HP OpenView VantagePoint Performance Agent

Dictionary of Operating System Performance Metrics

Contents

Introduction

This dictionary contains definitions of the MeasureWare Agent performance metrics on the following platforms:

- IBM AIX
- DEC
- HP-UX
- NCR
- SunOS
- Sinix
- Windows NT
- Windows 3.11/95

The metric names are listed first, grouped by operating system and data class. Use these metric names for exporting data with the MeasureWare Agent Extract utility. You can also use this list of metric names in determining the metrics you can use in defining alarm conditions in your MeasureWare Agent alarmdef file.

The metric definitions are listed next, in alphabetical order.

The glossary of performance metric terms is included at the end of this document.

Metric Data Classes by Operating System

AIX Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_BLOCKED_IO_QUEUE

GBL_COMPLETED_PROC

GBL_CPU_HISTOGRAM

GBL_CPU_IDLE_TIME

GBL_CPU_IDLE_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_CPU_WAIT_TIME

GBL_CPU_WAIT_UTIL

GBL_DISK_BLOCK_IO

GBL_DISK_BLOCK_IO_RATE

GBL_DISK_BLOCK_READ

GBL_DISK_BLOCK_READ_RATE

GBL_DISK_BLOCK_WRITE

GBL_DISK_BLOCK_WRITE_RATE

GBL_DISK_HISTOGRAM

GBL_DISK_PHYS_BYTE

GBL_DISK_PHYS_BYTE_RATE
GBL_DISK_PHYS_IO
GBL_DISK_PHYS_IO_RATE
GBL_DISK_PHYS_READ
GBL_DISK_PHYS_READ_BYTE_RATE
GBL_DISK_PHYS_READ_RATE
GBL_DISK_PHYS_WRITE
GBL_DISK_PHYS_WRITE_BYTE_RATE
GBL_DISK_PHYS_WRITE_RATE
GBL_DISK_RAW_IO
GBL_DISK_RAW_IO_RATE
GBL_DISK_RAW_READ
GBL_DISK_RAW_READ_RATE
GBL_DISK_RAW_WRITE
GBL_DISK_RAW_WRITE_RATE
GBL_DISK_TIME_PEAK
GBL_DISK_UTIL_PEAK
GBL_DISK_VM_IO
GBL_DISK_VM_IO_RATE
GBL_DISK_VM_READ
GBL_DISK_VM_READ_RATE
GBL_DISK_VM_WRITE
GBL_DISK_VM_WRITE_RATE
GBL_FS_SPACE_UTIL_PEAK
GBL_MEM_CACHE_HIT_PCT
GBL_MEM_FREE_UTIL
GBL_MEM_PAGEOUT
GBL_MEM_PAGEOUT_RATE
GBL_MEM_PAGE_REQUEST
GBL_MEM_PAGE_REQUEST_RATE
GBL_MEM_SYS_AND_CACHE_UTIL
GBL_MEM_USER_UTIL
GBL_MEM_UTIL
GBL_NET_COLLISION_1_MIN_RATE
GBL_NET_ERROR_1_MIN_RATE
GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NFS_CALL

GBL_NFS_CALL_RATE

GBL_NUM_NETWORK

GBL_NUM_USER

GBL_OTHER_QUEUE

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_STARTED_PROC

GBL_SUSPENDED_PROCS

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

AIX Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_DISK_BLOCK_IO

APP_DISK_BLOCK_IO_RATE

APP_DISK_BLOCK_READ

APP_DISK_BLOCK_READ_RATE

APP_DISK_BLOCK_WRITE

APP_DISK_BLOCK_WRITE_RATE

APP_DISK_PHYS_IO

APP_DISK_PHYS_IO_RATE

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_MAJOR_FAULT_RATE

APP_MEM_UTIL

APP_MEM_VIRT

APP_MINOR_FAULT_RATE

APP_NAME

APP_NUM

APP_PRI

APP_PRI_STD_DEV

APP_PROC_RUN_TIME

APP_SAMPLE

APP_SUSPENDED_PROCS

AIX Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

PROC_APP_ID

PROC_CPU_SYS_MODE_TIME

PROC_CPU_SYS_MODE_UTIL

PROC_CPU_TOTAL_TIME

PROC_CPU_TOTAL_TIME_CUM

PROC_CPU_TOTAL_UTIL

PROC_CPU_TOTAL_UTIL_CUM

PROC_CPU_USER_MODE_TIME

PROC_CPU_USER_MODE_UTIL

PROC_DISK_BLOCK_IO

PROC_DISK_BLOCK_IO_CUM

PROC_DISK_BLOCK_IO_RATE

PROC_DISK_BLOCK_IO_RATE_CUM

PROC_DISK_BLOCK_READ

PROC_DISK_BLOCK_READ_RATE

PROC_DISK_BLOCK_WRITE

PROC_DISK_BLOCK_WRITE_RATE

PROC_GROUP_ID

PROC_INTEREST

PROC_INTERVAL_ALIVE

PROC_IO_BYTE

PROC_IO_BYTE_CUM

PROC_IO_BYTE_RATE

PROC_IO_BYTE_RATE_CUM

PROC_MAJOR_FAULT

PROC_MEM_RES

PROC_MEM_VIRT

PROC_MINOR_FAULT

PROC_PARENT_PROC_ID

PROC_PRI

PROC_PROC_ID

PROC_PROC_NAME

PROC_RUN_TIME

PROC_STOP_REASON

PROC_TTY

PROC_USER_NAME

AIX Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_APP_NAME

TT_COUNT

TT_FAILED

TT_INFO

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_TRAN_ID

TT_UNAME

TT_WALL_TIME_PER_TRAN

AIX Disk Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYDSK_DEVNAME

BYDSK_HISTOGRAM

BYDSK_PHYS_BYTE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_UTIL

AIX Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_COLLISION

BYNETIF_COLLISION_RATE

BYNETIF_ERROR

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

AIX Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_CPU

GBL_NUM_DISK

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

GBL_THRESHOLD_CPU

GBL_THRESHOLD_DISK

GBL_THRESHOLD_NOKILLED

GBL_THRESHOLD_NONEW

TBL_BUFFER_CACHE_AVAIL

TBL_PROC_TABLE_AVAIL

DEC Global Metrics

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_COMPLETED_PROC

GBL_CPU_NICE_TIME

GBL_CPU_NICE_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_DISK_PHYS_BYTE_RATE

GBL_DISK_PHYS_IO_RATE

GBL_DISK_REQUEST_QUEUE

GBL_DISK_UTIL

GBL_DISK_UTIL_PEAK

GBL_FS_SPACE_UTIL_PEAK

GBL_LOADAVG

GBL_MEM_CACHE

GBL_MEM_CACHE_UTIL

GBL_MEM_FREE

GBL_MEM_FREE_UTIL

GBL_MEM_PAGEIN_BYTE_RATE

GBL_MEM_PAGEOUT

GBL_MEM_PAGEOUT_BYTE_RATE

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PAGE_FAULT_RATE

GBL_MEM_PAGE_REQUEST

GBL_MEM_PAGE_REQUEST_RATE

GBL_MEM_SYS

GBL_MEM_SYS_AND_CACHE_UTIL

GBL_MEM_SYS_UTIL

GBL_MEM_USER

GBL_MEM_USER_REFERENCED_UTIL

GBL_MEM_USER_UNREFERENCED_UTIL

GBL_MEM_USER_UTIL

GBL_MEM_UTIL

GBL_NET_COLLISION_1_MIN_RATE

GBL_NET_ERROR_1_MIN_RATE

GBL_NET_IN_ERROR_PCT

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUTQUEUE

GBL_NET_OUT_ERROR_PCT

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_USED_UTIL

GBL_SWAP_SPACE_UTIL

TBL_FILE_LOCK_USED

TBL_FILE_TABLE_USED

TBL_MSG_TABLE_UTIL

TBL_PROC_TABLE_UTIL

TBL_SEM_TABLE_UTIL

TBL_SHMEM_TABLE_UTIL

DEC Application Metrics

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_MEM_UTIL

APP_MEM_VIRT

APP_NAME

APP_NUM

APP_PRI

APP_PRI_STD_DEV

APP_PROC_RUN_TIME

APP_SAMPLE

DEC Process Metrics

PROC_CPU_SYS_MODE_TIME

PROC_CPU_SYS_MODE_UTIL

PROC_CPU_TOTAL_TIME

PROC_CPU_TOTAL_TIME_CUM

PROC_CPU_TOTAL_UTIL

PROC_CPU_TOTAL_UTIL_CUM

PROC_CPU_USER_MODE_TIME

PROC_CPU_USER_MODE_UTIL

PROC_GROUP_ID

PROC_INTERVAL_ALIVE

PROC_IO_BYTE

PROC_IO_BYTE_CUM

PROC_IO_BYTE_RATE

PROC_IO_BYTE_RATE_CUM

PROC_MEM_RES

PROC_MEM_VIRT

PROC_PARENT_PROC_ID

PROC_PROC_ID

PROC_PROC_NAME

PROC_RUN_TIME

PROC_STOP_REASON

PROC_TTY

PROC_USER_NAME

DEC Disk Metrics

BYDSK_CURR_QUEUE_LENGTH

BYDSK_DEVNAME

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_UTIL

DEC Network Interface Metrics

BYNETIF_COLLISION

BYNETIF_COLLISION_1_MIN_RATE

BYNETIF_COLLISION_RATE

BYNETIF_ERROR

BYNETIF_ERROR_1_MIN_RATE

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

BYNETIF_PACKET_RATE

DEC Configuration Metrics

GBL_MACHINE

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_DISK

GBL_NUM_NETWORK

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

TBL_MSG_TABLE_AVAIL

TBL_PROC_TABLE_AVAIL

TBL_SEM_TABLE_AVAIL

TBL_SHMEM_TABLE_AVAIL

HP-UX Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_COMPLETED_PROC

GBL_CPU_CSWITCH_TIME

GBL_CPU_CSWITCH_UTIL

GBL_CPU_HISTOGRAM

GBL_CPU_IDLE_TIME

GBL_CPU_IDLE_UTIL

GBL_CPU_INTERRUPT_TIME

GBL_CPU_INTERRUPT_UTIL

GBL_CPU_NICE_TIME

GBL_CPU_NICE_UTIL

GBL_CPU_NORMAL_TIME

GBL_CPU_NORMAL_UTIL

GBL_CPU_REALTIME_TIME

GBL_CPU_REALTIME_UTIL

GBL_CPU_SYSCALL_TIME

GBL_CPU_SYSCALL_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_DISK_FS_IO

GBL_DISK_FS_IO_RATE

GBL_DISK_FS_READ

GBL_DISK_FS_READ_RATE

GBL_DISK_FS_WRITE

GBL_DISK_FS_WRITE_RATE

GBL_DISK_HISTOGRAM

GBL_DISK_LOGL_IO

GBL_DISK_LOGL_IO_RATE

GBL_DISK_LOGL_READ

GBL_DISK_LOGL_READ_BYTE

GBL_DISK_LOGL_READ_BYTE_RATE

GBL_DISK_LOGL_READ_RATE

GBL_DISK_LOGL_WRITE

GBL_DISK_LOGL_WRITE_BYTE

GBL_DISK_LOGL_WRITE_BYTE_RATE

GBL_DISK_LOGL_WRITE_RATE

GBL_DISK_PHYS_BYTE

GBL_DISK_PHYS_BYTE_RATE

GBL_DISK_PHYS_IO

GBL_DISK_PHYS_IO_RATE

GBL_DISK_PHYS_READ

GBL_DISK_PHYS_READ_BYTE_RATE

GBL_DISK_PHYS_READ_RATE

GBL_DISK_PHYS_WRITE

GBL_DISK_PHYS_WRITE_BYTE_RATE

GBL_DISK_PHYS_WRITE_RATE

GBL_DISK_RAW_IO

GBL_DISK_RAW_IO_RATE

GBL_DISK_RAW_READ

GBL_DISK_RAW_READ_RATE

GBL_DISK_RAW_WRITE

GBL_DISK_RAW_WRITE_RATE

GBL_DISK_SUBSYSTEM_QUEUE

GBL_DISK_SYSTEM_IO

GBL_DISK_SYSTEM_IO_RATE
GBL_DISK_SYSTEM_READ
GBL_DISK_SYSTEM_READ_RATE
GBL_DISK_SYSTEM_WRITE
GBL_DISK_SYSTEM_WRITE_RATE
GBL_DISK_TIME_PEAK
GBL_DISK_UTIL_PEAK
GBL_DISK_VM_IO
GBL_DISK_VM_IO_RATE
GBL_DISK_VM_READ
GBL_DISK_VM_READ_RATE
GBL_DISK_VM_WRITE
GBL_DISK_VM_WRITE_RATE
GBL_FS_SPACE_UTIL_PEAK
GBL_IPC_SUBSYSTEM_QUEUE
GBL_LOST_MI_TRACE_BUFFERS
GBL_MEM_ACTIVE_VIRT_UTIL
GBL_MEM_CACHE_HIT_PCT
GBL_MEM_FREE_UTIL
GBL_MEM_PAGEOUT
GBL_MEM_PAGEOUT_RATE
GBL_MEM_PAGE_REQUEST
GBL_MEM_PAGE_REQUEST_RATE
GBL_MEM_QUEUE
GBL_MEM_SWAP
GBL_MEM_SWAPOUT_RATE
GBL_MEM_SWAP_1_HR_RATE
GBL_MEM_SYS_AND_CACHE_UTIL
GBL_MEM_SYS_UTIL
GBL_MEM_USER_UTIL
GBL_MEM_UTIL
GBL_NETWORK_SUBSYSTEM_QUEUE
GBL_NET_COLLISION_1_MIN_RATE
GBL_NET_COLLISION_PCT
GBL_NET_ERROR_1_MIN_RATE
GBL_NET_IN_ERROR_PCT

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUTQUEUE

GBL_NET_OUT_ERROR_PCT

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NFS_CALL

GBL_NFS_CALL_RATE

GBL_NUM_DISK

GBL_NUM_NETWORK

GBL_NUM_USER

GBL_OTHER_QUEUE

GBL_PRI_QUEUE

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_QUEUE_HISTOGRAM

GBL_RUN_QUEUE

GBL_SLEEP_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

GBL_SYSTEM_UPTIME_HOURS

GBL_TERM_IO_QUEUE

GBL_TT_OVERFLOW_COUNT

TBL_BUFFER_CACHE_USED

TBL_FILE_LOCK_UTIL

TBL_FILE_TABLE_UTIL

TBL_INODE_CACHE_USED

TBL_MSG_TABLE_UTIL

TBL_PROC_TABLE_UTIL

TBL_SEM_TABLE_UTIL

TBL_SHMEM_TABLE_UTIL

HP-UX Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_NICE_TIME

APP_CPU_NICE_UTIL

APP_CPU_NORMAL_TIME

APP_CPU_NORMAL_UTIL

APP_CPU_REALTIME_TIME

APP_CPU_REALTIME_UTIL

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_DISK_FS_IO

APP_DISK_FS_IO_RATE

APP_DISK_LOGL_IO

APP_DISK_LOGL_IO_RATE

APP_DISK_LOGL_READ

APP_DISK_LOGL_READ_RATE

APP_DISK_LOGL_WRITE

APP_DISK_LOGL_WRITE_RATE

APP_DISK_PHYS_IO

APP_DISK_PHYS_IO_RATE

APP_DISK_PHYS_READ

APP_DISK_PHYS_READ_RATE

APP_DISK_PHYS_WRITE

APP_DISK_PHYS_WRITE_RATE

APP_DISK_RAW_IO

APP_DISK_RAW_IO_RATE

APP_DISK_SUBSYSTEM_WAIT_PCT

APP_DISK_SYSTEM_IO

APP_DISK_SYSTEM_IO_RATE

APP_DISK_VM_IO

APP_DISK_VM_IO_RATE

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_IPC_SUBSYSTEM_WAIT_PCT

APP_MEM_RES

APP_MEM_VIRT

APP_MEM_WAIT_PCT

APP_NAME

APP_NETWORK_SUBSYSTEM_WAIT_PCT

APP_NUM

APP_OTHER_IO_WAIT_PCT

APP_PRI

APP_PRI_STD_DEV

APP_PRI_WAIT_PCT

APP_PROC_RUN_TIME

APP_SAMPLE

APP_SEM_WAIT_PCT

APP_SLEEP_WAIT_PCT

APP_TERM_IO_WAIT_PCT

HP-UX Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

PROC_APP_ID

PROC_CPU_CSWITCH_TIME

PROC_CPU_CSWITCH_UTIL

PROC_CPU_INTERRUPT_TIME
PROC_CPU_INTERRUPT_UTIL
PROC_CPU_NICE_TIME
PROC_CPU_NICE_UTIL
PROC_CPU_NORMAL_TIME
PROC_CPU_NORMAL_UTIL
PROC_CPU_REALTIME_TIME
PROC_CPU_REALTIME_UTIL
PROC_CPU_SYSCALL_TIME
PROC_CPU_SYSCALL_UTIL
PROC_CPU_SYS_MODE_TIME
PROC_CPU_SYS_MODE_UTIL
PROC_CPU_TOTAL_TIME
PROC_CPU_TOTAL_TIME_CUM
PROC_CPU_TOTAL_UTIL
PROC_CPU_TOTAL_UTIL_CUM
PROC_CPU_USER_MODE_TIME
PROC_CPU_USER_MODE_UTIL
PROC_DISK_FS_IO
PROC_DISK_FS_IO_RATE
PROC_DISK_FS_READ
PROC_DISK_FS_READ_RATE
PROC_DISK_FS_WRITE
PROC_DISK_FS_WRITE_RATE
PROC_DISK_LOGL_IO_CUM
PROC_DISK_LOGL_IO_RATE_CUM
PROC_DISK_LOGL_READ
PROC_DISK_LOGL_READ_RATE
PROC_DISK_LOGL_WRITE
PROC_DISK_LOGL_WRITE_RATE
PROC_DISK_PHYS_IO
PROC_DISK_PHYS_IO_CUM
PROC_DISK_PHYS_IO_RATE
PROC_DISK_PHYS_IO_RATE_CUM
PROC_DISK_SUBSYSTEM_WAIT_PCT
PROC_DISK_SUBSYSTEM_WAIT_TIME

PROC_DISK_SYSTEM_IO

PROC_DISK_SYSTEM_IO_RATE

PROC_DISK_VM_IO

PROC_DISK_VM_IO_RATE

PROC_GROUP_ID

PROC_INTEREST

PROC_INTERVAL_ALIVE

PROC_IO_BYTE

PROC_IO_BYTE_CUM

PROC_IO_BYTE_RATE

PROC_IO_BYTE_RATE_CUM

PROC_IPC_SUBSYSTEM_WAIT_PCT

PROC_IPC_SUBSYSTEM_WAIT_TIME

PROC_LAN_WAIT_PCT

PROC_LAN_WAIT_TIME

PROC_MAJOR_FAULT

PROC_MEM_RES

PROC_MEM_VIRT

PROC_MEM_WAIT_PCT

PROC_MEM_WAIT_TIME

PROC_MINOR_FAULT

PROC_NFS_WAIT_PCT

PROC_NFS_WAIT_TIME

PROC_OTHER_IO_WAIT_PCT

PROC_OTHER_IO_WAIT_TIME

PROC_OTHER_WAIT_PCT

PROC_OTHER_WAIT_TIME

PROC_PARENT_PROC_ID

PROC_PRI

PROC_PRI_WAIT_PCT

PROC_PRI_WAIT_TIME

PROC_PROC_ID

PROC_PROC_NAME

PROC_RUN_TIME

PROC_SEM_WAIT_PCT

PROC_SEM_WAIT_TIME

PROC_SLEEP_WAIT_PCT

PROC_SLEEP_WAIT_TIME

PROC_STOP_HISTOGRAM

PROC_STOP_REASON

PROC_SYS_WAIT_PCT

PROC_SYS_WAIT_TIME

PROC_TERM_IO_WAIT_PCT

PROC_TERM_IO_WAIT_TIME

PROC_THREAD_COUNT

PROC_TTY

PROC_USER_NAME

HP-UX Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5
TTBIN_UPPER_RANGE_6
TTBIN_UPPER_RANGE_7
TTBIN_UPPER_RANGE_8
TTBIN_UPPER_RANGE_9
TT_ABORT
TT_ABORT_WALL_TIME_PER_TRAN
TT_APP_NAME
TT_APP_TRAN_NAME
TT_CLIENT_ADDRESS
TT_CLIENT_ADDRESS_FORMAT
TT_CLIENT_TRAN_ID
TT_COUNT
TT_CPU_TOTAL_TIME_PER_TRAN
TT_DISK_LOGL_IO_PER_TRAN
TT_DISK_PHYS_IO_PER_TRAN
TT_FAILED
TT_INFO
TT_NAME
TT_NUM_BINS
TT_SLO_COUNT
TT_SLO_PERCENT
TT_SLO_THRESHOLD
TT_TERM_TRAN_1_HR_RATE
TT_TRAN_1_MIN_RATE
TT_TRAN_ID
TT_UNAME
TT_USER_MEASUREMENT_AVG
TT_USER_MEASUREMENT_AVG_2
TT_USER_MEASUREMENT_AVG_3
TT_USER_MEASUREMENT_AVG_4
TT_USER_MEASUREMENT_AVG_5
TT_USER_MEASUREMENT_AVG_6
TT_USER_MEASUREMENT_COUNT
TT_USER_MEASUREMENT_COUNT_2
TT_USER_MEASUREMENT_COUNT_3

