SLAC-PUB-3712
June 1985
(N)

Rx for Hung Users*

T.Y. Johnston
Stanford Linear Accelerator Center
2575 Sand Hill Road
Menlo Park, CA 94025

Installation Code: SLA

VM Systems Management Project

0608

ABSTRACT

An examination of a methodology for freeing hung virtual machines.

Presented at the SHARE 64 Conference, Los Angeles, California
February 24 – March 1, 1985

* Work supported by the Department of Energy, Contract DE-AC03-76 SF00515.

INTRODUCTION

The Stanford Linear Accelerator Center runs a 3081K and a 3033U under VM HPO 3.4. SLAC's version of VM batch is run on both processors. The 3033 is a batch only machine, while the 3081 supports both batch and interactive computing. A profile of the system during peak hours reveals:

1. Over 400 logged on users.
2. About 300 interactive users connected as follows:
   a. 115 on real 3278s.
   b. 135 on ASCII terminals through 5 series/1s to emulate either a 3278 mod 2 or mod 4.
   c. 50 on ASCII terminals in line by line mode.
3. Around 70 service virtual machines running including batch workers.

In this environment one would expect to see a large number of hung users, but that is not our current experience. We see many fewer today than we did say three years ago when we had approximately half the number of users. This is probably primarily due to improvements in software. I have heard about a total of 6 cases since Share 63.5 (November 14). Thus my sample is quite small. However, let me discuss our approach.

SLAC METHODOLOGY

We have a fundamental philosophy: FIX THE PROBLEM, NOT SYMPTOMS OF THE PROBLEM! There is a reason why the virtual machine is hung. Determine what that reason is and attempt to clear it. The virtual machine hang is probably

caused by an I/O operation that did not complete. If the hang is not associated with a user I/O operation, then you are probably not going to be able to fix it.

We run with the missing interrupt handler disabled. This goes back to treating the problem not the symptoms. If a control unit has a red light failure, doing HALT commands to devices on that control unit is not going to help. The control unit must first be reset before the problem can be corrected. Many preemptive software actions will make solving the problem even more difficult than before. There is another danger in the missing interrupt handler approach with failing hardware. A hardware device has failed to work properly, and now a set of actions are going to be done to that hardware. The software now expects the hardware to function correctly during the recovery process!

A practice that we discourage for hung users is the use of the CP FORCE command. If a hung user is in execution wait or console function wait (either flag VMCFWAIT or VMEXWAIT is turned on), then FORCE will not log off the user until these flags are reset. The proper action is to clear the state of the machine so that these flags are turned off. Forcing has the objectionable side effect when it is not successful that many commands no longer work. One cannot query about the virtual machine or if one is the secondary console for the machine one cannot use the SEND CP function to attempt to clear the hang state in the VM. This is particularly confusing to operations personnel, but is a pain to anyone who is trying to clear the problem.

Before discussing what to do, I would like to mention some of the hung states

that can exist. These are:

1. Instruction simulation wait. This is usually an I/O being done with diagnose I/O.
2. I/O wait.
3. Page wait. The virtual machine is waiting for a paging I/O operation. Fortunately, this happens rarely, since we have never been able to do anything about freeing such a machine. Note that this is not a user I/O operation.

TOOLS

Tools are the most important part of solving problems. IBM ships VM with almost no diagnostic tools. Let me discuss some that we use.

1. STATUS command. We obtained this from a tools tape. It is probably the most important single tool we use. It provides status for any virtual machine (including one that is logging off). The output contains the registers and psw, the I/O state of the machine, the last CP command, the last PRIVOP, and a host of other indicators.
2. DCONBUF command. SLAC written to display the in core page of the virtual machines console (if the console is spooled).
3. IOINT command. SLAC written to provide at the CP level any I/O interrupt for a real device. Interrupts such as channel available, control unit end, and device end can be presented to CP.
4. Query I/O command. SLAC written using SIO counts in RDEVBLOK to show SIO counts for a real device.

5. QVIO command. SLAC written using SIO counts in VDEVBLOK to show SIO counts for a VMs virtual devices.
6. KILL command. SLAC written command to generate a program check in a virtual machine.
7. SMART virtual machine. IBM field developed program. More useful for seeing abnormal system performance than for solving problems with a hung virtual machine.
8. HALT command. CP class A command to halt a real device.
9. DEVSTA display. 308x processor controller display to show the device status for all devices on a channel.
10. CHNCFA display. 308x processor controller display which allows the resetting of REAL channels.
11. ALTCONS EXEC. SLAC written EXEC to allow a privileged user to dynamically become the secondary console for a disconnected virtual machine.
12. WHATIS EXEC. SLAC written EXEC to provide a formatted display of the RDEVBLOK including status information.

