This includes Explorer.
- **−u** This parameter represents the username. If you want to kill a process on a remote system and the account you are executing in does not have administrative privileges on the remote system, then you must login as an administrator using this command line option. If you do not include the password with the −**p** option, then PsList prompts you for the password without echoing your input to the display.
- **−p** This parameter represents the password. This option lets you specify the login password on the command line so that you can use PsList from batch files. If you specify an account name and omit the −**p** option, PsList prompts you interactively for a password.
- **\\computer** Instead of showing process information for the local system, PsList shows information for the NT/Windows 2000 system specified. Include the −**u** switch with a username and password to login to the remote system if your security credentials do not permit you to obtain performance counter information from the remote system.
- **pid** Instead of listing all the running processes in the system, this parameter narrows PsList's scan to the process that has the specified Process ID (PID). For example, pslist 53 dumps statistics for the process with PID 53.

Setup PsList on the Cisco CallManager Server

Complete these steps to setup PsList on the Cisco CallManager server:

1. Create a batch file (memory.bat) with Notepad or a similar application. The content of the batch file contains:

```
echo %date% %time% >>c:\cron\mem.txt
c:\dnld\pslist.exe -m >>c:\cron\mem.tx
```
2. Enable the Task Scheduler service on the Cisco CallManager server.

      a. Select **Start > Programs > Administrative Tools > Services**.
      b. Right−click **Task Scheduler** (see arrow A in Figure 1).
      c. Click **Start** (see arrow B in) from the popup box.

      Figure 1: Services
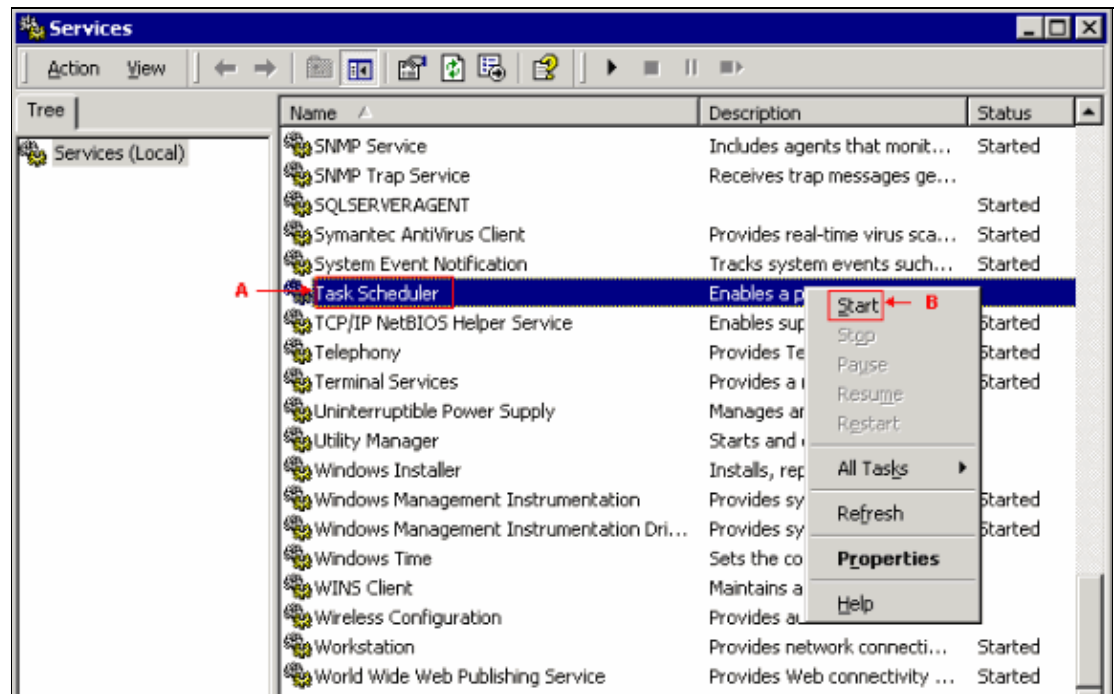
3. Schedule your batch file to run periodically.

Cisco recommends an interval of 10 minutes. You can adjust this interval subject to the problem need. It only takes a few seconds to run and therefore does not increase the load.

      a. Click **Start**, point to **Settings**, and click **Control Panel**.
      b. Double−click **Scheduled Tasks**.
      c. Click **Next**.
      d. Click **Browse**.
      e. Navigate to select the program to schedule.

      In this case, the directory is cron (see arrow A in Figure 2) and the program is mem.txt (see arrow B in Figure 2).