TT_USER_MEASUREMENT_COUNT_4
TT_USER_MEASUREMENT_COUNT_5
TT_USER_MEASUREMENT_COUNT_6
TT_USER_MEASUREMENT_MAX
TT_USER_MEASUREMENT_MAX_2
TT_USER_MEASUREMENT_MAX_3
TT_USER_MEASUREMENT_MAX_4
TT_USER_MEASUREMENT_MAX_5
TT_USER_MEASUREMENT_MAX_6
TT_USER_MEASUREMENT_MIN
TT_USER_MEASUREMENT_MIN_2
TT_USER_MEASUREMENT_MIN_3
TT_USER_MEASUREMENT_MIN_4
TT_USER_MEASUREMENT_MIN_5
TT_USER_MEASUREMENT_MIN_6
TT_USER_MEASUREMENT_NAME
TT_USER_MEASUREMENT_NAME_2
TT_USER_MEASUREMENT_NAME_3
TT_USER_MEASUREMENT_NAME_4
TT_USER_MEASUREMENT_NAME_5
TT_USER_MEASUREMENT_NAME_6
TT_WALL_TIME_PER_TRAN

HP-UX Disk Metrics

BLANK
DATE
DATE_SECONDS
DAY
INTERVAL
RECORD_TYPE
TIME
YEAR
BYDSK_AVG_SERVICE_TIME
BYDSK_DEVNAME
BYDSK_DIRNAME
BYDSK_FS_READ

BYDSK_FS_READ_RATE

BYDSK_FS_WRITE

BYDSK_FS_WRITE_RATE

BYDSK_HISTOGRAM

BYDSK_ID

BYDSK_LOGL_READ

BYDSK_LOGL_READ_RATE

BYDSK_LOGL_WRITE

BYDSK_LOGL_WRITE_RATE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_RAW_READ

BYDSK_RAW_READ_RATE

BYDSK_RAW_WRITE

BYDSK_RAW_WRITE_RATE

BYDSK_REQUEST_QUEUE

BYDSK_SYSTEM_IO

BYDSK_SYSTEM_IO_RATE

BYDSK_UTIL

BYDSK_VM_IO

BYDSK_VM_IO_RATE

HP-UX Logical Volume Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

FS_DIRNAME

LV_DIRNAME

LV_GROUP_NAME

LV_LOGL_READ

LV_LOGL_WRITE

LV_READ_BYTE_RATE

LV_READ_RATE

LV_SPACE_UTIL

LV_WRITE_BYTE_RATE

LV_WRITE_RATE

HP-UX Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_COLLISION

BYNETIF_COLLISION_RATE

BYNETIF_ERROR

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

HP-UX Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_MACHINE_MODEL

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_CPU

GBL_OSKERNELTYPE_INT

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

GBL_THRESHOLD_CPU

GBL_THRESHOLD_DISK

GBL_THRESHOLD_NOKILLED

GBL_THRESHOLD_NONEW

GBL_THRESHOLD_SHORTLIVED

GBL_THRESHOLD_WAIT_CPU

GBL_THRESHOLD_WAIT_DISK

GBL_THRESHOLD_WAIT_IMPEDE

GBL_THRESHOLD_WAIT_MEMORY

TBL_BUFFER_CACHE_AVAIL

TBL_FILE_LOCK_AVAIL

TBL_FILE_TABLE_AVAIL

TBL_INODE_CACHE_AVAIL

TBL_MSG_TABLE_AVAIL

TBL_PROC_TABLE_AVAIL

TBL_SEM_TABLE_AVAIL

TBL_SHMEM_TABLE_AVAIL

NCR Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_COMPLETED_PROC

GBL_CPU_HISTOGRAM

GBL_CPU_IDLE_TIME

GBL_CPU_IDLE_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_CPU_WAIT_TIME

GBL_CPU_WAIT_UTIL

GBL_DISK_BLOCK_IO

GBL_DISK_BLOCK_IO_RATE

GBL_DISK_BLOCK_READ

GBL_DISK_BLOCK_READ_RATE

GBL_DISK_BLOCK_WRITE

GBL_DISK_BLOCK_WRITE_RATE

GBL_DISK_HISTOGRAM

GBL_DISK_LOGL_IO

GBL_DISK_LOGL_IO_RATE

GBL_DISK_LOGL_READ

GBL_DISK_LOGL_READ_RATE

GBL_DISK_LOGL_WRITE

GBL_DISK_LOGL_WRITE_RATE

GBL_DISK_PHYS_BYTE

GBL_DISK_PHYS_BYTE_RATE

GBL_DISK_PHYS_IO

GBL_DISK_PHYS_IO_RATE

GBL_DISK_PHYS_READ

GBL_DISK_PHYS_READ_BYTE_RATE

GBL_DISK_PHYS_READ_RATE

GBL_DISK_PHYS_WRITE

GBL_DISK_PHYS_WRITE_BYTE_RATE

GBL_DISK_PHYS_WRITE_RATE

GBL_DISK_RAW_IO

GBL_DISK_RAW_IO_RATE

GBL_DISK_RAW_READ

GBL_DISK_RAW_READ_RATE

GBL_DISK_RAW_WRITE

GBL_DISK_RAW_WRITE_RATE

GBL_DISK_TIME_PEAK

GBL_DISK_UTIL_PEAK

GBL_DISK_VM_IO

GBL_DISK_VM_IO_RATE

GBL_DISK_VM_READ

GBL_DISK_VM_READ_RATE

GBL_DISK_VM_WRITE

GBL_DISK_VM_WRITE_RATE

GBL_FS_SPACE_UTIL_PEAK

GBL_MEM_CACHE_HIT_PCT

GBL_MEM_FREE_UTIL

GBL_MEM_PAGEOUT

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PAGE_REQUEST

GBL_MEM_PAGE_REQUEST_RATE

GBL_MEM_SWAPOUT_RATE

GBL_MEM_SYS_AND_CACHE_UTIL

GBL_MEM_USER_UTIL

GBL_MEM_UTIL

GBL_NET_ERROR_1_MIN_RATE

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NFS_CALL

GBL_NFS_CALL_RATE

GBL_NUM_NETWORK

GBL_NUM_USER

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

NCR Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_DISK_BLOCK_IO

APP_DISK_BLOCK_IO_RATE

APP_DISK_BLOCK_READ

APP_DISK_BLOCK_READ_RATE

APP_DISK_BLOCK_WRITE

APP_DISK_BLOCK_WRITE_RATE

APP_DISK_PHYS_IO

APP_DISK_PHYS_IO_RATE

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_MEM_UTIL

APP_MEM_VIRT

APP_NAME

APP_NUM

APP_PRI

APP_PRI_STD_DEV

APP_PROC_RUN_TIME

APP_SAMPLE

NCR Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

PROC_APP_ID

PROC_CPU_SYS_MODE_TIME

PROC_CPU_SYS_MODE_UTIL

PROC_CPU_TOTAL_TIME

PROC_CPU_TOTAL_TIME_CUM

PROC_CPU_TOTAL_UTIL

PROC_CPU_TOTAL_UTIL_CUM

PROC_CPU_USER_MODE_TIME

PROC_CPU_USER_MODE_UTIL
PROC_DISK_BLOCK_IO
PROC_DISK_BLOCK_IO_CUM
PROC_DISK_BLOCK_IO_RATE
PROC_DISK_BLOCK_IO_RATE_CUM
PROC_DISK_BLOCK_READ
PROC_DISK_BLOCK_READ_RATE
PROC_DISK_BLOCK_WRITE
PROC_DISK_BLOCK_WRITE_RATE
PROC_GROUP_ID
PROC_INTEREST
PROC_INTERVAL_ALIVE
PROC_IO_BYTE
PROC_IO_BYTE_CUM
PROC_IO_BYTE_RATE
PROC_IO_BYTE_RATE_CUM
PROC_MEM_RES
PROC_MEM_VIRT
PROC_PARENT_PROC_ID
PROC_PRI
PROC_PROC_ID
PROC_PROC_NAME
PROC_RUN_TIME
PROC_STOP_REASON
PROC_TTY
PROC_USER_NAME

NCR Transaction Metrics

BLANK
DATE
DATE_SECONDS
DAY
INTERVAL
RECORD_TYPE
TIME
YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_COUNT

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_WALL_TIME_PER_TRAN

NCR Disk Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYDSK_DEVNAME

BYDSK_HISTOGRAM

BYDSK_PHYS_BYTE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_UTIL

NCR Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_ERROR

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

NCR Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_CPU

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_DISK

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

GBL_THRESHOLD_CPU

GBL_THRESHOLD_DISK

GBL_THRESHOLD_NOKILLED

GBL_THRESHOLD_NONEW

TBL_BUFFER_CACHE_AVAIL

TBL_PROC_TABLE_AVAIL

SunOS Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE
TIME
YEAR
GBL_ACTIVE_CPU
GBL_ACTIVE_PROC
GBL_ALIVE_PROC
GBL_BLOCKED_IO_QUEUE
GBL_COMPLETED_PROC
GBL_CPU_HISTOGRAM
GBL_CPU_IDLE_TIME
GBL_CPU_IDLE_UTIL
GBL_CPU_NICE_TIME
GBL_CPU_NICE_UTIL
GBL_CPU_SYS_MODE_TIME
GBL_CPU_SYS_MODE_UTIL
GBL_CPU_TOTAL_TIME
GBL_CPU_TOTAL_UTIL
GBL_CPU_USER_MODE_TIME
GBL_CPU_USER_MODE_UTIL
GBL_CPU_WAIT_TIME
GBL_CPU_WAIT_UTIL
GBL_DISK_BLOCK_IO
GBL_DISK_BLOCK_IO_RATE
GBL_DISK_BLOCK_READ
GBL_DISK_BLOCK_READ_RATE
GBL_DISK_BLOCK_WRITE
GBL_DISK_BLOCK_WRITE_RATE
GBL_DISK_HISTOGRAM
GBL_DISK_PHYS_BYTE
GBL_DISK_PHYS_BYTE_RATE
GBL_DISK_PHYS_IO
GBL_DISK_PHYS_IO_RATE
GBL_DISK_PHYS_READ
GBL_DISK_PHYS_READ_BYTE_RATE
GBL_DISK_PHYS_READ_RATE
GBL_DISK_PHYS_WRITE

GBL_DISK_PHYS_WRITE_BYTE_RATE

GBL_DISK_PHYS_WRITE_RATE

GBL_DISK_RAW_IO

GBL_DISK_RAW_IO_RATE

GBL_DISK_RAW_READ

GBL_DISK_RAW_READ_RATE

GBL_DISK_RAW_WRITE

GBL_DISK_RAW_WRITE_RATE

GBL_DISK_TIME_PEAK

GBL_DISK_UTIL_PEAK

GBL_DISK_VM_IO

GBL_DISK_VM_IO_RATE

GBL_FS_SPACE_UTIL_PEAK

GBL_LOST_MI_TRACE_BUFFERS

GBL_MEM_CACHE_HIT_PCT

GBL_MEM_FREE_UTIL

GBL_MEM_PAGEOUT

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PAGE_REQUEST

GBL_MEM_PAGE_REQUEST_RATE

GBL_MEM_PG_SCAN_RATE

GBL_MEM_SWAP

GBL_MEM_SWAP_1_HR_RATE

GBL_MEM_SYS_AND_CACHE_UTIL

GBL_MEM_SYS_UTIL

GBL_MEM_USER_UTIL

GBL_MEM_UTIL

GBL_NET_COLLISION_1_MIN_RATE

GBL_NET_ERROR_1_MIN_RATE

GBL_NET_IN_ERROR_PCT

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUT_ERROR_PCT

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NFS_CALL

GBL_NFS_CALL_RATE

GBL_NUM_DISK

GBL_NUM_NETWORK

GBL_NUM_USER

GBL_OTHER_QUEUE

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_SLEEP_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

GBL_SYSTEM_UPTIME_HOURS

GBL_TT_OVERFLOW_COUNT

TBL_FILE_LOCK_USED

TBL_FILE_TABLE_UTIL

TBL_INODE_CACHE_USED

TBL_MSG_TABLE_UTIL

TBL_PROC_TABLE_UTIL

TBL_SEM_TABLE_UTIL

TBL_SHMEM_TABLE_UTIL

SunOS Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_DISK_BLOCK_IO

APP_DISK_BLOCK_IO_RATE

APP_DISK_BLOCK_READ

APP_DISK_BLOCK_READ_RATE

APP_DISK_BLOCK_WRITE

APP_DISK_BLOCK_WRITE_RATE

APP_DISK_PHYS_IO

APP_DISK_PHYS_IO_RATE

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_MAJOR_FAULT_RATE

APP_MEM_UTIL

APP_MEM_VIRT

APP_MINOR_FAULT_RATE

APP_NAME

APP_NUM

APP_PRI

APP_PRI_STD_DEV

APP_PROC_RUN_TIME

APP_REVERSE_PRI

APP_REV_PRI_STD_DEV

APP_SAMPLE

SunOS Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR
PROC_APP_ID
PROC_CPU_SYS_MODE_TIME
PROC_CPU_SYS_MODE_UTIL
PROC_CPU_TOTAL_TIME
PROC_CPU_TOTAL_TIME_CUM
PROC_CPU_TOTAL_UTIL
PROC_CPU_TOTAL_UTIL_CUM
PROC_CPU_USER_MODE_TIME
PROC_CPU_USER_MODE_UTIL
PROC_DISK_BLOCK_IO
PROC_DISK_BLOCK_IO_CUM
PROC_DISK_BLOCK_IO_RATE
PROC_DISK_BLOCK_IO_RATE_CUM
PROC_DISK_BLOCK_READ
PROC_DISK_BLOCK_READ_RATE
PROC_DISK_BLOCK_WRITE
PROC_DISK_BLOCK_WRITE_RATE
PROC_GROUP_ID
PROC_INTEREST
PROC_INTERVAL_ALIVE
PROC_IO_BYTE
PROC_IO_BYTE_CUM
PROC_IO_BYTE_RATE
PROC_IO_BYTE_RATE_CUM
PROC_MAJOR_FAULT
PROC_MEM_RES
PROC_MEM_VIRT
PROC_MINOR_FAULT
PROC_PARENT_PROC_ID
PROC_PRI
PROC_PROC_ID
PROC_PROC_NAME
PROC_REVERSE_PRI
PROC_RUN_TIME
PROC_STOP_REASON

PROC_TTY

PROC_USER_NAME

SunOS Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_APP_NAME

TT_CLIENT_ADDRESS

TT_CLIENT_ADDRESS_FORMAT

TT_CLIENT_TRAN_ID

TT_COUNT

TT_FAILED

TT_INFO

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_TRAN_ID

TT_UNAME

TT_USER_MEASUREMENT_AVG

TT_USER_MEASUREMENT_AVG_2

TT_USER_MEASUREMENT_MAX

TT_USER_MEASUREMENT_MAX_2

TT_USER_MEASUREMENT_MIN

TT_USER_MEASUREMENT_MIN_2

TT_USER_MEASUREMENT_NAME

TT_USER_MEASUREMENT_NAME_2

TT_WALL_TIME_PER_TRAN

SunOS Disk Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYDSK_AVG_SERVICE_TIME

BYDSK_DEVNAME

BYDSK_HISTOGRAM

BYDSK_PHYS_BYTE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_UTIL

SunOS Logical Volume Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

LV_DEVNAME_ALIAS

LV_DIRNAME_ALIAS

LV_READ_RATE

LV_SPACE_UTIL

LV_WRITE_RATE

SunOS Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_COLLISION

BYNETIF_COLLISION_RATE

BYNETIF_ERROR

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

SunOS Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_CPU

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SUBPROCSAMPLEINTERVAL

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

GBL_THRESHOLD_CPU

GBL_THRESHOLD_DISK

GBL_THRESHOLD_NOKILLED

GBL_THRESHOLD_NONEW

TBL_BUFFER_CACHE_AVAIL

TBL_FILE_TABLE_AVAIL

TBL_INODE_CACHE_AVAIL

TBL_MSG_TABLE_AVAIL

TBL_PROC_TABLE_AVAIL

TBL_SEM_TABLE_AVAIL

TBL_SHMEM_TABLE_AVAIL

Sinix Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_CPU

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_COMPLETED_PROC

GBL_CPU_HISTOGRAM

GBL_CPU_IDLE_TIME

GBL_CPU_IDLE_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_CPU_WAIT_TIME

GBL_CPU_WAIT_UTIL

GBL_DISK_BLOCK_IO

GBL_DISK_BLOCK_IO_RATE

GBL_DISK_BLOCK_READ

GBL_DISK_BLOCK_READ_RATE
GBL_DISK_BLOCK_WRITE
GBL_DISK_BLOCK_WRITE_RATE
GBL_DISK_HISTOGRAM
GBL_DISK_LOGL_IO
GBL_DISK_LOGL_IO_RATE
GBL_DISK_LOGL_READ
GBL_DISK_LOGL_READ_RATE
GBL_DISK_LOGL_WRITE
GBL_DISK_LOGL_WRITE_RATE
GBL_DISK_PHYS_BYTE
GBL_DISK_PHYS_BYTE_RATE
GBL_DISK_PHYS_IO
GBL_DISK_PHYS_IO_RATE
GBL_DISK_PHYS_READ
GBL_DISK_PHYS_READ_BYTE_RATE
GBL_DISK_PHYS_READ_RATE
GBL_DISK_PHYS_WRITE
GBL_DISK_PHYS_WRITE_BYTE_RATE
GBL_DISK_PHYS_WRITE_RATE
GBL_DISK_RAW_IO
GBL_DISK_RAW_IO_RATE
GBL_DISK_RAW_READ
GBL_DISK_RAW_READ_RATE
GBL_DISK_RAW_WRITE
GBL_DISK_RAW_WRITE_RATE
GBL_DISK_TIME_PEAK
GBL_DISK_UTIL_PEAK
GBL_DISK_VM_IO
GBL_DISK_VM_IO_RATE
GBL_DISK_VM_READ
GBL_DISK_VM_READ_RATE
GBL_DISK_VM_WRITE
GBL_DISK_VM_WRITE_RATE
GBL_FS_SPACE_UTIL_PEAK
GBL_LOST_MI_TRACE_BUFFERS

