
Rule WLM070: Terminal Output Wait swaps occur too often

Finding: CPEXpert has determined that an excessive number of Terminal Output Wait swaps occurred during the measurement interval.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on performance of your computer system. The level of impact depends upon how many Terminal Output Wait swaps occurred and upon your management objectives.

Logic flow: The following rule causes this rule to be invoked:
WLM450: Swap-in delay helped cause service class to miss goal

Discussion: Terminal Output Wait swaps occur after the SRM has been notified that a TSO session is in terminal wait after issuing a TPUT, and the address space is in a long wait condition. Terminal Output Wait swaps occur for a variety of reasons: (1) waiting for terminal output buffers, (2) TSO logon processing, (3) TSO batch SUBMIT, (4) entering and leaving ISPF, and (5) others.

- Terminal Output Wait swaps can occur because transactions are waiting for terminal output buffers. This means that TSO commands or CLISTS have produced output so fast that the TIOC does not have sufficient buffers to handle the terminal output. This cause has been touted as the primary cause of Terminal Output Wait swaps in many articles and journals¹, and CPEXpert earlier considered insufficient terminal buffers to be the cause of Terminal Output Waits swaps.
- A Terminal Output Wait swap occurs whenever a TSO user issues a TSO batch SUBMIT command. It is unclear why this occurs, but the fact that a Terminal Output Wait swap follows a TSO batch SUBMIT command has been verified using GTF trace of SRM SYSEVENTs.

¹As examples:

- "OS/VS2/MVS Resource Measurement Facility (RMF) Reference and User's Guide", publication SC28-0922, IBM Corporation.
- "MVS/XA Performance and Resource Measurement Facility (RMF) Data Analysis", J. William Mullen, *CMG'88 Conference Proceedings*, The Computer Measurement Group, Chicago, IL.
- "Cheryl Watson's TUNING Letter", January 1991, Watson & Walker, Inc., Lutz, FL

-
- Terminal Output Wait swaps can occur during TSO logon processing, if the terminal control address space (TCAS) program has not been marked non-swappable in the Program Properties Table in SCHED00.

When the TCAS is swappable, the SRM will often find that the address space has not been executed for a period of time, and cause the address space to be swapped out for a Detected Wait Swap.

If the TCAS program is swapped out for a Detected Wait Swap when a TSO user attempts to logon, the TSO user goes into a long wait and the SRM will swap the TSO user out for a Terminal Output Wait swap. The TCAS will be marked Ready and be swapped in to service the logon. Then the TSO user will be swapped in to perform whatever action is next appropriate.

- Terminal Output Wait swaps may occur if the VTAM **IOBUF** buffer pool start options are specified incorrectly. The IOBUF values describe how VTAM will manage the message pool in fixed storage. The **baseno** value establishes the base number of buffers in the pool. If this value is too low, VTAM will expand the buffer pool as required, and subsequently contract the pool. Any TSO user causing VTAM to expand the message pool will be swapped out for a Terminal Output Wait swap.
- Terminal Output Wait swaps occur whenever a TSO user enters or leaves ISPF.
- Terminal Output Wait swaps may occur whenever a TSO users operating under a multi-session manager transfers to another session. (We have been unable to verify that Terminal Output Waits result from this situation. However, preliminary analysis of GTF Trace of SRM SYSEVENTs indicate that Terminal Output Wait swaps may occur in this situation.)

The first three situations listed above are correctable. There apparently is nothing that can be done about the other situations.

CPEXpert produces Rule WLM070 if swap-in delay was a significant factor in causing a service class to miss its performance goal, if the service class related to TSO, if Terminal Output Wait swaps accounted for a significant percent of the total swaps, and if Terminal Output Wait swaps exceed the **SWAPTO** guidance variable in USOURCE(WLMGUIDE).

Suggestion: CPEXpert suggests you consider the following alternatives:

-
- **Verify that TSOKEYxx parameters are appropriate.** Terminal Output Wait swaps occur because transactions are waiting for terminal output buffers. This means that TSO commands or CLISTs have produced output so fast that the TIOC does not have sufficient buffers to handle the terminal output. You can determine whether this is a likely cause of the Terminal Output Wait swaps by examining the TSOKEYxx member of SYS1.PARMLIB.

TSO terminal output buffers are controlled by three keywords in the TSOKEYxx member of SYS1.PARMLIB: BUFRSIZE, HIBFREXT, and LOBFREXT.