EXAMPLES

I would now like to take five different hang examples and examine what action might be taken. The examples are:

1. A user virtual machine is hung in logoff and it is hung on terminal I/O to a 327x terminal.
2. A user virtual machine is hung in logoff and it is hung on terminal I/O to a line

by line terminal on a 3705.

3. A user virtual machine is hung on a tape drive.
4. A service virtual machine was attempting to IPL a strange device and hung itself and the channel.
5. A file on a CP printer is hung and cannot be cleared.

In each of these cases the same basic methodology is followed. Determine what the I/O problem is and clear the problem. Let us look at the various states than might exist from both a hardware and a software standpoint:

- Device is busy to both software and hardware.: In this case a HALT command should be able to clear the busy state.
- Device is busy to software and not busy to the hardware.: In this case if the device is a dedicated device such as a printer you should try doing STOP and START of the physical device (take the device from a ready state to a not ready and back to ready). If this does not correct the problem then an IOINT with the device end option may clear busy state.
- Device is busy to the hardware and not busy to the software.: In this case you should try and reset the device. If you cannot reset the device then a channel reset can be done.
- Device is not busy to either the hardware or software.: This is not a case that would apply to a virtual machine, but it can happen for a CP owned device such as a printer. Somehow the I/O block has been lost. Device end interrupts are thrown away as unsolicited.

In doing the problem determination the two most important tools are the STATUS command and the DEVSTA display. The STATUS command allows one to see the state of a virtual machine and the DEVSTA display allows one to see the state of a real device. Other commands such as Query IOcounts will show whether this state is constant or changing.

Hung User on a 3270 This was probably caused by some sort of a hardware problem with the terminal or controller. To clear it you do an IOINT DE (device end) for the terminal address. This will cause CP to force disconnect the user and the I/O will be cleared.

Hung User on a Line by Line Terminal This is a much more difficult case. A 3705 line is always busy because of the prepare command. Something that sometimes will work (with almost no danger) is a CP WNG message to the user. Doing an IOINT DE can hang the channel. If the software shows busy and the hardware shows not busy then the IOINT DE should work. If the hardware shows busy, then you must clear the hardware busy condition. There are procedures for clearing an I/O on a 3705 via the panel. This is the only way to free the user.

Hung User on a Tape Frequently, this is caused by an operator unloading a tape drive that still requires some I/O to free the device. Putting a scratch tape up on the drive may clear the condition. If this does not clear it and the hardware does not show busy, then an IOINT DE could cause CP to redrive the device.

Hung Service Virtual Machine A service virtual machine was attempting to IPL a control unit/device and it hung. The channel is probably also hung and not available

for other devices. The following sequence might work:

1. A KILL command is issued to the VM to make certain that it does not continue to drive I/O to the device.
2. Use of the DEVSTA frame of the 3082 will indicate which device is busy on the channel. It would be expected that the device would stay busy.
3. The CP HALT command is issued for the busy device. This is frequently sufficient; the channel is freed and the service VM can be logged on to, forced, or whatever is appropriate.
4. If this is not sufficient, then the CHNCFA frame can be selected and the channel reset and placed back on online. This will probably cause lost interrupts for other devices on the channel.
5. When SMART complains about the other interrupts on the channel issue an IOINT command with the device end option for each of SMART's indicated devices.

Hung File on a Printer A micro-programmed printer went down with hardware problems with a spool file on the printer. It is necessary to get the spool file off of the printer. From CP's viewpoint the printer is busy. To clear the problem:

1. Do an IOINT command for the printer with the STAT option of 8000. This generates an attention interrupt for the device. This is an unexpected completion to CP and it will hold the file and free it off the device. A device end would be an unsolicited device end and would be ignored.

SUMMARY

In each of the foregoing examples a hardware failure has triggered a software situation that CP has not handled well. I think one could say with 99.44the inability of software to cope with a hardware failure. Part of this is unavoidable in the software. "SMART" control units and devices can be re-imled without the software ever knowing that such an event has taken place. Although, this might not be classifiable as a hardware failure, it certainly makes it difficult for the software to manage the resource. Consequently, when a hang situations occurs it is important to determine why it happened. That is not always possible, but it generally is. Thus as in most cases of problem correction, problem determination is the most important aspect of solving the problem.

Once problem determination has been completed, then problem correction can be commenced. It should be noted in my list of tools that we used many more problem determination tools than problem correction tools. The hard thing is usually trying to determine what really went wrong. The 308x processors provide some powerful tools for problem isolation and correction. Learn them. Prayer and luck help too!