GBL_MEM_CACHE_HIT_PCT

GBL_MEM_FREE_UTIL

GBL_MEM_PAGEOUT

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PAGE_REQUEST

GBL_MEM_PAGE_REQUEST_RATE

GBL_MEM_SWAP

GBL_MEM_SWAPOUT_RATE

GBL_MEM_SYS_AND_CACHE_UTIL

GBL_MEM_SYS_UTIL

GBL_MEM_USER_UTIL

GBL_MEM_UTIL

GBL_NET_COLLISION_1_MIN_RATE

GBL_NET_ERROR_1_MIN_RATE

GBL_NET_IN_ERROR_PCT

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUT_ERROR_PCT

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NFS_CALL

GBL_NFS_CALL_RATE

GBL_NUM_DISK

GBL_NUM_NETWORK

GBL_NUM_USER

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

GBL_SYSTEM_UPTIME_HOURS

GBL_TT_OVERFLOW_COUNT

TBL_BUFFER_CACHE_USED

TBL_FILE_LOCK_UTIL

TBL_FILE_TABLE_UTIL

TBL_INODE_CACHE_USED

TBL_MSG_TABLE_UTIL

TBL_PROC_TABLE_UTIL

TBL_SEM_TABLE_UTIL

TBL_SHMEM_TABLE_UTIL

Sinix Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_DISK_BLOCK_IO

APP_DISK_BLOCK_IO_RATE

APP_DISK_BLOCK_READ

APP_DISK_BLOCK_READ_RATE

APP_DISK_BLOCK_WRITE

APP_DISK_BLOCK_WRITE_RATE

APP_DISK_PHYS_IO

APP_DISK_PHYS_IO_RATE

APP_IO_BYTE

APP_IO_BYTE_RATE

APP_MEM_UTIL

APP_MEM_VIRT

APP_NAME

APP_NUM

APP_PRI

APP_PRI_STD_DEV

APP_PROC_RUN_TIME

APP_SAMPLE

Sinix Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

PROC_APP_ID

PROC_CPU_SYS_MODE_TIME

PROC_CPU_SYS_MODE_UTIL

PROC_CPU_TOTAL_TIME

PROC_CPU_TOTAL_TIME_CUM

PROC_CPU_TOTAL_UTIL

PROC_CPU_TOTAL_UTIL_CUM

PROC_CPU_USER_MODE_TIME

PROC_CPU_USER_MODE_UTIL

PROC_DISK_BLOCK_IO

PROC_DISK_BLOCK_IO_CUM

PROC_DISK_BLOCK_IO_RATE

PROC_DISK_BLOCK_IO_RATE_CUM

PROC_DISK_BLOCK_READ

PROC_DISK_BLOCK_READ_RATE

PROC_DISK_BLOCK_WRITE

PROC_DISK_BLOCK_WRITE_RATE

PROC_GROUP_ID

PROC_INTEREST

PROC_INTERVAL_ALIVE

PROC_IO_BYTE

PROC_IO_BYTE_CUM

PROC_IO_BYTE_RATE

PROC_IO_BYTE_RATE_CUM

PROC_MEM_RES

PROC_MEM_VIRT

PROC_PARENT_PROC_ID

PROC_PRI

PROC_PROC_ID

PROC_PROC_NAME

PROC_RUN_TIME

PROC_STOP_REASON

PROC_TTY

PROC_USER_NAME

Sinix Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_APP_NAME

TT_CLIENT_ADDRESS

TT_CLIENT_ADDRESS_FORMAT

TT_CLIENT_TRAN_ID

TT_COUNT

TT_FAILED

TT_INFO

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_TRAN_ID

TT_UNAME

TT_USER_MEASUREMENT_AVG

TT_USER_MEASUREMENT_AVG_2

TT_USER_MEASUREMENT_MAX

TT_USER_MEASUREMENT_MIN

TT_USER_MEASUREMENT_NAME

TT_WALL_TIME_PER_TRAN

Sinix Disk Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYDSK_DEVNAME

BYDSK_HISTOGRAM

BYDSK_PHYS_BYTE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_UTIL

Sinix Logical Volume Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

LV_GROUP_NAME

LV_LOGL_READ

LV_LOGL_WRITE

LV_READ_RATE

LV_SPACE_UTIL

LV_WRITE_RATE

Sinix Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_COLLISION

BYNETIF_COLLISION_RATE

BYNETIF_ERROR

BYNETIF_ERROR_RATE

BYNETIF_IN_PACKET

BYNETIF_IN_PACKET_RATE

BYNETIF_NAME

BYNETIF_OUT_PACKET

BYNETIF_OUT_PACKET_RATE

Sinix Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_MEM_AVAIL

GBL_MEM_PHYS

GBL_NUM_CPU

GBL_NUM_LV

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SWAP_SPACE_AVAIL_KB

GBL_SYSTEM_ID

GBL_THRESHOLD_CPU

GBL_THRESHOLD_DISK

GBL_THRESHOLD_NOKILLED

GBL_THRESHOLD_NONEW

TBL_BUFFER_CACHE_AVAIL

TBL_FILE_LOCK_AVAIL

TBL_FILE_TABLE_AVAIL

TBL_INODE_CACHE_AVAIL

TBL_MSG_TABLE_AVAIL

TBL_PROC_TABLE_AVAIL

TBL_SEM_TABLE_AVAIL

TBL_SHMEM_TABLE_AVAIL

WinNT Global Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_ACTIVE_PROC

GBL_ALIVE_PROC

GBL_COMPLETED_PROC

GBL_CPU_HISTOGRAM

GBL_CPU_IDLE_TIME

GBL_CPU_IDLE_UTIL

GBL_CPU_INTERRUPT_TIME

GBL_CPU_INTERRUPT_UTIL

GBL_CPU_SYS_MODE_TIME

GBL_CPU_SYS_MODE_UTIL

GBL_CPU_TOTAL_TIME

GBL_CPU_TOTAL_UTIL

GBL_CPU_USER_MODE_TIME

GBL_CPU_USER_MODE_UTIL

GBL_DISK_CACHE_READ

GBL_DISK_CACHE_READ_RATE

GBL_DISK_HISTOGRAM

GBL_DISK_LOGL_READ

GBL_DISK_LOGL_READ_RATE

GBL_DISK_PHYS_BYTE

GBL_DISK_PHYS_BYTE_RATE

GBL_DISK_PHYS_IO

GBL_DISK_PHYS_IO_RATE

GBL_DISK_PHYS_READ

GBL_DISK_PHYS_READ_BYTE_RATE

GBL_DISK_PHYS_READ_RATE

GBL_DISK_PHYS_WRITE

GBL_DISK_PHYS_WRITE_BYTE_RATE

GBL_DISK_PHYS_WRITE_RATE

GBL_DISK_TIME_PEAK

GBL_DISK_UTIL_PEAK

GBL_FS_SPACE_UTIL_PEAK

GBL_MEM_CACHE_HIT_PCT

GBL_MEM_FREE_UTIL

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PAGE_REQUEST

GBL_MEM_PAGE_REQUEST_RATE

GBL_MEM_SYS_AND_CACHE_UTIL

GBL_MEM_USER_UTIL

GBL_MEM_UTIL

GBL_NET_IN_PACKET

GBL_NET_IN_PACKET_RATE

GBL_NET_OUT_PACKET

GBL_NET_OUT_PACKET_RATE

GBL_NET_PACKET_RATE

GBL_NUM_NETWORK

GBL_PROC_RUN_TIME

GBL_PROC_SAMPLE

GBL_RUN_QUEUE

GBL_STARTED_PROC

GBL_SWAP_SPACE_UTIL

GBL_SYSCALL_RATE

GBL_WEB_CACHE_HIT_PCT

GBL_WEB_CGI_REQUEST_RATE

GBL_WEB_CONNECTION_RATE

GBL_WEB_FILES_RECEIVED_RATE

GBL_WEB_FILES_SENT_RATE

GBL_WEB_FTP_READ_BYTE_RATE

GBL_WEB_FTP_WRITE_BYTE_RATE

GBL_WEB_GET_REQUEST_RATE

GBL_WEB_GOPHER_READ_BYTE_RATE

GBL_WEB_GOPHER_WRITE_BYTE_RATE

GBL_WEB_HEAD_REQUEST_RATE

GBL_WEB_HTTP_READ_BYTE_RATE

GBL_WEB_HTTP_WRITE_BYTE_RATE

GBL_WEB_ISAPI_REQUEST_RATE

GBL_WEB_LOGON_FAILURES

GBL_WEB_NOT_FOUND_ERRORS

GBL_WEB_OTHER_REQUEST_RATE

GBL_WEB_POST_REQUEST_RATE

GBL_WEB_READ_BYTE_RATE

GBL_WEB_WRITE_BYTE_RATE

WinNT Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PROC

APP_ALIVE_PROC

APP_COMPLETED_PROC

APP_CPU_SYS_MODE_TIME

APP_CPU_SYS_MODE_UTIL

APP_CPU_TOTAL_TIME

APP_CPU_TOTAL_UTIL

APP_CPU_USER_MODE_TIME

APP_CPU_USER_MODE_UTIL

APP_MEM_RES

APP_MEM_UTIL

APP_MEM_VIRT

APP_MINOR_FAULT_RATE

APP_NAME

APP_NUM

APP_PRI

APP_PROC_RUN_TIME

APP_SAMPLE

WinNT Process Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

PROC_APP_ID

PROC_CPU_SYS_MODE_TIME

PROC_CPU_SYS_MODE_UTIL

PROC_CPU_TOTAL_TIME

PROC_CPU_TOTAL_TIME_CUM

PROC_CPU_TOTAL_UTIL

PROC_CPU_TOTAL_UTIL_CUM

PROC_CPU_USER_MODE_TIME

PROC_CPU_USER_MODE_UTIL

PROC_INTEREST

PROC_INTERVAL_ALIVE

PROC_MEM_RES

PROC_MEM_VIRT

PROC_MINOR_FAULT

PROC_PRI

PROC_PROC_ID

PROC_PROC_NAME

PROC_RUN_TIME

WinNT Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_APP_NAME

TT_CLIENT_ADDRESS

TT_CLIENT_ADDRESS_FORMAT

TT_CLIENT_TRAN_ID

TT_COUNT

TT_FAILED

TT_INFO

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_TRAN_ID

TT_UNAME

TT_USER_MEASUREMENT_AVG

TT_USER_MEASUREMENT_AVG_2

TT_USER_MEASUREMENT_AVG_3

TT_USER_MEASUREMENT_AVG_4

TT_USER_MEASUREMENT_AVG_5

TT_USER_MEASUREMENT_AVG_6

TT_USER_MEASUREMENT_COUNT

TT_USER_MEASUREMENT_COUNT_2

TT_USER_MEASUREMENT_COUNT_3

TT_USER_MEASUREMENT_COUNT_4

TT_USER_MEASUREMENT_COUNT_5

TT_USER_MEASUREMENT_COUNT_6

TT_USER_MEASUREMENT_MAX

TT_USER_MEASUREMENT_MAX_2

TT_USER_MEASUREMENT_MAX_3

TT_USER_MEASUREMENT_MAX_4

TT_USER_MEASUREMENT_MAX_5

TT_USER_MEASUREMENT_MAX_6

TT_USER_MEASUREMENT_MIN

TT_USER_MEASUREMENT_MIN_2

TT_USER_MEASUREMENT_MIN_3

TT_USER_MEASUREMENT_MIN_4

TT_USER_MEASUREMENT_MIN_5

TT_USER_MEASUREMENT_MIN_6

TT_USER_MEASUREMENT_NAME

TT_USER_MEASUREMENT_NAME_2

TT_USER_MEASUREMENT_NAME_3

TT_USER_MEASUREMENT_NAME_4

TT_USER_MEASUREMENT_NAME_5

TT_USER_MEASUREMENT_NAME_6

TT_WALL_TIME_PER_TRAN

WinNT Disk Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYDSK_DEVNAME

BYDSK_HISTOGRAM

BYDSK_PHYS_BYTE

BYDSK_PHYS_BYTE_RATE

BYDSK_PHYS_IO

BYDSK_PHYS_IO_RATE

BYDSK_PHYS_READ

BYDSK_PHYS_READ_BYTE

BYDSK_PHYS_READ_BYTE_RATE

BYDSK_PHYS_READ_RATE

BYDSK_PHYS_WRITE

BYDSK_PHYS_WRITE_BYTE

BYDSK_PHYS_WRITE_BYTE_RATE

BYDSK_PHYS_WRITE_RATE

BYDSK_UTIL

WinNT Network Interface Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

BYNETIF_NAME

BYPROTOCOL_IN_PACKET

BYPROTOCOL_IN_PACKET_RATE

BYPROTOCOL_OUT_PACKET

BYPROTOCOL_OUT_PACKET_RATE

WinNT Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE
GBL_MEM_AVAIL
GBL_MEM_PHYS
GBL_NUM_CPU
GBL_NUM_DISK
GBL_OSNAME
GBL_OSRELEASE
GBL_OSVERSION
GBL_SWAP_SPACE_AVAIL_KB
GBL_SYSTEM_ID
GBL_THRESHOLD_CPU
GBL_THRESHOLD_NOKILLED
GBL_THRESHOLD_NONEW

Win3x/95 Global Metrics

BLANK
DATE
DATE_SECONDS
DAY
INTERVAL
RECORD_TYPE
TIME
YEAR
GBL_ALIVE_PROC
GBL_ALIVE_THREAD
GBL_CPU_TOTAL_TIME
GBL_CPU_TOTAL_UTIL
GBL_DISK_FS_IO_RATE
GBL_DISK_FS_READ
GBL_DISK_FS_READ_RATE
GBL_DISK_FS_WRITE
GBL_DISK_FS_WRITE_RATE
GBL_DISK_LOGL_IO_RATE
GBL_DISK_SPACE
GBL_DISK_SPACE_FREE
GBL_GUI_DELAY_INDEX

GBL_GUI_INPUT_COUNT

GBL_GUI_INPUT_DELAY

GBL_GUI_INPUT_RATE

GBL_GUI_KEYBOARD_COUNT

GBL_GUI_KEYBOARD_RATE

GBL_GUI_MOUSE_COUNT

GBL_GUI_MOUSE_RATE

GBL_MEM_CACHE_HIT_PCT

GBL_MEM_COMMIT_PCT

GBL_MEM_DISCARD

GBL_MEM_DISCARD_RATE

GBL_MEM_FAULT_RATE

GBL_MEM_GDIRES_FREE

GBL_MEM_LOAD_INDEX

GBL_MEM_PAGEIN

GBL_MEM_PAGEIN_RATE

GBL_MEM_PAGEOUT

GBL_MEM_PAGEOUT_RATE

GBL_MEM_PHYS

GBL_MEM_SYSRES_FREE

GBL_MEM_USERRES_FREE

GBL_NET_BYTE_RATE

GBL_NET_PACKET_RATE

GBL_PARTITION_SPACE_MIN

GBL_RDR_BYTE_RATE

GBL_RDR_REQUEST_RATE

GBL_SVR_BYTE_RATE

GBL_SWAP_SPACE_AVAIL

GBL_SWAP_SPACE_FREE

GBL_SWAP_SPACE_RESERVED

GBL_SWAP_SPACE_USED

TBL_BUFFER_CACHE_AVAIL

Win3x/95 Application Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

APP_ACTIVE_PCT

APP_ACTIVE_TIME

APP_GUI_INPUT_COUNT

APP_GUI_INPUT_RATE

APP_GUI_KEYBOARD_COUNT

APP_GUI_KEYBOARD_DELAY

APP_GUI_KEYBOARD_RATE

APP_GUI_MOUSE_COUNT

APP_GUI_MOUSE_DELAY

APP_GUI_MOUSE_RATE

APP_NAME

Win3x/95 Transaction Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

TTBIN_TRANS_COUNT_1

TTBIN_TRANS_COUNT_10

TTBIN_TRANS_COUNT_2

TTBIN_TRANS_COUNT_3

TTBIN_TRANS_COUNT_4

TTBIN_TRANS_COUNT_5

TTBIN_TRANS_COUNT_6

TTBIN_TRANS_COUNT_7

TTBIN_TRANS_COUNT_8

TTBIN_TRANS_COUNT_9

TTBIN_UPPER_RANGE_1

TTBIN_UPPER_RANGE_10

TTBIN_UPPER_RANGE_2

TTBIN_UPPER_RANGE_3

TTBIN_UPPER_RANGE_4

TTBIN_UPPER_RANGE_5

TTBIN_UPPER_RANGE_6

TTBIN_UPPER_RANGE_7

TTBIN_UPPER_RANGE_8

TTBIN_UPPER_RANGE_9

TT_ABORT

TT_ABORT_WALL_TIME_PER_TRAN

TT_COUNT

TT_NAME

TT_NUM_BINS

TT_SLO_COUNT

TT_SLO_PERCENT

TT_SLO_THRESHOLD

TT_TERM_TRAN_1_HR_RATE

TT_TRAN_1_MIN_RATE

TT_WALL_TIME_PER_TRAN

Win3x/95 Configuration Metrics

BLANK

DATE

DATE_SECONDS

DAY

INTERVAL

RECORD_TYPE

TIME

YEAR

GBL_COLLECTOR

GBL_LOGFILE_VERSION

GBL_LOGGING_TYPES

GBL_MACHINE

GBL_NUM_CPU

GBL_OSNAME

GBL_OSRELEASE

GBL_OSVERSION

GBL_SYSTEM_ID

Metric Definitions

APP_ACTIVE_PCT

PLATFORMS: Win3X/95

The proportion of the interval that this APP_NAME has been active. An application is considered active if it has a window that is in the active state. Applications that do not display windows, or that perform processing activity while their windows are not in the active state are not considered active.

APP_ACTIVE_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

An active process is one that exists and consumes some CPU time. APP_ACTIVE_PROC is the sum of the alive-process-time/interval-time ratios of every process belonging to an application that is active (uses any CPU time) during an interval.

The following diagram of a four second interval showing two processes, A and B, for an application should be used to understand the above definition. Note the difference between active processes, which consume CPU time, and alive processes which merely exist on the system.

```
            ----------- Seconds -----------
               1         2         3      4
     Proc
     ---- ----      ----      ----    ----
     A    live      live      live    live

     B    live/CPU  live/CPU  live    dead
```

Process A is alive for the entire four second interval, but consumes no CPU. A's contribution to APP_ALIVE_PROC is 4*1/4. A contributes 0*1/4 to APP_ACTIVE_PROC. B's contribution to APP_ALIVE_PROC is 3*1/4. B contributes 2*1/4 to APP_ACTIVE_PROC. Thus, for this interval, APP_ACTIVE_PROC equals 0.5 and APP_ALIVE_PROC equals 1.75.

Because a process may be alive but not active, APP_ACTIVE_PROC will always be less than or equal to APP_ALIVE_PROC.

This metric indicates the number of processes in an application group that are competing for the CPU. This metric is useful, along with other metrics, for comparing loads placed on the system by different groups of processes.

SunOS AIX NCR Sinix DEC

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

APP_ACTIVE_TIME

PLATFORMS: Win3X/95

The number of seconds during the interval that this application had windows in the active state.

APP_ALIVE_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

An alive process is one that exists on the system. APP_ALIVE_PROC is the sum of the alive-process-time/interval-time ratios for every process belonging to a given application.

The following diagram of a four second interval showing two processes, A and B, for an application should be used to understand the above definition. Note the difference between active processes, which consume CPU time, and alive processes which merely exist on the system.

```
          ----------- Seconds -----------
            1         2         3      4
    Proc
    ---- ----      ----      ----   ----
    A    live      live      live   live
    B    live/CPU  live/CPU  live   dead
```

Process A is alive for the entire four second interval but consumes no CPU. A's contribution to APP_ALIVE_PROC is 4*1/4. A contributes 0*1/4 to APP_ACTIVE_PROC. B's contribution to APP_ALIVE_PROC is 3*1/4. B contributes 2*1/4 to APP_ACTIVE_PROC. Thus, for this interval, APP_ACTIVE_PROC equals 0.5 and APP_ALIVE_PROC equals 1.75.

Because a process may be alive but not active, APP_ACTIVE_PROC will always be less than or equal to APP_ALIVE_PROC.

SunOS AIX NCR Sinix DEC

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

APP_COMPLETED_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of processes in this group that completed during the interval.

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

APP_CPU_NICE_TIME

PLATFORMS: HP-UX

The time, in seconds, that processes in this group were using the CPU in user mode at a nice priority during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_NICE_UTIL

PLATFORMS: HP-UX

The percentage of time that processes in this group were using the CPU in user mode at a nice priority during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_NORMAL_TIME

PLATFORMS: HP-UX

The time, in seconds, that processes in this group were in user mode at a normal priority during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

APP_CPU_NORMAL_UTIL

PLATFORMS: HP-UX

The percentage of time that processes in this group were in user mode running at normal priority during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_REALTIME_TIME

PLATFORMS: HP-UX

The time, in seconds, that the processes in this group were in user mode at a “realtime” priority during the interval. “Realtime” priority is 0-127.

APP_CPU_REALTIME_UTIL

PLATFORMS: HP-UX

The percentage of time that processes in this group were in user mode at a “realtime” priority during the interval. “Realtime” priority is 0-127.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_SYS_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The time, in seconds, during the interval that the CPU was in system mode for processes in this group.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_SYS_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of time during the interval that the CPU was used in system mode for processes in this group.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

High system CPU utilizations are normal for IO intensive groups. Abnormally high system CPU utilization can indicate that a hardware problem is causing a high interrupt rate. It can also indicate programs that are not making efficient system calls.

APP_CPU_TOTAL_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The total CPU time, in seconds, devoted to processes in this group during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_TOTAL_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of the total CPU time devoted to processes in this group during the interval. This indicates the relative CPU load placed on the system by processes in this group.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

Large values for this metric may indicate that this group is causing a CPU bottleneck. This would be normal in a computation-bound workload, but might mean that processes are using excessive CPU time and perhaps looping.