- The BUFRSIZE value specifies the size in bytes of a VTIOC buffer. Input and output data smaller than the BUFRSIZE value use storage equal to the BUFRSIZE value. Buffers are dynamically allocated equal to the data size if input or output data is greater than the BUFRSIZE value.

If the BUFRSIZE value is too large compared with the actual size of terminal data, central storage will be wasted while the buffers are used. For example, a LISTC command generates 80-character output. Suppose that the BUFRSIZE value were specified as 2048 (this size has been recommended by some authors²). Each buffer would waste 1968 (2048 - 80) bytes while the buffer was holding data. (This amount of wasted storage is not generally significant, considering the vast amount of storage available in modern systems.)

If the BUFRSIZE value is too small compared with the actual size of the terminal data, storage will be dynamically allocated (using a GETMAIN). This dynamical allocation conserves storage, but uses processor resources to execute the GETMAIN and subsequent FREEMAIN.

The BUFRSIZE value generally should be large enough to accommodate the buffer size requirements of 80% to 90% of your TPUTs.

- Unless virtual storage is a constraint, it is better to specify a relatively large value for the BUFRSIZE value. A large value will avoid unnecessary GETMAIN/FREEMAIN processing.

With most modern systems, virtual storage is not a constraint for TSO environments. Consequently, CPExpert normally suggests

²For example, "Cheryl Watson's TUNING Letter", January 1991, Watson & Walker, Inc., Lutz, FL

that a relatively large value (e.g., 2048) be specified for the BUFERSIZE value.

- If virtual storage is a constraint, it is better to specify a relatively small value and avoid wasting virtual storage. This situation is extremely unlikely.
- The HIBFREXT value specifies the maximum amount of virtual storage that can be dynamically allocated for output data. When this value is reached, no more output requests are honored until the LOBFREXT value is reached. The default HIBFREXT value in MVS/XA and MVS/ESA is 48000 bytes.

The HIBFREXT value is used in conjunction with the BUFERSIZE value to determine the maximum number of buffers that can be allocated to a user ($\text{HIBFREXT} \div \text{BUFERSIZE} = \text{maximum number of buffers}$).

If the HIBFREXT value is too low, the buffers can become exhausted and generate unnecessary swaps. For example, consider a BUFERSIZE of 2048 as mentioned above. If the HIBFREXT value were at the default of 48000 bytes, only 23 ($48000 \div 2048$) TPUTs could be serviced before a user would be swapped out for a Terminal Output Wait swap.

Suppose that a LISTC command generated 50 lines. Each line would require a buffer, and only 23 buffers would fill before the user would be swapped out for Terminal Output Wait. The user would not even fill the standard 24-line screen before being swapped out. If the BUFERSIZE value were at its default of 132, then 363 buffers ($48000 \div 132$) would be available (assuming that no TPUTs were greater than the 132).

If the HIBFREXT value is too high, then virtual storage could be wasted. However, keep in mind that this is virtual storage and virtual storage is rarely a constraint to an individual TSO user.

CPEXpert suggests that you consider increasing the HIBFREXT value to accommodate a **significant** number of terminal output buffers. Many installations increase the HIBFREXT value to 96,000 bytes or higher. Recall that this specification is **virtual storage**, and virtual storage will not be a constraint for many TSO address spaces. As described above, significant performance degradation can result from specifying too low a value for HIBFREXT.

-
- LOBFREXT specifies the minimum number of virtual storage bytes that can be dynamically allocated for output data. When this value is reached, tasks that are swapped out for Terminal Output Wait swaps are marked dispatchable. The default value for LOBFREXT in MVS/XA and MVS/ESA is 24000 bytes.

If the LOBFREXT value is too high, a user swapped out for Terminal Output Wait swap will be marked ready and swapped in after only a few buffers have been emptied.

For example, suppose that the BUFRSIZE value were specified as 2048 (as mentioned above), and that the default HIBFREXT and LOBFREXT values were used (these are 48000 and 24000 bytes, respectively).

Suppose that a LISTC command generated 50 lines. Each line would require a buffer. As described above, only 23 buffers would fill before the user would be swapped out for Terminal Output Wait. The user would be marked Ready and swapped back in after only 12 buffers were emptied ($48000 - (12 * 2048) < 24000$). These 12 buffers would fill and the user would again be swapped out. The user would experience a total of 3 Terminal Output Wait swaps in order to generate the 50 lines (initial 23 lines followed by swap out, 12 more lines followed by swap out, 12 more lines followed by swap out)!

As another example, the Broadcast data set often will send many lines to a user during logon, and these are sent in line mode. Each line will require a buffer and the number of buffers could easily be exhausted. With inappropriate specification for HIBFREXT and LOBFREXT values, a TSO user could experience **several** Terminal Output Wait swaps during logon.