      Figure 2: Select Program to Schedule
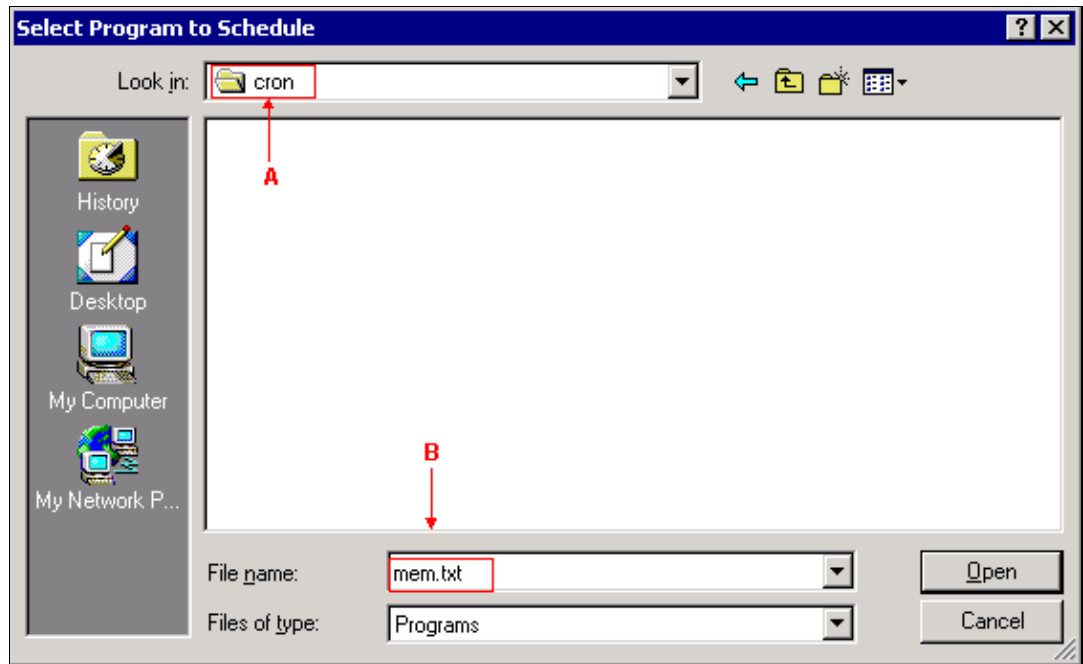
f. Click **Open**.

> **Note:** If you use the Scheduled Task Wizard, check **Open advanced properties for this task when I click Finish** and then click **Finish** to continue.

g. Type a name for this task.

h. Click **Daily**.

i. Click **Next**.

j. Select the time and day you want this task to start.

k. Click **Next**.

l. Enter the name and password of a user. The task runs as if it were started by the user.

m. Click **Next**.

n. Check **Open advanced properties for this task when I click Finish** and then click **Finish** to continue.

o. Click **Schedule**.

p. Click **Advanced**.

q. Check **Repeat task** and then specify the number of minutes or hours you want the task to repeat.

r. Click **OK** twice.

4. Wait for usage to grow.

5. Gather the log. For each run of PsList, this is the output:

```
Process memory detail for EVOICE-R16-CM1:

Name            Pid      VM      WS   Priv Priv Pk    Faults   NonP Page
Idle              0       0      16      0       0         1      0    0
System            8    1676     224     24     164    562211      0    0
SMSS            204    5256     376   1076    2760       886      1    6
CSRSS           232   30880    2604   1556    1592      2333      7   61
WINLOGON        256   35580    1296   7084    8800     16883     66   35
SERVICES        284  127396   89808   4032    6636   6713985    622  122
..
..

13:15:29.35
```

> **Note:** Disable the scheduled job when you finish. Otherwise, the log fills the disk.

6. Use the pslistmem.pl script to parse the log and prepend the instance number to every line.

```perl
#!/usr/bin/perl
 $inst=0; #init var to count number of instances
$tinfo=""; #init var to store date/time info
while($_ = <STDIN>){
 #if this line contains a timestamp like 11:11:11.11
 if (/(\d+:\d+:\d+.\d+)/){
  $inst++;      #increment the instance counter
  $tinfo=$_;   #save time/date to append to all rows of this instance
  next; # skip to next line of input
 }
 #if this line contains a word followed by at least 8 numbers
 if (/((\w)(\s)+(\d)+(\s)+(\d)+(\s)+(\d)+(\s)+(\d)+(\s)+(\d)+(\s)+(\d)+(\s
  tr/\r\n//d; #strip CR LF to keep the date/time on the same line
  print "$inst\t$_\t$tinfo"; # print instance counter, input line,
   then date/time for this instance
  next; # skip to next line of input
 }
```

The script prepends the number 1 to all lines in the first set of ouput from PsList. It prepends the number 2 to the second set and so forth (see arrows A, B and C in Figure 2). It also strips header lines. This allows you to sort by PID and use the first column to keep the lines in temporal order.

Note: In order to run this Perl script, you need to download a Perl interpreter such as ActivePerl from www.activestate.com.

7. Invoke the pslistmam.pl script as this output shows:

```
cat mem.txt | pslistmem.pl | sort -k 3 >out.txt
```

Mem.txt and out.txt represent input and output file respectively for pslistmem.pl. It sorts the output by the process name. In the output file (see Figure 3), column 1 is the set number prepended, column 2 is the process name, column 3 is the process ID, column 4 is the virtual memory (VM), column 5 is the working set (WS), column 6 is the private byte count, and column 7 is the peak of private byte count. If you watch the trends of VM, WS, and private byte count, they present you with a general trend of memory consumption by a specific process.

Figure 3: Output for the PsListMem.pl Script

Related Information

- **Download PsList Utility**
- **Download Perl**

- **Voice Technology Support**
- **Voice and IP Communications Product Support**
- **Troubleshooting Cisco IP Telephony** ⬈
- **Technical Support & Documentation – Cisco Systems**

Updated: Feb 03, 2006   Document ID: 66967