If the “other” application shows significant amounts of CPU, you may want to consider tuning your parm file so that process activity is accounted for in known applications.

```
APP_CPU_TOTAL_UTIL =
  APP_CPU_SYS_MODE_UTIL +
  APP_CPU_USER_MODE_UTIL
```

WinNT

NOTE: The sum of the APP_CPU_TOTAL_UTIL metrics may not equal GBL_CPU_TOTAL_UTIL. Microsoft states that “this is expected behavior” because the GBL_CPU_TOTAL_UTIL metric is taken from the NT performance library Processor objects while the APP_CPU_TOTAL_UTIL metrics are taken from the Process objects. Microsoft states that there can be CPU time accounted for in the Processor system objects that may not be seen in the Process objects.

APP_CPU_USER_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The time, in seconds, that processes in this group were in user mode during the interval.

HP-UX

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_CPU_USER_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of time that processes in this group were using the CPU in user mode during the interval.

High user mode CPU percentages are normal for computation-intensive groups. Low values of user CPU utilization compared to relatively high values for APP_CPU_SYS_MODE_UTIL can indicate a hardware problem or improperly tuned programs in this group.

HP-UX

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

HP-UX SunOS Sinix DEC

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

APP_DISK_BLOCK_IO

PLATFORMS: SunOS AIX Sinix NCR

The number of block IOs to the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_BLOCK_IO_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block IOs per second to the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_BLOCK_READ

PLATFORMS: SunOS AIX NCR Sinix

The number of block reads from the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical readsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_BLOCK_READ_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block reads per second from the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical readsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_BLOCK_WRITE

PLATFORMS: SunOS AIX NCR Sinix

The number of block writes to the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical writesgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero.

These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_BLOCK_WRITE_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block writes per second from the file system buffer cache for processes in this group during the interval.

SunOS

On SunOS 4.1.X, these are physical writesgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock

updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

APP_DISK_FS_IO

PLATFORMS: HP-UX

The number of file system disk IOs for processes in this group during the interval.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical IOs in this category (they appear under virtual memory IOs).

APP_DISK_FS_IO_RATE

PLATFORMS: HP-UX

The number of file system disk IOs for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical IOs generated by user file system access and do not include virtual memory IOs, system IOs (inode updates), or IOs relating to raw disk access. An exception is user files accessed via the mmap(2) call, which will not show their physical IOs in this category. They appear under virtual memory IOs.

APP_DISK_LOGL_IO

PLATFORMS: HP-UX

The number of logical IOs for processes in this group during the interval.

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_LOGL_IO_RATE

PLATFORMS: HP-UX

The number of logical IOs per second for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_LOGL_READ

PLATFORMS: HP-UX

The number of logical reads for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_LOGL_READ_RATE

PLATFORMS: HP-UX

The number of logical reads per second for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_LOGL_WRITE

PLATFORMS: HP-UX

The number of logical writes for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_LOGL_WRITE_RATE

PLATFORMS: HP-UX

The number of logical writes per second for processes in this group during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

APP_DISK_PHYS_IO

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of physical IOs for processes in this group during the interval.

SunOS

On Sun systems, this metric is only available on Sun 5.X or later.

APP_DISK_PHYS_IO_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of physical IOs per second for processes in this group during the interval.

SunOS AIX

For Sun and IBM AIX systems, this is calculated as

```
APP_DISK_PHYS_IO_RATE =
  APP_DISK_BLOCK_READ_RATE +
  APP_DISK_BLOCK_WRITE_RATE
```

SunOS

On Sun systems, this metric is only available on Sun 5.X or later.

APP_DISK_PHYS_READ

PLATFORMS: HP-UX

The number of physical reads for processes in this group during the interval.

APP_DISK_PHYS_READ_RATE

PLATFORMS: HP-UX

The number of physical reads per second for processes in this group during the interval.

APP_DISK_PHYS_WRITE

PLATFORMS: HP-UX

The number of physical writes for processes in this group during the interval.

APP_DISK_PHYS_WRITE_RATE

PLATFORMS: HP-UX

The number of physical writes per second for processes in this group during the interval.

APP_DISK_RAW_IO

PLATFORMS: HP-UX

The total number of raw IOs for processes in this group during the interval.

Only accesses to local disk devices are counted.

APP_DISK_RAW_IO_RATE

PLATFORMS: HP-UX

The total number of raw IOs for processes in this group during the interval. Only accesses to local disk devices are counted.

APP_DISK_SUBSYSTEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on the disk subsystem (waiting for their file system IOs to complete) during the interval.

This is the sum of processes or kernel threads in the DISK, INODE, CACHE and CDFS wait states. It does not include processes or kernel threads doing raw IO to disk devices.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_DISK_SYSTEM_IO

PLATFORMS: HP-UX

The number of physical IOs generated by the kernel for file system management (inode accesses or updates) for processes in this group during the interval

APP_DISK_SYSTEM_IO_RATE

PLATFORMS: HP-UX

The number of physical IOs per second generated by the kernel for file system management (inode accesses or updates) for processes in this group during the interval.

APP_DISK_VM_IO

PLATFORMS: HP-UX

The number of virtual memory IOs made on behalf of processes in this group during the interval.

IOs to user file data are not included in this metric unless they were done via the mmap(2) system call.

APP_DISK_VM_IO_RATE

PLATFORMS: HP-UX

The number of virtual memory IOs per second made on behalf of processes in this group during the interval.

IOs to user file data are not included in this metric unless they were done via the mmap(2) system call.

APP_GUI_INPUT_COUNT

PLATFORMS: Win3X/95

The number of keyboard strokes and mouse clicks during the time the application was active.

APP_GUI_INPUT_RATE

PLATFORMS: Win3X/95

The number of keyboard strokes and mouse clicks per minute during the time the application was active.

APP_GUI_KEYBOARD_COUNT

PLATFORMS: Win3X/95

The number of keyboard depressions during the time the application was active.

APP_GUI_KEYBOARD_DELAY

PLATFORMS: Win3X/95

The average delay in milliseconds between system receipt of a keyboard message and the delivery of that message to an application.

APP_GUI_KEYBOARD_RATE

PLATFORMS: Win3X/95

The rate per minute of keyboard depressions during the time the application was active.

APP_GUI_MOUSE_COUNT

PLATFORMS: Win3X/95

The number of mouse clicks during the time the application was active.

APP_GUI_MOUSE_DELAY

PLATFORMS: Win3X/95

The average delay in milliseconds between system receipt of a mouse message and the delivery of that message to an application.

APP_GUI_MOUSE_RATE

PLATFORMS: Win3X/95

The rate per minute of mouse clicks during the time the application was active.

APP_IO_BYTE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of characters (in KB) transferred for processes in this group to all devices during the interval. This includes IO to disk, terminal, tape and printers.

SunOS

On Sun systems, this metric is only available on 5.x or later.

APP_IO_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of characters (in KB) per second transferred for processes in this group to all devices during the interval. This includes IO to disk, terminal, tape and printers.

SunOS

On Sun systems, this metric is only available on 5.x or later.

APP_IPC_SUBSYSTEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on the InterProcess Communication (IPC) subsystems (waiting for their interprocess communication activity to complete) during the interval.

This is the sum of processes or kernel threads in the IPC, MSG, SEM, PIPE, SOCKT (that is, sockets) and STRMS (that is, streams IO) wait states.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_MAJOR_FAULT_RATE

PLATFORMS: HP-UX SunOS AIX

The number of major page faults per second that required a disk IO for processes in this group during the interval.

APP_MEM_RES

PLATFORMS: HP-UX WinNT

HP-UX

The size (in KB) of resident memory for processes in this group that were alive at the end of the interval. This consists of text, data, stack, as well as the process' portion of shared memory regions (such as, shared libraries, text segments, and shared data).

For each process, resident memory (RSS) is calculated as

```
RSS = sum of private region pages +
      (sum of shared region pages /
       number of references)
```

The number of references is a count of the number of attachments to the memory region. Attachments, for shared regions, may come from several processes sharing the same memory, a single process with multiple attachments, or combinations of these.

This value is only updated when a process uses CPU. Thus, under memory pressure, this value may be higher than the actual amount of resident memory for processes which are idle.

Refer to the help text for PROC_MEM_RES for additional information.

WinNT

The sum of the size (in KB) of the working sets for processes in this group during the interval. The working set counts memory pages referenced recently by the threads making up this group. Note that the size of the working set is often larger than the amount of pagefile space consumed (APP_MEM_VIRT).

APP_MEM_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX SunOS AIX NCR Sinix DEC

The approximate percentage of the system's physical memory used as resident memory by processes in this group that were alive at the end of the interval.

This metric summarizes process private and shared memory in each application.

HP-UX

This consists of text, data, stack, as well the process' portion of shared memory regions (such as, shared libraries, text segments, and shared data). The sum of the shared region pages is divided by the number of references.

SunOS AIX NCR Sinix DEC

This consists of text, data, stack, as well as an estimate of the process' portion of shared memory.

HP-UX SunOS AIX NCR Sinix DEC

Each application's total resident memory is summed. This value is then divided by the summed total of all applications resident memory and then multiplied by the ratio of available user memory versus total physical memory to arrive at a computed percent of total physical memory.

It must be remembered, however, that this is a computed metric that shows the approximate percentage of the physical memory used as resident memory by the processes in this application during the interval.

HP-UX

This metric is not available for HP-UX MeasureWare Agent. It is available for HP-UX GlancePlus.

WinNT

An estimate of the percentage of the system's physical memory allocated for working set memory by processes in this group during the interval.

The sum of the working set sizes for each process in this group is kept as APP_MEM_RES. This value is divided by the sum of APP_MEM_RES for all applications defined on the system to come up with a ratio of this application's working set size to the total. This value is then multiplied by the ratio of available user memory versus total physical memory to arrive at a computed percent of total physical memory.

APP_MEM_VIRT

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The approximate size (in KB) of virtual memory for processes in this group that were alive at the end of the interval.

HP-UX SunOS Sinix DEC

This is the sum of the virtual memory region sizes for all processes in this group. Since this virtual memory size for each process includes shared regions, such as library text and data, the shared regions are counted multiple times in this metric. For example, if two processes are attached to a 10MB shared region, then 20MB is reported in this metric.

This value is not affected by the reference count. As such, this metric can overestimate the virtual memory being used by processes in this group when they share memory regions.

AIX NCR

This is the sum of the virtual memory region sizes for all processes in this group.

WinNT

The size (in KB) of paging file space used for processes in this group during the interval. This is the sum of the pagefile space used for all processes in this group. Groups of processes may have working set sizes (APP_MEM_RES) larger than the size of their pagefile space.

APP_MEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on memory (waiting for virtual memory disk accesses to complete) during the interval.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_MINOR_FAULT_RATE

PLATFORMS: HP-UX SunOS AIX WinNT

The number of minor page faults per second satisfied in memory (pages were reclaimed from one of the free lists) for processes in this group during the interval.

APP_NAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

The name of the application (up to 20 characters). This comes from the parm file where the applications are defined.

The application called “other” captures all other processes not defined in the parm file.

APP_NETWORK_SUBSYSTEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on the network subsystem (waiting for their network activity to complete) during the interval.

This is the sum of processes or kernel threads in the LAN, NFS, and RPC wait states. This does not include processes or kernel threads blocked on SOCKT (that is, socket) waits, as some processes or kernel threads sit idle in SOCKT waits for long periods.

This is calculated as the accumulated time that all processes or kernel threads in this group spent blocked on (LAN + NFS + RPC) divided by the interval time.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_NUM

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The sequentially assigned number of this application.

APP_OTHER_IO_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on “other IO” during the interval.

"Other IO” includes all IO directed at a device (connected to the local computer) which is not a terminal or LAN. Examples of “other IO” devices are local printers, tapes, instruments, and disks. Time waiting for character (raw) IO to disks is included in this measurement. Time waiting for file system buffered IO to disks will typically been seen as IO or CACHE wait. Time waiting for IO to NFS disks is reported as NFS wait.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT

PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_PRI

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX SunOS AIX NCR Sinix DEC

The average priority of the processes in this group during the interval.

WinNT

The average base priority of the processes in this group during the interval.

SunOS

On Sun systems, this metric is only available on 4.1.X.

APP_PRI_STD_DEV

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The standard deviation of priorities of the processes in this group during the interval.

HP-UX

This metric is available on HP-UX 10.20.

SunOS

On Sun systems, this metric is only available on 4.1.X.

APP_PRI_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on PRI (waiting for their priority to become high enough to get the CPU) during the interval.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_PROC_RUN_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The average run time for processes in this group that completed during the interval.

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

APP_REVERSE_PRI

PLATFORMS: SunOS NCR Sinix

The average priority of the processes in this group during the interval. Lower values for this metric always imply higher processing priority. The range is from 0 to 127. Since priority ranges can be customized on this OS, this metric provides a standardized way of interpreting priority that is consistent with other versions of UNIX. See also the APP_PRI metric.

This is derived from the PRI field of the ps command when the -c option is not used.

SunOS

On Sun systems, this metric is only available on 5.X or later.

APP_REV_PRI_STD_DEV

PLATFORMS: SunOS NCR Sinix

The standard deviation of priorities of the processes in this group during the interval. Priorities are mapped into a traditional lower value implies higher priority scheme.

SunOS

On Sun systems, this metric is only available on 5.X or later.

APP_SAMPLE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of samples of process data that have been averaged or accumulated during this sample.

APP_SEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on semaphores (waiting for their semaphore operations to complete) during the interval.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_SLEEP_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on SLEEP (waiting to awaken from sleep system calls) during the interval.

A process or kernel thread enters the SLEEP state by putting itself to sleep using system calls such as sleep, wait, pause, sigpause, sigsuspend, poll and select.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

The Application QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues, within the context of a specific application.

The Application WAIT PCT metrics, which are also based on block states, represent the percentage of processes or kernel threads that were alive on the system within the context of a specific application. These values will vary greatly depending on the application.

No direct comparison is reasonable with the Global Queue metrics since they represent the average number of all processes or kernel threads that were alive on the system. As such, the Application WAIT PCT metrics cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

APP_SUSPENDED_PROCS

PLATFORMS: AIX

The average number of processes in this group which have been either marked as should be suspended (SGETOUT) or have been suspended (SSWAPPED) during the interval.

Processes are suspended when the OS detects that memory thrashing is occurring. The scheduler looks for processes that have a high repage rate when compared with the number of major page faults the process has done and suspends these processes.

If this metric is not zero, there is a memory bottleneck on the system.

APP_TERM_IO_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time processes or kernel threads in this group were blocked on terminal IO (waiting for terminal IO to complete) during the interval.

A percentage of time spent in a wait state is calculated as the accumulated time kernel threads belonging to processes in this group spent waiting in this state, divided by accumulated alive time of kernel threads belonging to processes in this group during the interval.

For example, assume an application has 20 kernel threads. During the interval, ten kernel threads slept the entire time, while ten kernel threads waited on terminal input. As a result, the application wait percent values would be 50% for SLEEP and 50% for TERM (that is, terminal IO).

This metric is available on HP-UX 10.20.

BLANK

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

An empty field used for spacing reports. For example, this field can be used to create a blank column in a spreadsheet that may be used to sum several items.

BYDSK_AVG_SERVICE_TIME

PLATFORMS: HP-UX SunOS

The average time, in milliseconds, that this disk device spent processing each disk request during the interval. For example, a value of 5.14 would indicate that disk requests during the last interval took on average slightly longer than five one-thousandths of a second to complete for this device.

This is a measure of the speed of the disk, because slower disk devices typically show a larger average service time. Average service time is also dependent on factors such as the distribution of I/O requests over the interval and their locality. It can also be influenced by disk driver and controller features such as I/O merging and command queueing. Note that this service time is measured from the perspective of the kernel, not the disk device itself. For example, if a disk device can find the requested data in its cache, the average service time could be quicker than the speed of the physical disk hardware.

This metric can be used to help determine which disk devices are taking more time than usual to process requests.

BYDSK_CURR_QUEUE_LENGTH

PLATFORMS: SunOS Sinix DEC

The average number of physical IO requests that were in the wait and service queues for this disk device during the interval.

BYDSK_DEVNAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The name identifying the specific disk spindle is the hardware path which specifies the address of the hardware components leading to the disk device.

SunOS

For CDs and disks, this is the device name compliant with the SVR4 Interface Definition and the slice (partition) number is replaced with an asterisk. An example of a device name is “c0t3d0s*”. The naming scheme follows:

```
SVID Disk Name Format
c#t#d#s*
c#  Specifies the controller, where
    “#” is the controller number.
t#  Specifies the target, where “#”
    is the target ID number.
d#  Specifies the device, where “#”
    is the device number.
s*  Represents the slice (or
    partition) in a wildcard format
    since more than one partition
    may be formatted on a device.
    This extends to a complete file
    name which can be looked up in
    the /dev/dsk directory.
```

If the SVID device name cannot be determined, the system name of the device is given. These names are the same disk names displayed by “iostat”. A system name consists of a device type label plus an instance number. For example, “sd2” would be provided for SCSI instance (device) 2. If the device is unrecognizable “Unknown” appears.

The SVID device name is determined by using the information in the /etc/path_to_inst file with the symbolic links in the /dev/dsk directory. See the man page for “disks” if your device labels are not SVID format. For more information about “instances”, see the man page “path_to_inst”.

Device names are determined at the start of collection. When a device is brought online in the middle of collection, its name is determined at that time.

```
Floppy Disk Name Format
  /dev/diskette#
```

Floppy devices are labeled with the device file name link from the /dev directory where “#” specifies a floppy device instance.

(Additional note)

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

AIX

The path name string of this disk device. This is the fsname parameter in the mount(1M) command. If more than one file system is contained on a device (that is, the device is partitioned), this is indicated by an asterisk (“*”) at the end of the path name.

NCR

For disks on the system, this is the device name compliant with the SVR4 Interface Definition with the slice (partition) number replaced by an asterisk. An example of a device name is “c0t3d0s*”. The naming scheme follows:

```
SVID Disk Name Format
c#t#d#s*
c#  Specifies the controller, where
    “#” is the controller number.
t#  Specifies the target, where “#”
    is the target ID number.
d#  Specifies the device, where “#”
    is the device number.
s*  Represents the slice (or
    partition) in a wildcard format
    since more than one partition
    may be formatted on a device.
    This extends to a complete file
    name which can be looked up in
    the /dev/dsk directory.
```

Sinix

For disk spindles on the system, this is the name compliant with SINIX device naming convetions. Names are displayed without leading /dev/ directory and with the slice (partition) number omitted. An example for a disk name is “ios/sdisk015”. For a SCSI disks the naming scheme is as follows:

```
     ios#/sdisk###s*
ios#     Specifies the bus (0 for
           onboard, 1 for other).
sdisk### The three digits that
         follow “sdisk” denote:
         # The number of the SCSI
           host adaptor.
         # The controller number of
           the SCSI host adaptor.
         # The device unit number
           (address).
```

More information on device naming can be found in devname(7) and sdisk(7) man pages.

DEC

The path name string of this disk device. This is the file-system parameter in the mount(1M) command.

WinNT

The unit number of this disk device.

BYDSK_DIRNAME

PLATFORMS: HP-UX SunOS

The name of the file system directory mounted on this disk device. If more than one file system is mounted on this device, “Multiple FS” is seen.

BYDSK_FS_READ

PLATFORMS: HP-UX

The number of physical file system reads from this disk device during the interval.

BYDSK_FS_READ_RATE

PLATFORMS: HP-UX

The number of physical file system reads per second from this disk device during the interval.

BYDSK_FS_WRITE

PLATFORMS: HP-UX

The number of physical file system writes to this disk device during the interval.

BYDSK_FS_WRITE_RATE

PLATFORMS: HP-UX

The number of physical file system writes per second to this disk device during the interval.

BYDSK_HISTOGRAM

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

A bar chart of the disk IO.

HP-UX

Shows breakout of the disk IO.

```
Disk IO Rate = BYDSK_FS_READ_RATE
             + BYDSK_FS_WRITE_RATE
             + BYDSK_RAW_READ_RATE
             + BYDSK_RAW_WRITE_RATE
             + BYDSK_VM_IO_RATE
             + BYDSK_SYSTEM_IO_RATE
```

ASCII and binary files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk IO on a character mode terminal display.

AIX NCR Sinix

Shows a breakout of the disk IO.

```
Disk IO Rate = BYDSK_PHYS_READ_RATE
             + BYDSK_PHYS_WRITE_RATE
```

ASCII and binary files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk IO on a character mode terminal display.

SunOS WinNT

Shows a breakout of the disk IO.

```
Disk IO Rate = BYDSK_PHYS_READ_RATE
             + BYDSK_PHYS_WRITE_RATE
```

ASCII and binary files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk IO on a character mode terminal display.