The LOBFREXT value should be small to minimize the repeated swapping of users. If LOBFREXT is too high, swapped out address spaces are marked ready and are swapped in, only to fill a few output buffers (this was illustrated in the above discussion). Swap delay time (time from when a swapped out user is marked ready until the user is swapped in) generally is fairly short. This particularly is true for interactive TSO users who probably are logically swapped rather than physically swapped.

The swap-in delay should be **extremely** short, as the Workload Manager will manage the MPL constraints of the domain serving the TSO service class and normally will ensure that TSO users are not delayed by swap-in. Few users should be intolerant of such short delays. In fact, the only reason to have a LOBFREXT greater than

zero is to reduce the delay from the time a user is finished with one screen of output until the user receives the next screen of output.

Consider adjusting the LOBFREXT value to accommodate only 1 or 2 buffers. If expanded storage is installed, swap-in delay should be significantly less (the Terminal Output Wait swaps will almost certainly be directed to expanded storage). Therefore, consider specifying a LOBFREXT value of zero with expanded storage.

In summary, CPEXpert suggests that you consider specifying the following values for the TSOKEYxx keywords for MVS/ESA environments:

```
BUFSIZE = 2048
HIBFREXT = 96000
LOBFREXT = 0
```

These values should, of course, be altered for special circumstances.

- The suggested BUFSIZE value generally should be set to a size sufficient to accommodate the buffer size requirements of 80% to 90% of your TPUTs.
- The HIBFREXT value could be significantly increased (e.g., doubled) if Terminal Output Wait swaps are still a significant percent of terminal wait swaps.
- The LOBFREXT value should be increased to either the SCRSIZE value or the BUFSIZE value if there are unacceptable delays to a terminal user.

Some systems programmers are reluctant to significantly increase the TSOKEYxx buffer values. A part of this reluctance may be caused by remembering earlier systems when real and virtual storage were significant constraints, and when the TSO user community was quite small.

Even if real storage should be a constraint, it generally is better to incur some modest increased paging if the savings is a reduction in swapping. Swaps are **much** more expensive than paging operations.

- **Verify that TCAS is non-swappable.** The default Program Properties Table (module IEFSDPPT in SYS1.LINKLIB) specifies that TCAS is "privileged" but that it is swappable. This means that TCAS will be swapped when the SRM detects that the address space is in a long wait.

This default is specified because TSO is not used at all sites or is not a high priority at some IBM sites.

If you have a significant number of TSO logons and each logon generates two swaps (the Detected Wait and the Terminal Output Wait), you can save the swaps by making TCAS non-swappable. TCAS will still be subject to normal page stealing, so the storage is not completely reserved to TCAS.

Modify the SCHEDxx member of SYS1.PARMLIB by including the following:

```
PPT  PGNAME(IKTCAS00)
      KEY(6)
      NOCANCEL
      NOSWAP
      PRIV
      SYST
```

The above causes the IKTCAS00 terminal control address space program to be non-swappable.

Note that unless your system is extremely storage-constrained so that logically swapping is not being used often, the TCAS program is only logically swapped. It is using central storage, even though it is swapped out. You are simply incurring the overhead of the Detected Wait swaps and Terminal Output Wait swaps needlessly.

- **Verify VTAM IOBUF pool definitions.** You should check the VTAM **IOBUF** pool definitions to determine whether the values are sufficient to prevent unnecessary expansion of the pool.

Reference: MVS Initialization and Tuning Reference

MVS/ESA SP5.1: Chapter 3.60

MVS/ESA SP5.2: Chapter 3.60

OS/390 (V1R1): Chapter 3.61

OS/390 (V1R2): Chapter 3.61

OS/390 (V1R3): Chapter 3.61

OS/390 (V2R4): Chapter 4.17

OS/390 (V2R5): Chapter 66.4

OS/390 (V2R6): Chapter 67.4

OS/390 (V2R7): Chapter 68.4

OS/390 (V2R8): Chapter 68.4

OS/390 (V2R9): Chapter 68.4

OS/390 (V2R10): Chapter 67.4
z/OS (V1R1): Chapter 67.4
z/OS (V1R2): Chapter 67.4
z/OS (V1R3): Chapter 68.4
z/OS (V1R4): Chapter 68.4

VTAM Installation and Resource Definition, pages 294 to 300

MVS Performance Notebook, page 246

VTAM Installation and Resource Definition, pages 294 to 300

MVS Performance Notebook, page 246