BYDSK_ID

PLATFORMS: HP-UX SunOS AIX

This is an identification number assigned to the disk device by scope.

BYDSK_LOGL_READ

PLATFORMS: HP-UX

The number of logical reads for this disk device during the interval.

Logical IO rates by disk device cannot be obtained in a multi-disk LVM configuration because there is no reasonable means of tying logical IO transactions to physical spindles spanned on the logical volume.

Therefore, if you have a multi-disk LVM configuration, you always see “na” for this metric.

BYDSK_LOGL_READ_RATE

PLATFORMS: HP-UX

The number of logical reads per second for this disk device during the interval.

Logical IO rates by disk device cannot be obtained in a multi-disk LVM configuration because there is no reasonable means of tying logical IO transactions to physical spindles spanned on the logical volume.

Therefore, if you have a multi-disk LVM configuration, you always see “na” for this metric.

BYDSK_LOGL_WRITE

PLATFORMS: HP-UX

The number of logical writes for this disk device during the interval.

Logical IO rates by disk device cannot be obtained in a multi-disk LVM configuration because there is no reasonable means of tying logical IO transactions to physical spindles spanned on the logical volume.

Therefore, if you have a multi-disk LVM configuration, you always see “na” for this metric.

BYDSK_LOGL_WRITE_RATE

PLATFORMS: HP-UX

The number of logical writes per second for this disk device during the interval.

Logical IO rates by disk device cannot be obtained in a multi-disk LVM configuration because there is no reasonable means of tying logical IO transactions to physical spindles spanned on the logical volume.

Therefore, if you have a multi-disk LVM configuration, you always see “na” for this metric.

BYDSK_PHYS_BYTE

PLATFORMS: SunOS AIX NCR Sinix WinNT

The average KB transferred to or from the current disk device during the interval.

SunOS

On Sun systems, this metric is only available on Sun 5.X or later.

BYDSK_PHYS_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The average KB per second transferred to or from this disk device during the interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

BYDSK_PHYS_IO

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of physical IOs for this disk device during the interval.

BYDSK_PHYS_IO_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The average number of physical IO requests per second for this disk device during the interval.

HP-UX SunOS AIX NCR Sinix

This counts disk reads and writes of all types, including virtual memory and raw IO.

Sinix

The count includes operations performed on virtual disks that completely or partially reside on this disk device.

BYDSK_PHYS_READ

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical reads for this disk device during the interval.

AIX

This is an estimated value based on the ratio of read bytes to total bytes transferred. The actual number of reads is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_READ =
  BYDSK_PHYS_IO *
  (BYDSK_PHYS_READ_BYTE /
   BYDSK_PHYS_IO_BYTE)
```

BYDSK_PHYS_READ_BYTE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The KB transferred from this disk device during the interval.

HP-UX SunOS AIX NCR Sinix

This counts all types of disk reads, including file system, virtual memory, and raw IO.

SunOS

On Sun systems, this metric is only available on 5.X or later.

Sinix

This is an estimated value based on the ratio of read requests to total physical IO requests. The actual number of bytes read is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_READ_BYTE =
  BYDSK_PHYS_BYTE *
  (BYDSK_PHYS_READ /
   BYDSK_PHYS_IO_RATE)
```

BYDSK_PHYS_READ_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average KB per second transferred from this disk device during the interval.

HP-UX SunOS AIX NCR Sinix

This counts all types of disk reads, including file system, virtual memory, and raw IO.

SunOS

On Sun systems, this metric is only available on 5.X or later.

Sinix

This is an estimated value based on the ratio of read requests to total physical IO requests. The actual number of bytes read is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_READ_BYTE_RATE =
  BYDSK_PHYS_BYTE_RATE *
  (BYDSK_PHYS_READ_RATE /
   BYDSK_PHYS_IO_RATE)
```

BYDSK_PHYS_READ_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average number of physical reads per second for this disk device during the interval.

AIX

This is an estimated value based on the ratio of read bytes to total bytes transferred. The actual number of reads is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_READ_RATE =
  BYDSK_PHYS_IO_RATE *
  (BYDSK_PHYS_READ_BYTE /
   BYDSK_PHYS_IO_BYTE)
```

BYDSK_PHYS_WRITE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical writes for this disk device during the interval.

AIX

This is an estimated value based on the ratio of write bytes to total bytes transferred because the actual number of writes is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_WRITE =
  BYDSK_PHYS_IO *
  (BYDSK_PHYS_WRITE_BYTE /
   BYDSK_PHYS_IO_BYTE)
```

BYDSK_PHYS_WRITE_BYTE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The KB transferred to this disk device during the interval.

HP-UX SunOS AIX NCR Sinix

This counts all types of disk writes, including file system, virtual memory, and raw IO.

SunOS

On Sun systems, this metric is only available on 5.X or later.

Sinix

This is an estimated value based on the ratio of read requests to total physical IO requests. The actual number of bytes read is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_WRITE_BYTE =
  BYDSK_PHYS_BYTE *
  (BYDSK_PHYS_WRITE /
   BYDSK_PHYS_IO_RATE)
```

BYDSK_PHYS_WRITE_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average KB per second transferred to this disk device during the interval.

HP-UX SunOS AIX NCR Sinix

This counts all types of writes, including file system, virtual memory, and raw IO.

SunOS

On Sun systems, this metric is only available on 5.X or later.

Sinix

This is an estimated value based on the ratio of write requests to total physical IO requests. The actual number of bytes written is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_WRITE_BYTE_RATE =
    BYDSK_PHYS_BYTE_RATE *
    (BYDSK_PHYS_WRITE_RATE /
     BYDSK_PHYS_IO_RATE)
```

BYDSK_PHYS_WRITE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average number of physical writes per second for this disk device during the interval.

AIX

This is an estimated value based on the ratio of write bytes to total bytes transferred. The actual number of writes is not tracked by the kernel.

This is computed as

```
BYDSK_PHYS_WRITE_RATE =
    BYDSK_PHYS_IO_RATE *
    (BYDSK_PHYS_WRITE_BYTE /
     BYDSK_PHYS_IO_BYTE)
```

BYDSK_RAW_READ

PLATFORMS: HP-UX

The number of physical raw reads made from this disk device during the interval.

BYDSK_RAW_READ_RATE

PLATFORMS: HP-UX

The number of raw reads per second made from this disk device during the interval.

BYDSK_RAW_WRITE

PLATFORMS: HP-UX

The number of physical raw writes made to this disk device during the interval.

BYDSK_RAW_WRITE_RATE

PLATFORMS: HP-UX

The number of raw writes per second made to this disk device during the interval.

BYDSK_REQUEST_QUEUE

PLATFORMS: HP-UX

The average number of IO requests that were in the wait queue for this disk device during the interval. These requests are the physical requests (as opposed to logical IO requests).

BYDSK_SYSTEM_IO

PLATFORMS: HP-UX

The number of physical system reads or writes to this disk device during the interval.

BYDSK_SYSTEM_IO_RATE

PLATFORMS: HP-UX

The number of physical system reads or writes per second to this disk device during the interval.

BYDSK_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The percentage of the time during the interval that the disk device had IO in progress from the point of view of the Operating System. In other words, the utilization or percentage of time busy servicing requests for this device.

SunOS AIX NCR Sinix WinNT

The percentage of the time that this disk device was busy transferring data during the interval.

This is a measure of the ability of the IO path to meet the transfer demands being placed on it. Slower disk devices may show a higher utilization with lower IO rates than faster disk devices such as disk arrays. A value of greater than 50% utilization over time may indicate that this device or its IO path is a bottleneck, and the access pattern of the workload, database, or files may need reorganizing for better balance of disk IO load.

BYDSK_VM_IO

PLATFORMS: HP-UX

The number of virtual memory IOs to this disk device during the interval.

BYDSK_VM_IO_RATE

PLATFORMS: HP-UX

The number of virtual memory IOs per second to this disk device during the interval.

BYNETIF_COLLISION

PLATFORMS: HP-UX SunOS AIX Sinix DEC

The number of physical collisions that occurred on the network interface during the interval. A rising rate of collisions versus outbound packets is an indication that the network is becoming increasingly congested. This metric does not currently include deferred packets.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

For HP-UX 10.20 and earlier releases, this is the same as the “Coll” column from the “netstat -i” command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the “Single Collision Frames”, “Multiple Collision Frames”, “Late Collisions”, and “Excessive Collisions” values from the output of the “lanadmin” utility for the network interface. Remember that “lanadmin” reports cumulative counts. For this release and beyond, “netstat -i” shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

This data is not available for loop-back (lo) or fiber-link (fddi) devices and is always zero.

This is the same as the “Collis” column from the “netstat -i” command for a network device. See also netstat(1).

AIX

AIX does not support the collision count for ethernet interface. The collision count is supported for token ring (tr) and loopback (lo) interface.

This is the same as the “Coll” column from the “netstat -i” command for a network device. See also netstat(1).

Sinix DEC

This data is not collected for non-broadcasting devices such as loopback (lo) or SLIP (sl) interface, and is always zero.

This is the same as the “colls” column from the “netstat” command for a network device. This refers to the “netstat” command form “netstat -I lo0 5” for example. See also netstat(1).

BYNETIF_COLLISION_1_MIN_RATE

PLATFORMS: HP-UX SunOS AIX Sinix DEC

The number of physical collisions per minute on the network interface during the interval. A rising rate of collisions versus outbound packets is an indication that the network is becoming increasingly congested. This metric does not currently include deferred packets.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

This data is not available for loop-back (lo) or fiber-link (fddi) devices and is always zero.

Sinix DEC

This data is not collected for non-broadcasting devices such as loopback (lo) or SLIP (sl) interface, and is always zero.

BYNETIF_COLLISION_RATE

PLATFORMS: HP-UX SunOS AIX Sinix DEC

The number of physical collisions per second on the network interface during the interval. A rising rate of collisions versus outbound packets is an indication that the network is becoming increasingly congested. This metric does not currently include deferred packets.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

This data is not available for loop-back (lo) or fiber-link (fddi) devices and is always zero.

Sinix DEC

This data is not collected for non-broadcasting devices such as loopback (lo) or SLIP (sl) interface, and is always zero.

BYNETIF_ERROR

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of physical errors that occurred on the network interface during the interval. An increasing number of errors may indicate a hardware problem in the network.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

For HP-UX 10.20 and earlier releases, this is the same as the sum of “Ierrs” and “Oerrs” from the “netstat -i” command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the “Inbound Errors” and “Outbound Errors” values from the output of the “lanadmin” utility for the network interface. Remember that “lanadmin” reports cumulative counts. For this release and beyond, “netstat -i” shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS AIX NCR Sinix DEC

This is the same as the sum of “Ierrs” and “Oerrs” from the “netstat -i” command for a network device. See also netstat(1).

SunOS Sinix DEC

This data is not available for loop-back (lo) devices and is always zero.

NCR

Currently on NCR systems, collisions are counted as outbound errors and the collision metrics are zero. This is true with all utilities that get their metrics from the kernel, such as netstat.

BYNETIF_ERROR_1_MIN_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of physical errors per minute on the network interface during the interval.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS Sinix

This data is not available for loop-back (lo) or fiber-link (fddi) devices and is always zero.

NCR

Currently on NCR systems, collisions are counted as outbound errors and the collision metrics are zero. This is true with all utilities that get their metrics from the kernel, such as netstat.

DEC

This data is not available for loop-back (lo) or SLIP (sl) interface and is always zero.

BYNETIF_ERROR_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of physical errors per second on the network interface during the interval.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS Sinix

This data is not available for loop-back (lo) or fiber-link (fddi) devices and is always zero.

NCR

Currently on NCR systems, collisions are counted as outbound errors and the collision metrics are zero. This is true with all utilities that get their metrics from the kernel, such as netstat.

DEC

This data is not available for loop-back (lo) or SLIP (sl) interface and is always zero.

BYNETIF_IN_PACKET

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of successful physical packets received through the network interface during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

For HP-UX 10.20 and earlier releases, this is the same as the “Ipkts” column from the “netstat -i” command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the “Inbound Unicast Packets” and “Inbound Non-Unicast Packets” values from the output of the “lanadmin” utility for the network interface. Remember that “lanadmin” reports cumulative counts. For this release and beyond, “netstat -i” shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS AIX NCR Sinix DEC

This is the same as the “Ipkts” column from the “netstat -i” command for a network device. See also netstat(1).

BYNETIF_IN_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of successful physical packets per second received through the network interface during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

BYNETIF_NAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The name of the network interface.

HP-UX

For HP-UX 10.20 and earlier releases, these are the same names that appear in the “Name” column of the “netstat -i” command.

For HP-UX 11.0 and beyond, these are the same names that appear in the “Description” field of the “lanadmin” command output.

SunOS

These are the same names that appear in the “Name” column of the “netstat -i” command. Some examples of device names are:

```
lo   loop-back driver
le   Lance Ethernet driver
ie   Intel Ethernet driver
bf   fiber optic driver
```

AIX NCR

These are the same names that appear in the “Name” column of the “netstat -i” command. Some examples of device names are:

```
lo   loop-back driver
en   Standard Ethernet driver
tr   Token-Ring driver
et   Ether Twist driver
```

All the lan name will have the unit number appended to the name. For example, a loop-back device in unit 0 will be “lo0”.

Sinix DEC

These are the same names that appear in the "Name" column of the "netstat -i" command. Some examples of device names are "lo". which identifies the loop-back driver and "pnet" which identifies the Standard Ethernet driver. All the names will have the unit number appended to the name. So, the name for a loop-back device in unit 0 will be "lo0".

BYNETIF_OUT_PACKET

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of successful physical packets sent through the network interface during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

For HP-UX 10.20 and earlier releases, this is the same as the "Opkts" column from the "netstat -i" command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the "Outbound Unicast Packets" and "Outbound Non-Unicast Packets" values from the output of the "lanadmin" utility for the network interface. Remember that "lanadmin" reports cumulative counts. For this release and beyond, "netstat -i" shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS AIX NCR Sinix DEC

This is the same as the "Opkts" column from the "netstat -i" command for a network device. See also netstat(1).

BYNETIF_OUT_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of successful physical packets per second sent through the network interface during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

BYNETIF_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of successful physical packets per second sent and received through the network interface during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

Physical statistics are packets recorded by the network drivers. These numbers most likely will not be the same as the logical statistics.

Logical statistics are packets seen only by the Interface Protocol (IP) layer of the networking subsystem. Not all packets seen by IP will go out and come in through a network driver. Examples cases are the 127.0.0.1 (loopback interface). Pings or other network generating commands (ftp, rlogin, and so forth) to 127.0.0.1 will not change physical driver statistics. Pings to IP addresses on remote systems will change physical driver statistics.

This is different from pre-11.0 systems where commands addressed to the local host always went down to the driver and the logical and physical counters were always updated.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

BYPROTOCOL_IN_PACKET

PLATFORMS: WinNT

The number of successful packets received via this protocol during the interval. Successful packets are those that have been processed without errors or collisions.

For NBT connections, the packet size is defined as 1 Kbytes.

BYPROTOCOL_IN_PACKET_RATE

PLATFORMS: WinNT

The number of successful packets per second received via this protocol during the interval. Successful packets are those that have been processed without errors or collisions.

For NBT connections, the packet size is defined as 1 Kbytes.

BYPROTOCOL_OUT_PACKET

PLATFORMS: WinNT

The number of successful packets sent via this protocol during the interval. Successful packets are those that have been processed without errors or collisions.

For NBT connections, the packet size is defined as 1 Kbytes.

BYPROTOCOL_OUT_PACKET_RATE

PLATFORMS: WinNT

The number of successful packets per second sent via this protocol during the interval. Successful packets are those that have been processed without errors or collisions.

For NBT connections, the packet size is defined as 1 Kbytes.

DATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The date the information in this record was captured, based on local time. The date is an ASCII field in mm/dd/yy format unless localized. If localized, the separators may be different and the subfield may be in a different sequence. In ASCII files this field will always contain 8 characters. Each subfield (mm, dd, yy) will contain a leading zero if the value is less than 10. This metric is extracted from GBL_STATTIME, which is obtained using the time() system call at the time of data collection.

This field responds to language localization. For example, in Germany the field would appear as dd.mm.yy and in Italy it would be dd/mm/yy.

In binary files this field is in MPE CALENDAR format in the least significant 16 bits of the field. The most significant 16 bits should all be zero. Dividing the field by 512 will isolate the year (that is, 94). This field MOD 512 will isolate the day of the year.

DATE_SECONDS

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The time that the data was captured, expressed in seconds since January 1, 1970. This is the standard measurement of time returned by the time() system call.

DAY

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The julian day of the year that the data in this record was captured. This metric is extracted from GBL_STATTIME.

FS_DIRNAME

PLATFORMS: HP-UX

The path name of the mount point of the file system if the logical volume has a mounted file system. This is the directory parameter of the mount(1M) command for most entries.

Exceptions:

```
* For lvm swap areas, this field
  contains “lvm swap device”.
* For logical volumes with no
  mounted file systems, this field
  contains “Raw Logical Volume”
  (relevant only to MeasureWare
  Agent).
```

The file names are in the same order as shown in the /usr/sbin/mount -p command. File systems are not displayed until they exhibit IO activity once the midaemon has been started. Also, once a device is displayed, it continues to be displayed (even after the device is unmounted) until the midaemon process terminates.

GBL_ACTIVE_CPU

PLATFORMS: SunOS Sinix

The number of CPUs online on the system.

SunOS

The commands psrinfo(1M) and psradm(1M) allow you to check or change the status of the system CPUs.

Sinix

The undocumented system utility “mpcntl” allows you to check or change the status of the system CPUs.

GBL_ACTIVE_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

An active process is one that exists and consumes some CPU time. GBL_ACTIVE_PROC is the sum of the alive-process-time/interval-time ratios of every process that is active (uses any CPU time) during an interval.

The following diagram of a four second interval during which two processes exist on the system should be used to understand the above definition. Note the difference between active processes, which consume CPU time, and alive processes which merely exist on the system.

```
         ----------- Seconds -----------
           1         2         3      4
   Proc
```

```
---- ----      ----      ----    ----
A    live      live      live    live
B    live/CPU  live/CPU  live    dead
```

Process A is alive for the entire four second interval but consumes no CPU. A's contribution to GBL_ALIVE_PROC is 4*1/4. A contributes 0*1/4 to GBL_ACTIVE_PROC. B's contribution to GBL_ALIVE_PROC is 3*1/4. B contributes 2*1/4 to GBL_ACTIVE_PROC. Thus, for this interval, GBL_ACTIVE_PROC equals 0.5 and GBL_ALIVE_PROC equals 1.75.

Because a process may be alive but not active, GBL_ACTIVE_PROC will always be less than or equal to GBL_ALIVE_PROC.

This metric is a good overall indicator of the workload of the system. An unusually large number of active processes could indicate a CPU bottleneck.

To determine if the CPU is a bottleneck, compare this metric with GBL_CPU_TOTAL_UTIL and GBL_RUN_QUEUE. If GBL_CPU_TOTAL_UTIL is near 100 percent and GBL_RUN_QUEUE is greater than one, there is a bottleneck.

SunOS AIX Sinix DEC WinNT

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

GBL_ALIVE_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS AIX NCR Sinix DEC WinNT

An alive process is one that exists on the system. GBL_ALIVE_PROC is the sum of the alive-process-time/interval-time ratios for every process.

The following diagram of a four second interval during which two processes exist on the system should be used to understand the above definition. Note the difference between active processes, which consume CPU time, and alive processes which merely exist on the system.

```
     ----------- Seconds -----------
        1         2         3      4
Proc
---- ----      ----      ----    ----
A    live      live      live    live
B    live/CPU  live/CPU  live    dead
```

Process A is alive for the entire four second interval but consumes no CPU. A's contribution to GBL_ALIVE_PROC is 4*1/4. A contributes

0*1/4 to GBL_ACTIVE_PROC. B's contribution to GBL_ALIVE_PROC is 3*1/4. B contributes 2*1/4 to GBL_ACTIVE_PROC. Thus, for this interval, GBL_ACTIVE_PROC equals 0.5 and GBL_ALIVE_PROC equals 1.75.

Because a process may be alive but not active, GBL_ACTIVE_PROC will always be less than or equal to GBL_ALIVE_PROC.

SunOS AIX Sinix DEC WinNT

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

GBL_ALIVE_THREAD

PLATFORMS: Win3X/95

The average number of threads sampled during a monitoring interval (Windows 95 only). Samples are collected every 30 seconds. A single virtual machine may have one or many threads of execution.

GBL_BLOCKED_IO_QUEUE

PLATFORMS: SunOS AIX

The average number of processes blocked on local disk resources (IO, paging). This metric is an indicator of disk contention among active processes. It should normally be a very small number. If GBL_DISK_UTIL_PEAK is near 100 percent and GBL_BLOCKED_IO_QUEUE is greater than 1, a disk bottleneck is probable.

SunOS

This is the same as the “procs b” field reported in vmstat.

GBL_COLLECTOR

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

ASCII field containing collector name and version. The collector name will appear as “SCOPE/xx V.UU.FF(#)” where xx is UX, NT or WD, where V = version, UU = update level, FF = fix level and # is the lab fix id. For example, SCOPE/UX A.09.18(32), SCOPE/NT A.00.00(42), SCOPE/WD A.00.00(12).

GBL_COMPLETED_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of processes that terminated during the interval.

SunOS AIX NCR Sinix DEC WinNT

This metric is derived from sampled process data. Since the data for a process is not available after the process has died on this operating system, a process whose life is shorter than the sampling interval may not be seen when the samples are taken. Thus this metric may be slightly less than the actual value. Increasing the sampling frequency captures a more accurate count, but the overhead of collection may also rise.

GBL_CPU_CSWITCH_TIME

PLATFORMS: HP-UX

The time, in seconds, that the CPU spent context switching during the interval.

This includes context switches that result in the execution of a different process and those caused by a process stopping, then resuming, with no other process running in the meantime.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_CSWITCH_UTIL

PLATFORMS: HP-UX

The percentage of time that the CPU spent context switching during the interval.

This includes context switches that result in the execution of a different process and those caused by a process stopping, then resuming, with no other process running in the meantime.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_HISTOGRAM

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

Histogram of CPU utilization components.

HP-UX

Shows breakout:

```
GBL_CPU_TOTAL_UTIL = GBL_CPU_SYSCALL_UTIL
                   + GBL_CPU_NICE_UTIL
                   + GBL_CPU_NORMAL_UTIL
                   + GBL_CPU_REALTIME_UTIL
                   + GBL_CPU_CSWITCH_UTIL
                   + GBL_CPU_INTERRUPT_UTIL
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of CPU usage on a character-mode terminal display.

SunOS AIX NCR Sinix

Shows breakout:

```
GBL_CPU_TOTAL_UTIL = GBL_CPU_SYS_MODE_UTIL
```

```
                       + GBL_CPU_USER_MODE_UTIL
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of CPU usage on a character-mode terminal display.

WinNT

Shows breakout:

```
GBL_CPU_TOTAL_UTIL = GBL_CPU_SYS_MODE_UTIL
                   + GBL_CPU_USER_MODE_UTIL
                   + GBL_CPU_INTERRUPT_UTIL
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of CPU usage on a character-mode terminal display.

GBL_CPU_IDLE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The time, in seconds, that the CPU was idle during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online.

GBL_CPU_IDLE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The percentage of time that the CPU was idle during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online.

GBL_CPU_INTERRUPT_TIME

PLATFORMS: HP-UX WinNT

The time, in seconds, that the CPU spent processing interrupts during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_INTERRUPT_UTIL

PLATFORMS: HP-UX WinNT

The percentage of time that the CPU spent processing interrupts during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_NICE_TIME

PLATFORMS: HP-UX DEC SunOS

The time, in seconds, that the CPU was in user mode at a nice priority during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_NICE_UTIL

PLATFORMS: HP-UX DEC SunOS

The percentage of time that the CPU was in user mode at a nice priority during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_NORMAL_TIME

PLATFORMS: HP-UX

The time, in seconds, that the CPU was in user mode at normal priority during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_NORMAL_UTIL

PLATFORMS: HP-UX

The percentage of time that the CPU was in user mode at normal priority during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_REALTIME_TIME

PLATFORMS: HP-UX

The time, in seconds, that the CPU was in user mode at a realtime priority during the interval.

Running at a realtime priority means that the process or kernel thread was run using the rtprio command or the rtprio system call to alter its priority. Realtime priorities range from zero to 127 and are absolute priorities, meaning the realtime process with the lowest priority runs as long as it wants to. Since this can have a huge impact on the system, the realtime CPU is tracked separately to make visible the effect of using realtime priorities.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_REALTIME_UTIL

PLATFORMS: HP-UX

The percentage of time that the CPU was in user mode at a realtime priority during the interval.

Running at a realtime priority means that the process or kernel thread was run using the rtprio command or the rtprio system call to alter its priority. Realtime priorities range from zero to 127 and are absolute priorities, meaning the realtime process with the lowest priority runs as long as it wants to. Since this can have a huge impact on the system, the realtime CPU is tracked separately to make visible the effect of using realtime priorities.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_SYSCALL_TIME

PLATFORMS: HP-UX

The time, in seconds, that the CPU was in system mode (excluding interrupt, context switch, trap, or vfault CPU) during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_SYSCALL_UTIL

PLATFORMS: HP-UX

The percentage of time that the CPU was in system mode (excluding interrupt, context switch, trap, or vfault CPU) during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_SYS_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The time, in seconds, that the CPU was in system mode during the interval.

HP-UX SunOS AIX NCR

A UNIX process operates in either system mode (also called kernel mode) or user mode. When a process requests services from the operating system with a system call, it switches into the machine's privileged protection mode and runs in system mode.

WinNT

A process operates in either system mode (also called privileged mode) or user mode. When a process requests services from the operating system, it switches into the machine's privileged protection mode and runs in system mode.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_SYS_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

Percentage of time the CPU was in system mode during the interval.

HP-UX SunOS AIX NCR

A UNIX process operates in either system mode (also called kernel mode) or user mode. When a process requests services from the operating system with a system call, it switches into the machine's privileged protection mode and runs in system mode.

WinNT

A process operates in either system mode (also called privileged mode) or user mode. When a process requests services from the operating system, it switches into the machine's privileged protection mode and runs in system mode.

This metric is a subset of the GBL_CPU_TOTAL_UTIL percentage.

This is NOT a measure of the amount of time used by system daemon processes, since most system daemons spend part of their time in user mode and part in system calls, like any other process.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

High system mode CPU percentages are normal for IO intensive applications. Abnormally high system mode CPU percentages can indicate that a hardware problem is causing a high interrupt rate. It can also indicate programs that are not calling system calls efficiently.

GBL_CPU_TOTAL_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

The total time, in seconds, that the CPU was not idle in the interval.

HP-UX SunOS AIX DEC

This is calculated as

```
GBL_CPU_TOTAL_TIME =
  GBL_CPU_USER_MODE_TIME +
  GBL_CPU_SYS_MODE_TIME
```

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_TOTAL_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

Percentage of time the CPU was not idle during the interval.

HP-UX SunOS AIX DEC

This is calculated as

```
GBL_CPU_TOTAL_UTIL =
  GBL_CPU_USER_MODE_UTIL +
  GBL_CPU_SYS_MODE_UTIL
```

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

```
GBL_CPU_TOTAL_UTIL +
 GBL_CPU_IDLE_UTIL = 100%
```

This metric varies widely on most systems, depending on the workload. A consistently high CPU utilization can indicate a CPU bottleneck, especially when other indicators such as GBL_RUN_QUEUE and GBL_ACTIVE_PROC are also high. High CPU utilization can also occur on systems that are bottlenecked on memory, because the CPU spends more time paging and swapping.

WinNT

NOTE: This metric may not equal the sum of the APP_CPU_TOTAL_UTIL metrics. Microsoft states that “this is expected behavior” because this GBL_CPU_TOTAL_UTIL metric is taken from the NT performance library Processor objects while the APP_CPU_TOTAL_UTIL metrics are taken from the Process objects. Microsoft states that there can be CPU time accounted for in the Processor system objects that may not be seen in the Process objects.

GBL_CPU_USER_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The time, in seconds, that the CPU was in user mode during the interval.

HP-UX

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

GBL_CPU_USER_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of time the CPU was in user mode during the interval.

HP-UX

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

This metric is a subset of the GBL_CPU_TOTAL_UTIL percentage.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

High user mode CPU percentages are normal for computation-intensive applications. Low values of user CPU utilization compared to relatively high values for GBL_CPU_SYS_MODE_UTIL can indicate an application or hardware problem.

GBL_CPU_WAIT_TIME

PLATFORMS: SunOS AIX NCR Sinix

The time, in seconds, that the CPU was idle and there were processes waiting for physical IOs to complete during the interval.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_CPU_WAIT_UTIL

PLATFORMS: SunOS AIX NCR Sinix

The percentage of time during the interval that the CPU was idle and there were processes waiting for physical IOs to complete.

On a system with multiple CPUs, this metric is normalized. That is, the CPU used over all processors is divided by the number of processors online. This represents the usage of the total processing capacity available.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_BLOCK_IO

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The total number of block IOs during the interval (Sun 5.X or later).

These are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The total number of block IOs during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These do include the IO of the inode (system write) and the file system data IO.

NCR Sinix

The total number of block IOs during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_BLOCK_IO_RATE

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The total number of block IOs per second during the interval (Sun 5.X or later).

These are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The total number of block IOs per second during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These do include the IO of the inode (system write) and the file system data IO.

NCR Sinix

The total number of block IOs per second during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_BLOCK_READ

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The number of block reads during the interval (Sun 5.X or later).

These are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The number of block reads during the interval.

These are physical reads generated by file system access and do not include virtual memory reads, or reads relating to raw disk access. These do include the read of the inode (system read) and the file data read.

NCR Sinix

The number of block reads during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_BLOCK_READ_RATE

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The number of block reads per second during the interval (Sun 5.X or later).

These are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The number of block reads per second during the interval.

These are physical reads generated by file system access and do not include virtual memory reads, or reads relating to raw disk access. These do include the read of the inode (system read) and the file data read.

NCR Sinix

The number of block reads per second during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_BLOCK_WRITE

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The number of block writes during the interval (Sun 5.X or later).

These are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The number of block writes during the interval.

These are physical writes generated by file system access and do not include virtual memory writes, or writes relating to raw disk access. These do include the write of the inode (system write) and the file system data write.

NCR Sinix

The number of block writes during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_BLOCK_WRITE_RATE

PLATFORMS: SunOS AIX NCR Sinix

SunOS

The number of block writes per second during the interval (Sun 5.X or later).

These are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

The number of block writes per second during the interval.

These are physical writes generated by file system access and do not include virtual memory writes, or writes relating to raw disk access. These do include the write of the inode (system write) and the file system data write.

NCR Sinix

The number of block writes per second during the interval.

These are physical IOs generated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache.

GBL_DISK_CACHE_READ

PLATFORMS: WinNT

The number of cached reads made during the interval.

GBL_DISK_CACHE_READ_RATE

PLATFORMS: WinNT

The number of cached reads per second made during the interval.

GBL_DISK_FS_IO

PLATFORMS: HP-UX

The total of physical file system disk reads and writes during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical IOs generated by user file system access and do not include virtual memory IOs, system IOs (inode updates), or IOs relating to raw disk access. An exception is user files accessed via the mmap(2) call, which will not show their physical IOs in this category. They appear under virtual memory IOs.

GBL_DISK_FS_IO_RATE

PLATFORMS: HP-UX Win3X/95

The total of file system disk physical reads and writes per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical IOs generated by user file system access and do not include virtual memory IOs, system IOs (inode updates), or IOs relating to raw disk access. An exception is user files accessed via the mmap(2) call, which will not show their physical IOs in this category. They appear under virtual memory IOs.

GBL_DISK_FS_READ

PLATFORMS: HP-UX Win3X/95

The number of file system disk reads during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical reads in this category. They appear under virtual memory reads.

GBL_DISK_FS_READ_RATE

PLATFORMS: HP-UX Win3X/95

The number of file system disk reads per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical reads in this category. They appear under virtual memory reads.

GBL_DISK_FS_WRITE

PLATFORMS: HP-UX Win3X/95

The number of file system disk writes during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical writes generated by user file system access and do not include virtual memory writes, system writes (inode updates), or writes relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical writes in this category. They appear under virtual memory writes.

GBL_DISK_FS_WRITE_RATE

PLATFORMS: HP-UX Win3X/95

The number of file system disk writes per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical writes generated by user file system access and do not include virtual memory writes, system writes (inode updates), or writes relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical writes in this category. They appear under virtual memory writes.

GBL_DISK_HISTOGRAM

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

Histogram of physical Disk IO rate components.

HP-UX

Shows breakout:

```
GBL_DISK_PHYS_IO_RATE = GBL_DISK_VM_IO_RATE
                      + GBL_DISK_SYSTEM_IO_RATE
                      + GBL_DISK_FS_IO_RATE
                      + GBL_DISK_RAW_IO_RATE
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk usage on a character-mode terminal display.

AIX NCR Sinix

Shows breakout:

```
GBL_DISK_PHYS_IO_RATE = GBL_DISK_BLOCK_IO_RATE
                      + GBL_DISK_VM_IO_RATE
                      + GBL_DISK_RAW_IO_RATE
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk usage on a character-mode terminal display.

SunOS

On Solaris, shows breakout:

```
GBL_DISK_PHYS_IO_RATE = GBL_DISK_BLOCK_READ_RATE
                      + GBL_DISK_BLOCK_WRITE_RATE
                      + GBL_DISK_RAW_READ_RATE
                      + GBL_DISK_RAW_WRITE_RATE
                      + GBL_DISK_VM_IO_RATE
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk usage on a character-mode terminal display.

On SunOS 4.1.3, shows breakout:

```
GBL_DISK_PHYS_IO_RATE = GBL_DISK_PHYS_READ_RATE
                      + GBL_DISK_PHYS_WRITE_RATE
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk usage on a character-mode terminal display.

WinNT

Shows breakout:

```
GBL_DISK_PHYS_IO_RATE = GBL_DISK_PHYS_READ_RATE
                      + GBL_DISK_PHYS_WRITE_RATE
```

ASCII and BINARY files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of Disk usage on a character-mode terminal display.

GBL_DISK_LOGL_IO

PLATFORMS: HP-UX SunOS NCR Sinix

The number of logical IOs made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file.

Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_LOGL_IO_RATE

PLATFORMS: HP-UX SunOS NCR Sinix Win3X/95

The number of logical IOs per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_LOGL_READ

PLATFORMS: HP-UX SunOS NCR Sinix WinNT

HP-UX NCR Sinix WinNT

The number of logical reads made during the interval.

SunOS

The number of logical block reads made during the interval.

WinNT

This includes both buffered (cached) read requests and unbuffered reads.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_LOGL_READ_BYTE

PLATFORMS: HP-UX

The number of KBs transferred through logical reads during the last interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

GBL_DISK_LOGL_READ_BYTE_RATE

PLATFORMS: HP-UX

The number of KBs transferred per second via logical reads during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

GBL_DISK_LOGL_READ_RATE

PLATFORMS: HP-UX SunOS NCR Sinix WinNT

HP-UX NCR Sinix WinNT

The average number of logical reads per second made during the interval.

SunOS

The average number of logical block reads per second made during the interval.

WinNT

This includes both buffered (cached) read requests and unbuffered reads.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_LOGL_WRITE

PLATFORMS: HP-UX SunOS NCR Sinix

HP-UX NCR Sinix

The number of logical writes made during the interval.

SunOS

The number of logical block writes during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_LOGL_WRITE_BYTE

PLATFORMS: HP-UX

The number of KBs transferred via logical writes during the last interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

GBL_DISK_LOGL_WRITE_BYTE_RATE

PLATFORMS: HP-UX

The number of KBs per second transferred via logical writes during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

GBL_DISK_LOGL_WRITE_RATE

PLATFORMS: HP-UX SunOS NCR Sinix

HP-UX NCR Sinix

The average number of logical writes per second made during the interval.

SunOS

The average number of logical block writes per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX NCR Sinix

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

GBL_DISK_PHYS_BYTE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of KBs of data transferred to and from all local disks on the system during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

It is not directly related to the number of IOs, since IO requests can be of differing lengths.

HP-UX SunOS AIX NCR Sinix DEC

This metric counts all types of physical IOs, including file system, virtual memory, and raw IO.

WinNT

This metric counts all types of physical IOs.

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The average number of KBs per second at which data was transferred to and from disks during the interval. The bytes for all types physical IOs are counted.

Only local disks are counted in this measurement. NFS devices are excluded.

This is a measure of the physical data transfer rate. It is not directly related to the number of IOs, since IO requests can be of differing lengths.

This is an indicator of how much data is being transferred to and from disk devices. Large spikes in this metric can indicate a disk bottleneck.

HP-UX SunOS AIX NCR Sinix DEC

This includes file system, virtual memory, and raw IO.

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_IO

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of physical IOs during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX SunOS AIX NCR Sinix DEC

This includes all types of physical reads and writes to disk, including virtual memory IO and raw IO.

Sinix

It also includes the operations performed on disk based virtual disks.

HP-UX

This is calculated as

```
GBL_DISK_PHYS_IO =
  GBL_DISK_FS_IO +
  GBL_DISK_VM_IO +
  GBL_DISK_SYSTEM_IO +
  GBL_DISK_RAW_IO
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_IO_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of physical IOs per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX SunOS AIX NCR Sinix DEC

This includes all types of physical IOs to disk, including virtual memory IO and raw IO.

Sinix

It also includes the operations performed on disk based virtual disks.

HP-UX

This is calculated as

```
GBL_DISK_PHYS_IO_RATE =
  GBL_DISK_FS_IO_RATE +
  GBL_DISK_VM_IO_RATE +
```

```
    GBL_DISK_SYSTEM_IO_RATE +
    GBL_DISK_RAW_IO_RATE
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_READ

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical reads during the interval.

HP-UX SunOS AIX NCR Sinix

This includes all types of physical reads from disk, including VM and raw reads.

Sinix

It also includes the reads performed on disk based virtual disks.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

There are many reasons why there is not a direct correlation between the number of logical IOs and physical IOs. For example, small sequential logical reads may be satisfied from the buffer cache, resulting in fewer physical IOs than logical IOs. Conversely, large logical IOs or small random IOs may result in more physical than logical IOs. Logical volume mappings, logical disk mirroring, and disk striping also tend to remove any correlation.

This is calculated as

```
GBL_DISK_PHYS_READ =
  GBL_DISK_FS_READ +
  GBL_DISK_VM_READ +
  GBL_DISK_SYSTEM_READ +
  GBL_DISK_RAW_READ
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_READ_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average number of KBs transferred from the disk per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the "by-disk" data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

Sinix

This is an estimated value based on the ratio of read requests to total physical IO requests. The actual number of bytes read is not tracked by the kernel.

GBL_DISK_PHYS_READ_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical reads per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX SunOS AIX NCR Sinix

This includes all types of physical reads from disk, including VM and raw reads.

Sinix

It also includes the reads performed on disk based virtual disks.

HP-UX

This is calculated as

```
GBL_DISK_PHYS_READ_RATE =
  GBL_DISK_FS_READ_RATE +
  GBL_DISK_VM_READ_RATE +
  GBL_DISK_SYSTEM_READ_RATE +
  GBL_DISK_RAW_READ_RATE
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the "by-disk" data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_WRITE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical writes during the interval.

HP-UX SunOS AIX NCR Sinix

This includes all types of physical writes to disk, including virtual memory and raw IO.

Sinix

It also includes writes performed on disk based virtual disks.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

There are many reasons why there is not a direct correlation between logical IOs and physical IOs. For example, small logical writes may end up entirely in the buffer cache, and later generate fewer physical IOs when written to disk due to the larger IO size. Or conversely, small logical writes may require physical prefetching of the corresponding disk blocks before the data is merged and posted to disk. Logical volume mappings, logical disk mirroring, and disk striping also tend to remove any correlation.

This is calculated as

```
GBL_DISK_PHYS_WRITE =
  GBL_DISK_FS_WRITE +
  GBL_DISK_VM_WRITE +
  GBL_DISK_SYSTEM_WRITE +
  GBL_DISK_RAW_WRITE
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_PHYS_WRITE_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average number of disk KB transferred to the disk per second during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

Sinix

This is an estimated value based on the ratio of write requests to total physical IO requests. The actual number of bytes written is not tracked by the kernel.

GBL_DISK_PHYS_WRITE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of physical writes per second during the interval.

HP-UX SunOS AIX NCR Sinix

This includes all types of physical writes to disk, including virtual memory IO and raw IO.

Sinix

It also includes writes performed on disk based virtual disks.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

This is calculated as

```
GBL_DISK_PHYS_WRITE_RATE =
  GBL_DISK_FS_WRITE_RATE +
  GBL_DISK_VM_WRITE_RATE +
  GBL_DISK_SYSTEM_WRITE_RATE +
  GBL_DISK_RAW_WRITE_RATE
```

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_RAW_IO

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The total number of raw reads and writes during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

SunOS

On Sun systems, this metric is only available on 5.X or later.

On Sun systems, tape drive accesses are included in raw IOs, but not in physical IOs. To determine if raw IO is tape access versus disk access, compare the global physical disk accesses to the total raw, block, and vm IOs. If the totals are the same, the raw IO activity is to a disk, floppy, or CD drive. Check physical IO data for each individual disk device to isolate a device. If the totals are different, there is raw IO activity to a non-disk device like a tape drive.

GBL_DISK_RAW_IO_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The total number of raw reads and writes per second during the interval.

Only accesses to local disk devices are counted.

SunOS

On Sun systems, this metric is only available on 5.X or later.

On Sun systems, tape drive accesses are included in raw IOs, but not in physical IOs. To determine if raw IO is tape access versus disk access, compare the global physical disk accesses to the total raw, block, and vm IOs. If the totals are the same, the raw IO activity is to a disk, floppy, or CD drive. Check physical IO data for each individual disk device to isolate a device. If the totals are different, there is raw IO activity to a non-disk device like a tape drive.

GBL_DISK_RAW_READ

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of raw reads during the interval. Only accesses to local disk devices are counted.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_RAW_READ_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of raw reads per second during the interval. Only accesses to local disk devices are counted.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_RAW_WRITE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of raw writes during the interval. Only accesses to local disk devices are counted.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_RAW_WRITE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of raw writes per second during the interval. Only accesses to local disk devices are counted.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_REQUEST_QUEUE

PLATFORMS: SunOS NCR Sinix DEC WinNT

The total length of all of the disk queues at the end of the interval.

SunOS

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the "by-disk" data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

GBL_DISK_SPACE

PLATFORMS: Win3X/95

The total disk capacity of the installed local fixed disk in megabytes.

GBL_DISK_SPACE_FREE

PLATFORMS: Win3X/95

The total free space on all local, fixed disk partitions. The value is sampled every 30 seconds and the minimum sample in the recording Interval is reported.

GBL_DISK_SUBSYSTEM_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on the disk subsystem (in a "queue" waiting for their file system disk IO to complete) during the interval. This is the sum of processes or kernel threads in the DISK, INODE, CACHE and CDFS wait states. Processes or kernel threads doing raw IO to a disk are not included in this measurement. As this number rises, it is an indication of a disk bottleneck.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on (DISK + INODE + CACHE + CDFS) divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_DISK_SYSTEM_IO

PLATFORMS: HP-UX

The number of physical disk IOs generated by the kernel for file system management (inode accesses or updates) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_SYSTEM_IO_RATE

PLATFORMS: HP-UX

The number of physical disk IOs per second generated by the kernel for file system management (inode accesses or updates) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_SYSTEM_READ

PLATFORMS: HP-UX

Number of physical disk reads generated by the kernel for file system management (inode accesses) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_SYSTEM_READ_RATE

PLATFORMS: HP-UX

Number of physical disk reads per second generated by the kernel for file system management (inode accesses) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_SYSTEM_WRITE

PLATFORMS: HP-UX

Number of physical disk writes generated by the kernel for file system management (inode updates) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_SYSTEM_WRITE_RATE

PLATFORMS: HP-UX

Number of physical disk writes per second generated by the kernel for file system management (inode updates) during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_TIME_PEAK

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The time, in seconds, during the interval that the busiest disk was performing IO transfers. This is for the busiest disk only, not all disk devices. This counter is based on an end-to-end measurement for each IO transfer updated at queue entry and exit points.

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The average percentage of time during the interval that all disks had IO in progress from the point of view of the Operating System. This is the average utilization for all disks.

SunOS AIX NCR Sinix DEC

The average percentage of disk in use time of the total interval (that is, the average utilization).

Only local disks are counted in this measurement. NFS devices are excluded.

GBL_DISK_UTIL_PEAK

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The utilization of the busiest disk during the interval.

HP-UX

This utilization is the percentage of time during the interval that the busiest disk device had IO in progress from the point of view of the Operating System.

SunOS AIX NCR Sinix DEC WinNT

This utilization is the percentage of time during the interval that the busiest disk was performing IO transfers. It is not an average utilization over all the disk devices.

Only local disks are counted in this measurement. NFS devices are excluded.

A peak disk utilization of more than 50 percent often indicates a disk IO subsystem bottleneck situation. A bottleneck may not be in the physical disk drive itself, but elsewhere in the IO path.

GBL_DISK_VM_IO

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The total number of virtual memory IOs made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

IOs to user file data are not included in this metric unless they were done via the mmap(2) system call.

SunOS

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

This metric is calculated by subtracting raw and block IOs from physical IOs. Tape drive accesses are included in the raw IOs, but not in the physical IOs. Therefore, when tape drive accesses are occurring on a system, all virtual memory and raw IO is counted as raw IO. For example, you may see heavy raw IO occurring during system backup. Raw IOs for disks are counted in the physical IOs.

To determine if the raw IO is tape access versus disk access, compare the global physical disk accesses to the total of raw, block, and VM IOs. If the totals are the same, the raw IO activity is to a disk, floppy, or CD drive.

Check physical IO data for each individual disk device to isolate a device. If the totals are different, there is raw IO activity to a non-disk device like a tape drive.

On Sun systems, this metrics is only available on 5.X or later.

GBL_DISK_VM_IO_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of virtual memory IOs per second made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

IOs to user file data are not included in this metric unless they were done via the mmap(2) system call.

SunOS

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

This metric is calculated by subtracting raw and block IOs from physical IOs. Tape drive accesses are included in the raw IOs, but not in the physical IOs. Therefore, when tape drive accesses are occurring on a system, all virtual memory and raw IO is counted as raw IO. For example, you may see heavy raw IO occurring during system backup. Raw IOs for disks are counted in the physical IOs.

To determine if the raw IO is tape access versus disk access, compare the global physical disk accesses to the total of raw, block, and VM IOs. If the totals are the same, the raw IO activity is to a disk, floppy, or CD drive.

Check physical IO data for each individual disk device to isolate a device. If the totals are different, there is raw IO activity to a non-disk device like a tape drive.

On Sun systems, this metric is only available on 5.X or later.

GBL_DISK_VM_READ

PLATFORMS: HP-UX AIX NCR Sinix

The number of virtual memory reads made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Reads to user file data are not included in this metric unless they were accessed via the mmap(2) system call.

GBL_DISK_VM_READ_RATE

PLATFORMS: HP-UX AIX NCR Sinix

The number of virtual memory reads per second made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Reads to user file data are not included in this metric unless they were accessed via the mmap(2) system call.

GBL_DISK_VM_WRITE

PLATFORMS: HP-UX AIX NCR Sinix

The number of virtual memory writes made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Writes to user file data are not included in this metric unless they were done via the mmap(2) system call.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

GBL_DISK_VM_WRITE_RATE

PLATFORMS: HP-UX AIX NCR Sinix

The number of virtual memory writes per second made during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

Writes to user file data are not included in this metric unless they were done via the mmap(2) system call.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

GBL_FS_SPACE_UTIL_PEAK

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of occupied disk space to total disk space for the fullest file system found during the interval. Only locally mounted file systems are counted in this metric.

HP-UX AIX NCR Sinix DEC

CDROM and PC file systems are also excluded.

This metric can be used as an indicator that at least one file system on the system is running out of disk space.

This metric can exceed 100 percent. This is because a portion of the file system space is reserved as a buffer and can only be used by root. If the root user has made the file system grow beyond the reserved buffer, the utilization will be greater than 100 percent. This is a dangerous situation since if the root user totally fills the file system, the system may crash.

SunOS

CDROM and PC file systems are also excluded.

This metric can be used as an indicator that at least one file system on the system is running out of disk space.

This is because a portion of the file system space is reserved (minfree) and can only be used by root. Root user can make the file system grow beyond the reserved portion of file system,and this can be a dangerous situation as if the root user totally fills the file system, then the system may crash.

WinNT

CDROM file systems are also excluded.

GBL_GUI_DELAY_INDEX

PLATFORMS: Win3X/95

A value ranging from 0 (fast) to 100 (slow) indicating the delay in processing Windows messages (Windows 3.11 only). Slow message processing (delays greater than around 20) indicates sluggish responses to mouse and keyboard actions. This measurement is used only on 16-bit Windows platforms. The metric GBL_GUI_INPUT_DELAY provides similar system responsiveness information on 32-bit platforms.

GBL_GUI_INPUT_COUNT

PLATFORMS: Win3X/95

The number of keyboard strokes and mouse clicks during the interval.

GBL_GUI_INPUT_DELAY

PLATFORMS: Win3X/95

The average delay in milliseconds in servicing keyboard strokes and mouse clicks. This metric is a general service measurement suitable for 32-bit platforms. It provides a “one number” indicator of the overall system performance seen by the user. The value should generally be close to zero. Although this metric may be reported for 16-bit platforms, it can be volatile and less reliable due to the lower system clock resolution on DOS-based Windows systems. The metric GBL_GUI_DELAY_INDEX performs a similar function on 16-bit platforms. This number is an average that is often influenced by single, lengthy delays. While normal delays may be less than a few ms, occasional delays (for example, during a large program load) may be several thousands of ms. Isolated long delay averages probably do not indicate poor service. Sustained average delays above a few ms probably do indicate poor service.

GBL_GUI_INPUT_RATE

PLATFORMS: Win3X/95

The number of keyboard strokes and mouse clicks per minute.

GBL_GUI_KEYBOARD_COUNT

PLATFORMS: Win3X/95

The number of keyboard strokes during the interval.

GBL_GUI_KEYBOARD_RATE

PLATFORMS: Win3X/95

The number of keyboard strokes per minute.

GBL_GUI_MOUSE_COUNT

PLATFORMS: Win3X/95

The number of mouse clicks and double clicks during the interval.

GBL_GUI_MOUSE_RATE

PLATFORMS: Win3X/95

The number of mouse clicks and double clicks per minute.

GBL_IPC_SUBSYSTEM_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on the InterProcess Communication (IPC) subsystems (waiting for their interprocess communication activity to complete) during the interval. This is the sum of processes or kernel threads in the IPC, MSG, SEM, PIPE, SOCKT (that is, sockets) and STRMS (that is, streams IO) wait states.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on (IPC + MSG + SEM + PIPE + SOCKT + STRMS) divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_LOADAVG

PLATFORMS: SunOS AIX NCR Sinix DEC

The average load average of the system during the interval.

This metric is derived from a kernel variable (avenrun) which is calculated by summing the number of runnable processes and averaging the samples over the last minute. Processes marked “runnable” include:

```
* a process using the CPU at the
  time of the sample
* a process waiting for the CPU at
  the time of the sample
* a process paused on a “short disk
```

```
wait" at the time of the sample
```

Because this metric can include processes which are waiting for disk IO to complete, it is not a reliable CPU bottleneck indicator. Several standard UNIX commands, such as uptime(1), display avenrun as the "1-minute Load Average."

GBL_LOGFILE_VERSION

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

Three byte ASCII field containing the log file version number. The log file version is assigned by scopeux and is incremented when changes to the log file causes the layout to be different from previous versions. The current version is " D". Every effort is made to protect the information investment maintained in historical log files by providing forward compatibility and/or conversion utilities when log files change.

GBL_LOGGING_TYPES

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

An 9-byte field indicating the types of data logged by the collector. This is controlled by the LOG statement in the parm file. Each position will contain either a space or the characters noted below. Note that positions three (all applications) and five (all processes) were implemented for HP internal use only and are not normally used outside of HP.

An @ in position three indicates that all applications are logged each five minute interval even if they had no activity during the interval. An @ in position five indicates that all processes, not just the interesting ones, are logged each one minute interval. This can result in very large log files.

Position	Char	Meaning
1	space	Not Used
2	G	Global data
3	@	All applications
4	A	Applications
5	@	All processes
6	P	Interesting processes
7	F	File system device
8	D	Disk or volume device
9	T	Transaction data

HP-UX

Note: Position 7, File system device, is not currently implemented.

By default, global and interesting process data is logged in which case this field would be " G   P ".

SunOS AIX

By default, global, interesting process, and disk device data is logged, in which case this field would be " G   P D".

GBL_LOST_MI_TRACE_BUFFERS

PLATFORMS: HP-UX SunOS Sinix

HP-UX

The number of trace buffers lost by the measurement processing daemon. If this value is > 0, the measurement subsystem is not keeping up with the system events that generate traces.

SunOS Sinix

The number of trace buffers lost by the measurement processing daemon. If this value is > 0, the measurement subsystem is not keeping up with the ARM API calls that generate traces.

GBL_MACHINE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX NCR Sinix DEC

A text string representing the type of computer. This is similar to what is returned by the command “uname -m”.

SunOS

A text string representing the type of computer as returned by the command “uname -m”. For example, “sun4m”.

AIX

A text string representing the model number of the computer.

This is calculated by decoding the unique ID number portion of the machine ID number returned by the “uname -m” command. A sample value for this metric might be “7012/320”.

Because one code may correspond to more than one model number, the larger model number is generally chosen in the code translation unless field length restriction would make that selection impractical.

The table below lists the “uname -m” codes and associated RISC model numbers which were current as of the last product release. As more RISC models are released, this table and the associated translation table will become incomplete. For this situation, this metric will contain the unique two digit machine ID code since no translation will be available.

```
"uname -m"
   code    RISC Model Numbers
---------  ------------------------
02 or 20   7015/930
10         7016/730, 7013/530
11 or 14   7013/540
18         7013/53H
1C         7013/550
2E         7015/950
30         7018/740/741, 7013/520
31         7012/320
34         7013/52H
```

```
35         7012/32H
37         7012/340
38 or 77   7012/350
41         7011/220
42         7006/41W/41T
43         7008/M20/M2A
46         7011/250
47         7011/230
48         7009/C10
49         7011/25F (80MHz 601)
4C         7248/43P/All, 7025/F40
           7025/F50, 7026/H10,
           7043/140, 7043/240
4D         7020/40P
57         7030/3BT, 7012/390,
           9076/SP2/thin 66MHz
58         7030/3AT, 7012/380
59         7030/3CT, 7012/39H,
           9076/SP2/thin w/L2
5C         7013/560
63         7015/970/97B
64         7015/980/98B
66         7013/580
67         7018/770/771, 7013/570
           7015/R10
70         7013/590, 9076/SP2/wide
71         7013/58H
72         7015/R20, 7013/59H,
           9076/SP2/wide w/RPQ
75         7012/370/37T/375,
           9076/SP1/thin
76         7012/360/36T/365
77         7013/550L, 7012/34H/350
        7012/355
79         7015/R21, 7013/591,
           9076/SP2/wide 77MHz
80         7015/990
81         9076/SP2/thin 120MHz
82         7015/R24
89         7013/595,
           9076/SP2/wide 135MHz
90         7009/C20
91         7006/42W/42T
A0         7013/J30-604,
           7013/J30-601
A1         7013/J50, 7013/J40
A3         7015/R30-604,
           7015/R30-601
A4         7015/R50, 7015/R40,
```

```
            9076/SP2/high
A6          7012/G30-604,
            7012/G30-601
A7          7012/G40
C0          7024/E30, 7024/E20
C4          7025/F30
F0          7007/N40
```

WinNT

A text string representing the type of the computer. For example, “80686”.

GBL_MACHINE_MODEL

PLATFORMS: HP-UX SunOS

HP-UX

The CPU model. This is similar to the information returned by the GBL_MACHINE metric and the “uname -m” command. However, this metric returns more information on some processors like the T500 series.

SunOS

The CPU model. This is silmilar to the information returned by the GBL_MACHINE metric and the “uname -i” command. However, this metric returns more information on some processors like the SUNW,Ultra-1

GBL_MEM_ACTIVE_VIRT_UTIL

PLATFORMS: HP-UX

The percentage of total virtual memory active at the end of the interval.

Active virtual memory is the virtual memory associated with processes that are currently on the run queue or processes that have executed recently. This is the sum of the virtual memory sizes of the data and stack regions for these processes.

GBL_MEM_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The amount of physical available memory in the system (in KB unless otherwise specified).

WinNT

Memory resident operating system code and data is not included as available memory.

GBL_MEM_CACHE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The amount of physical memory (in KB unless otherwise specified) used by the buffer cache during the interval.

HP-UX

The buffer cache is a memory pool used by the system to stage disk IO data for the driver.

SunOS

This value is obtained by multiplying the system page size times the number of buffer headers (nbuf). For example, on a SPARCstation 10 the buffer size is usually (200 (page size buffers) * 4096 (bytes/page) = 800 KB).

The buffer cache is a memory pool used by the system to cache inode, indirect block and cylinder group related disk accesses. This is different from the traditional concept of a buffer cache that also holds file system data. On Solaris 2.X, as file data is cached, accesses to it show up as virtual memory IOs. File data caching occurs through memory mapping managed by the virtual memory system, not through the buffer cache. The “nbuf” value is dynamic, but it is very hard to create a situation where the memory cache metrics change, since most systems have more than adequate space for inode, indirect block, and cylinder group data caching. This cache is more heavily utilized on NFS file servers.

AIX

This value should be minimal since most disk IOs are done through memory mapped files.

GBL_MEM_CACHE_HIT_PCT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

HP-UX

The percentage of buffer cache reads satisfied in the file system buffer cache (rather than going to disk) during the interval.

Buffer cache reads can occur as a result of a logical read (for example, file read system call), a read generated by a client, a read-ahead on behalf of a logical read or a system procedure.

This metric is obtained by measuring the number of buffered read calls that were satisfied by the data that was in the file system buffer cache. Reads to filesystem file buffers that are not in the buffer cache result in disk IO. Reads to raw IO and virtual memory IO (including memory mapped files), do not go through the filesystem buffer cache, and so are not relevant to this metric.

A low cache hit rate may indicate low efficiency of the buffer cache, either because applications have poor data locality or because the buffer cache is too small.

Overly large buffer cache sizes can lead to a memory bottleneck. The buffer cache should be sized small enough so that pageouts do not occur even when the system is busy.

However, in the case of VxFS, all memory-mapped IOs show up as page ins/page outs and are not a result of memory pressure.

SunOS

The percentage of physical reads satisfied in memory (rather than going to disk) during the interval. This includes inode, indirect block and cylinder group related disk reads, plus file reads from files memory mapped by the virtual memory IO system.

AIX

The percentage of physical reads satisfied in the file system buffer cache (rather than going to disk) during the interval.

The traditional file system buffer cache is not normally used, since files are implicitly memory mapped and the access is through the virtual memory system rather than the buffer cache. However, if a file is read as a block device (e.g /dev/hdisk1), the file system buffer cache is used, making this metric meaningful in that situation. If no IO through the buffer cache occurs during the interval, this metric is 0.

NCR Sinix

The percentage of logical reads satisfied in memory (rather than going to disk) during the interval. This includes inode, indirect block and cylinder group related disk reads, plus file reads from files memory mapped by the virtual memory IO system.

WinNT

The percentage of buffered reads satisfied in the buffer cache (rather than going to disk) during the interval.

This metric is obtained by measuring the number of buffered read calls that were satisfied by the data that was in the system buffer cache. Reads that are not in the buffer cache result in disk IO. Unbuffered IO and virtual memory IO (including memory mapped files), are not counted in this metric.

GBL_MEM_CACHE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of physical memory used by the buffer cache during the interval.

HP-UX

The buffer cache is a memory pool used by the system to stage disk IO data for the driver.

SunOS

This percentage is based on calculating the buffer cache size by multiplying the system page size times the number of buffer headers (nbuf). For example, on a SPARCstation 10 the buffer size is usually (200 (page size buffers) * 4096 (bytes/page) = 800 KB).

The buffer cache is a memory pool used by the system to cache inode, indirect block and cylinder group related disk accesses. This is different from the traditional concept of a buffer cache that also holds file system data. On Solaris 2.X, as file data is cached, accesses to it show up as virtual memory IOs. File data caching occurs through memory mapping managed by the virtual memory system, not through the buffer cache. The “nbuf” value is dynamic, but it is very hard to create a situation where the memory cache metrics change, since most systems have more than adequate space for inode, indirect block, and cylinder group data caching. This cache is more heavily utilized on NFS file servers.

AIX

This value should be minimal since most disk IOs are done through memory mapped files.

GBL_MEM_COMMIT_PCT

PLATFORMS: Win3X/95

On Windows 3.11, this is the sum of allocated swap space and allocated RAM memory expressed as a percentage of available RAM memory.

On Windows 95, this is the amount of committed virtual storage expressed as a percentage of available RAM memory.

GBL_MEM_DISCARD

PLATFORMS: Win3X/95

The total number of discards in the interval (Windows 95 only).

GBL_MEM_DISCARD_RATE

PLATFORMS: Win3X/95

The number of pages discarded from memory per second. (The pages aren't swapped to the disk because the information is already on the disk.) A high rate is an important indicator of memory pressure (Windows 95 only).

GBL_MEM_FAULT_RATE

PLATFORMS: Win3X/95

The number of page faults per second. Page faults occur when a virtual storage linear address is accessed whose corresponding real storage page is not present. A page fault may require disk IO to PAGE IN the missing virtual storage. Disk IO may also be required to PAGE OUT virtual storage already in memory to make room for the new page. Alternatively, the page may be RECLAIMED from lists of pages waiting to be paged out or reused. (In this case, no disk IO occurs.) On a 386 Enhanced Mode system with a swap file, page faulting is an important indicator of memory pressure. Page faulting is caused by a combination of memory over commitment, program memory reference patterns (working set), and the frequency with which the user swaps between applications.

GBL_MEM_FREE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The amount of memory not allocated (in KB unless otherwise specified). As this value drops, the likelihood increases that swapping or paging out to disk may occur to satisfy new memory requests.

GBL_MEM_FREE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of physical memory that was free at the end of the interval.

GBL_MEM_GDIRES_FREE

PLATFORMS: Win3X/95

The smallest value returned from a monitoring interval that represents a percentage of the Graphics Device Interface resources still free. Windows 3.11 provides only limited space in its Graphics Device Interface and User modules to hold the handles (memory identifiers) that describe window objects and other program items.

GBL_MEM_LOAD_INDEX

PLATFORMS: Win3X/95

The level of memory use according to a number (Memory Load index) obtained from the Windows 95 operating system. This number is calculated internally by the operating system. A number of 0 indicates no memory use and a number of 100 indicates full memory use. The monitor samples this value every 30 seconds and reports the maximum of the samples recorded during the measurement interval. (The calculation compares committed memory to physical memory. When committed memory is equal to physical memory, it reports 0; when committed memory is 2 times physical memory, it reports 100.)

GBL_MEM_PAGEIN

PLATFORMS: Win3X/95 DEC

The total number of page ins from the disk during the interval. This includes pages paged in from paging space and from the file system.

GBL_MEM_PAGEIN_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of KBs per second of page ins during the interval.

GBL_MEM_PAGEIN_RATE

PLATFORMS: Win3X/95 DEC

The total number of page ins per second during the interval. This includes pages paged in from paging space and from the file system.

GBL_MEM_PAGEOUT

PLATFORMS: HP-UX AIX NCR Sinix DEC SunOS Win3X/95

HP-UX AIX NCR Sinix DEC

The total number of page outs to the disk during the interval. This includes pages paged out to paging space and to the file system.

HP-UX

This metric is available on HP-UX 11.0 and beyond.

AIX

This is the same as the “page outs” value from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

AIX

This metric cannot be compared to the “po” value from the “vmstat” command. The “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_MEM_PAGEOUT) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_PAGEOUT_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

HP-UX SunOS

The total number of page outs to the disk during the interval.

This includes pages paged out to paging space and to the file system.

Win3x/95

The total number of page outs to the disk during the interval (Windows 95 only).

GBL_MEM_PAGEOUT_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of KBs (or MBs if specified) per second of page outs during the interval.

GBL_MEM_PAGEOUT_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS AIX NCR Sinix DEC WinNT

The total number of page outs to the disk per second during the interval. This includes pages paged out to paging space and to the file system.

HP-UX SunOS AIX

This is the same as the “page outs” value from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

AIX

This metric cannot be compared to the “po” value from the “vmstat” command. The “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_MEM_PAGEOUT) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_PAGEOUT_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

WinNT

This counter also includes paging traffic on behalf of the system cache to access file data for applications and so may be high when there is no memory pressure.

GBL_MEM_PAGE_FAULT_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of page faults per second during the interval.

GBL_MEM_PAGE_REQUEST

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of page requests to or from the disk during the interval.

SunOS AIX

This includes pages paged in and out to paging space. It also includes IO to files in the local file system since files are implicitly memory mapped and the IO is handled by the virtual memory system.

HP-UX SunOS AIX DEC

This is the same as the sum of the “page ins” and “page outs” values from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

AIX

This metric cannot be compared to either the “pi” or “po” values from the “vmstat” command. The “pi” value only reports the number of pages paged in from paging space, while the “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_PAGE_MEM_REQUEST) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_PAGE_REQUEST_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

GBL_MEM_PAGE_REQUEST_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of page requests to or from the disk per second during the interval.

SunOS AIX

This includes pages paged in/out to paging space. It also includes IO to files in the local file system since files are implicitly memory mapped and the IO is handled by the virtual memory system.

HP-UX SunOS AIX DEC

This is the same as the sum of the “page ins” and “page outs” values from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

AIX

This metric cannot be compared to either the “pi” or “po” values from the “vmstat” command. The “pi” value only reports the number of pages paged in from paging space, while the “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_PAGE_MEM_REQUEST) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_PAGE_REQUEST_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

HP-UX

Higher than normal rates can indicate either a memory or a disk bottleneck. Compare GBL_DISK_UTIL_PEAK and GBL_MEM_UTIL to determine which resource is more constrained. High rates may also indicate memory thrashing caused by a particular application or set of applications. Look for processes with high major fault rates to identify the culprits.

GBL_MEM_PG_SCAN_RATE

PLATFORMS: SunOS AIX NCR Sinix

SunOS NCR Sinix

The number of pages scanned per second by the pageout daemon during the interval. The clock hand algorithm is used to control page aging.

AIX

The number of pages scanned per second by the Clock Hand of the System during the interval. The clock hand algorithm is used to control page aging.

NCR Sinix

Under no memory pressure, page scan rate remains close to 0. If a process demands lots of memory and there is not a sufficient amount of memory available, the page scan daemon will increase the scan rate in order to free up memory. Also, if the Free Frame fall below the Desperation level threshold (GBL_FREE_FRAME_DESPERATION_THRESHOLD), the scan rate increases. If the Free Frame (GBL_FREE_FRAME_CURR) falls below the Low Memory Threshold (GBL_FREE_FRAME_LOWER_THRESHOLD), the system stops everything and the page scan daemon starts scanning pages at the highest speed possible to free up pages. Usually a scan rate of more than 100 is an indication of memory pressure.

GBL_MEM_PHYS

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

The amount of physical memory in the system (in KB unless otherwise specified).

HP-UX

Banks with bad memory are not counted.

Note that on some machines, the Processor Dependent Code (PDC) code uses the upper 1MB of memory and thus reports less than the actual physical memory of the system. Thus, on a system with 256MB of physical memory, this metric and dmesg(1M) might only report 267,386,880 bytes (255MB). This is all the physical memory that software on the machine can access.

SunOS AIX NCR Sinix DEC

Most systems fall in the 16 MB through 256 MB range.

WinNT

This is the total memory available to Windows NT, which may be slightly less than the total amount of physical memory present in the system. This value is also reported in the Control Panel's About Windows NT help topic.

GBL_MEM_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on memory (waiting for virtual memory disk accesses to complete) during the interval. This typically happens when processes or kernel threads are allocating a large amount of memory. It can also happen when processes or kernel threads access memory that has been paged out to disk (swap) because of overall memory pressure on the system. Note that large programs can block on VM disk access when they are initializing, bringing their text and data pages into memory. When this metric rises, it can be an indication of a memory bottleneck, especially if overall system memory utilization (GBL_MEM_UTIL) is near 100% and there is also swapout or page out activity.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on memory divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_MEM_SWAP

PLATFORMS: HP-UX AIX Sinix SunOS

HP-UX

The total number of deactivations/reactivations during the interval.

Process swapping has been replaced in HPUX 10.0 by process deactivation. Instead of swapping an entire process to a swap area, processes that place higher demands on memory resources and/or have been inactive for long periods of time are marked as deactivated. Their associated memory regions are deactivated and pages within these regions can be reused or paged out by the memory management vhand process in favor of pages belonging to processes that are not deactivated. Unlike traditional process swapping, deactivated memory pages may or may not be written out to the swap area, because a process could be reactivated before the paging occurs.

To summarize, a process swap-out in HPUX 10.0 is a process deactivation. A swap-in is a reactivation of a deactivated process. Swap metrics that report swap-out bytes now represent bytes paged out to swap areas from deactivated regions. Because these pages are pushed out over time based on memory demands, these counts are much smaller than 9.0 counts where the entire process was written to the swap area when it was swapped-out. Likewise, swap-in bytes now represent bytes paged in as a result of reactivating a deactivated process and reading in any pages that were actually paged out to the swap area while the process was deactivated.

AIX Sinix

The total number of swap ins and swap outs during the interval.

HP-UX AIX

HP-UX

This is the same as the sum of the “swap ins” and “swap outs” values from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

This metric can be compared to the sum of the “si” and “so” values from the “vmstat -S” command. The “si” value reports the number of processes swapped in (or reactivated), while the “so” value reports the number of processes swapped out (or deactivated) in HP-UX 10.0.

AIX

This is the same as the sum of the “paging space page ins” and “paging space page outs“ values from the ”vmstat -s“ command. Remember that “vmstat -s” reports cumulative counts.

This metric can be compared to the sum of the “pi” and “po” values from the “vmstat” command. The “pi” value only reports the number of pages paged in from paging space, while the “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_MEM_SWAP) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_SWAP_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

GBL_MEM_SWAPOUT_RATE

PLATFORMS: HP-UX AIX NCR Sinix

HP-UX

The number of deactivations per second during the interval.

Process swapping has been replaced in HPUX 10.0 by process deactivation. Instead of swapping an entire process to a swap area, processes that place higher demands on memory resources and/or have been inactive for long periods of time are marked as deactivated. Their associated memory regions are deactivated and pages within these regions can be reused or paged out by the memory management vhand process in favor of pages belonging to processes that are not deactivated. Unlike traditional process swapping, deactivated memory pages may or may not be written out to the swap area, because a process could be reactivated before the paging occurs.

To summarize, a process swap-out in HPUX 10.0 is a process deactivation. A swap-in is a reactivation of a deactivated process. Swap metrics that report swap-out bytes now represent bytes paged out to swap areas from deactivated regions. Because these pages are pushed out over time based on memory demands, these counts are much smaller than 9.0 counts where the entire process was written to the swap area when it was swapped-out. Likewise, swap-in bytes now represent bytes paged in as a result of reactivating a deactivated process and reading in any pages that were actually paged out to the swap area while the process was deactivated.

AIX

The number of swap outs per second during the interval.

Under no memory pressure, page scan rates remain close to zero. If a process demands memory and an insufficient amount is available, the page scan daemon will increase the scan rate in order to free up memory. Also, if the Free Frame falls below the Low Memory Threshold, the system suspends existing processes and prevents new ones from starting. At that point, the page scan daemon starts scanning pages at the highest speed possible to free up pages and the operating system attempts to steal real memory from pages unlikely to be referenced in the near future. If it fails to reach the free memory goal this way, then swap outs begin.

Usually a scan rate greater than 150 and a swap out rate greater than 100 indicates memory pressure. High swap out rates also indicate memory thrashing.

The size of the free list must be kept above the low threshold for several reasons. For example, the AIX operating system sequential prefetch algorithm requires up to 8 free frames at a time for each process performing sequential reads. Also, the Virtual Memory Management must avoid deadlocks within the operating system itself, which can occur if there were not enough space to read in a required page in order to free a page frame.

NCR Sinix

The number of swap outs per second during the interval.

HP-UX AIX

HP-UX

This is the same as the “swap outs” value from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

This metric can be compared to the “so” value from the “vmstat -S” command. The “so” value reports the number of processes swapped out (or deactivated) in HP-UX 10.0.

AIX

This is the same as the “paging space page outs” value from the “vmstat -s” command. Remember that “vmstat -s” reports cumulative counts.

This metric can be compared to the “po” value from the “vmstat” command. The “po” value only reports the number of pages paged out to paging space.

To determine the count (that is, GBL_MEM_SWAPOUT) for the current interval, subtract the previous value from the current value.

To determine the rate (that is, GBL_MEM_SWAPOUT_RATE) for the current interval, subtract the previous value from the current value and then divide by the length of the interval.

Keep in mind that whenever any comparisons are made with other tools, both tools must be interval synchronized with each other in order to be valid.

NCR

On the NCR Unix platform, swap ins and outs are normally not done. They will only be seen under situations of extreme memory pressure.

The swap metrics will normally be zero on nearly all systems. Paging rates can be a better indicator of memory pressure. A certain amount of paging activity is normal. Memory bottlenecks are shown by consistently high or higher than normal paging activity.

GBL_MEM_SWAP_1_HR_RATE

PLATFORMS: HP-UX SunOS

HP-UX

The number of deactivations/reactivations per hour during the interval.

Process swapping has been replaced in HPUX 10.0 by process deactivation. Instead of swapping an entire process to a swap area, processes that place higher demands on memory resources and/or have been inactive for long periods of time are marked as deactivated. Their associated memory regions are deactivated and pages within these regions can be reused or paged out by the memory management vhand process in favor of pages belonging to processes that are not deactivated. Unlike traditional process swapping, deactivated memory pages may or may not be written out to the swap area, because a process could be reactivated before the paging occurs.

To summarize, a process swap-out in HPUX 10.0 is a process deactivation. A swap-in is a reactivation of a deactivated process. Swap metrics that report swap-out bytes now represent bytes paged out to swap areas from deactivated regions. Because these pages are pushed out over time based on memory demands, these counts are much smaller than 9.0 counts where the entire process was written to the swap area when it was swapped-out. Likewise, swap-in bytes now represent bytes paged in as a result of reactivating a deactivated process and reading in any pages that were actually paged out to the swap area while the process was deactivated.

SunOS

The number of swap ins and swap outs per hour during the interval (on SunOS 4.1.3 only). This metric is not on SunOS 5.x as swap in and swap out statistics are counted as paging.

This metric does not necessarily indicate memory pressure, because it also records inactive processes undergoing soft swap-ins where pages are reclaimed from the freelist without generating disk activity.

GBL_MEM_SYS

PLATFORMS: HP-UX DEC

The amount of physical memory (in KB unless otherwise specified) used by the system (kernel) during the interval. System memory does not include the buffer cache.

HP-UX

This does not include some kinds of dynamically allocated kernel memory.

GBL_MEM_SYSRES_FREE

PLATFORMS: Win3X/95

The lesser of USER and Graphics Device Interface resources free. This is a significant measurement for Windows 3.11 systems. These systems often become unstable when resources fall to low values. Windows will not start new tasks and will issue “Out of Memory” messages if the free percentage falls below 10%.

GBL_MEM_SYS_AND_CACHE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of physical memory used by the system (kernel) and the buffer cache at the end of the interval.

HP-UX

This does not include some kinds of dynamically allocated kernel memory.

GBL_MEM_SYS_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of physical memory used by the system during the interval.

System memory does not include the buffer cache.

HP-UX

This does not include some kinds of dynamically allocated kernel memory.

GBL_MEM_USER

PLATFORMS: HP-UX DEC

The amount of physical memory (in KB unless otherwise specified) allocated to user code and data at the end of the interval. User memory regions include code, heap, stack, and other data areas including shared memory.

HP-UX

This also includes some kinds of dynamically allocated system memory. This does not include memory for buffer cache.

Large fluctuations in this metric can be caused by programs which allocate large amounts of memory and then either release the memory or terminate. A slow continual increase in this metric may indicate a program with a memory leak.

GBL_MEM_USERRES_FREE

PLATFORMS: Win3X/95

The smallest value returned from a monitoring interval that represents a percentage of the User resources still free. Windows 3.11 provides only limited space in its Graphics Device Interface and User modules to hold the handles (memory identifiers) that describe window objects and other program items.

GBL_MEM_USER_REFERENCED_UTIL

PLATFORMS: NCR DEC

The percent of physical memory referenced by user code and data at the end of the interval. This metric shows the percent of memory owned by user memory regions such as user code, heap, stack and other data areas including shared memory. This does not include memory for buffer cache.

GBL_MEM_USER_UNREFERENCED_UTIL

PLATFORMS: NCR DEC

The percent of physical memory not referenced by user code and data which was allocated for user code and date at the end of the interval.

GBL_MEM_USER_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percent of physical memory allocated to user code and data at the end of the interval. This metric shows the percent of memory owned by user memory regions such as user code, heap, stack and other data areas including shared memory.

HP-UX

This also includes some kinds of dynamically allocated system memory. This does not include memory for buffer cache.

Large fluctuations in this metric can be caused by programs which allocate large amounts of memory and then either release the memory or terminate. A slow continual increase in this metric may indicate a program with a memory leak.

GBL_MEM_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The percentage of physical memory in use during the interval. This includes system memory (occupied by the kernel), buffer cache and user memory.

HP-UX

This calculation is done using the byte values for physical memory and used memory, and is therefore more accurate than comparing the reported kilobyte values for physical memory and used memory.

GBL_NETWORK_SUBSYSTEM_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on the network subsystem (waiting for their network activity to complete) during the interval. This is the sum of processes or kernel threads in the LAN, NFS, and RPC wait states. This does not include processes or kernel threads blocked on SOCKT (that is, sockets) waits, as some processes or kernel threads sit idle in SOCKT waits for long periods.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on (LAN + NFS + RPC) divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_NET_BYTE_RATE

PLATFORMS: Win3X/95

The total kilobytes per second transferred on the network adapter (Windows 95 only). This measurement requires that the Microsoft Network Monitor protocol driver be installed. The metric derives from the bytes per second count for either Ethernet or Token Ring adapters. On Windows 95 systems, you can also use the GBL_RDR_BYTE_RATE and the GBL_SVR_BYTE_RATE to show client (re-director) and server network traffic.

GBL_NET_COLLISION_1_MIN_RATE

PLATFORMS: HP-UX SunOS AIX Sinix DEC

The number of collisions per minute on all network interfaces during the interval.

Collisions occur on any busy network, but abnormal collision rates could indicate a hardware or software problem.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

GBL_NET_COLLISION_PCT

PLATFORMS: HP-UX SunOS AIX Sinix DEC

The percentage of collisions to total outbound packet attempts during the interval. Outbound packet attempts include both successful packets and collisions.

A rising rate of collisions versus outbound packets is an indication that the network is becoming increasingly congested.

This metric does not currently include deferred packets.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

AIX

AIX does not support the collision count for ethernet interface. The collision count is supported for token ring (tr) and loopback (lo) interface. For more information please refer to netstat(1) man page.

GBL_NET_ERROR_1_MIN_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The number of errors per minute on all network interfaces during the interval. This rate should normally be zero or very small. A large error rate can indicate a hardware or software problem.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

NCR

Currently on NCR systems, collisions are counted as outbound errors and the collision metrics are zero. This is true with all utilities that get their metrics from the kernel, such as netstat.

GBL_NET_IN_ERROR_PCT

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The percentage of inbound network errors to total inbound packet attempts during the interval. Inbound packet attempts include both packets successfully received and those that encountered errors.

A large number of errors may indicate a hardware problem on the network. The percentage of inbound errors to total packets attempted should remain low.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

GBL_NET_IN_PACKET

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of successful packets received through all network interfaces during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

For HP-UX 10.20 and earlier releases, this is the same as the sum of the “Ipkts” column from the “netstat -i” command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the “Inbound Unicast Packets” and “Inbound Non-Unicast Packets” values from the output of the “lanadmin” utility for the network interface. Remember that “lanadmin” reports cumulative counts. For this release and beyond, “netstat -i” shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS AIX NCR DEC

This is the same as the sum of the “Ipkts” column from the “netstat -i” command for a network device. See also netstat(1).

Sinix

This is the same as the “(Total) input packets” column from the “netstat -i 5” form of the command. See netstat(1).

WinNT

For NBT connections, the packet size is defined as 1 Kbytes.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_NET_IN_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of successful packets per second received through all network interfaces during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

WinNT

For NBT connections, the packet size is defined as 1 Kbytes.

GBL_NET_OUTQUEUE

PLATFORMS: HP-UX DEC

The sum of the outbound queue lengths for all network interfaces (BYNETIF_QUEUE). This metric is derived from the same source as the Outbound Queue Length shown in the lanadmin(1M) program.

For most interfaces, the outbound queue is usually zero. When the value is non-zero over a period of time, the network may be experiencing a bottleneck. Determine which network interface has a non-zero queue and compare its traffic levels to normal. Also see if processes are blocking on network wait states.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

This metric is available on HP-UX 11.0 and beyond.

GBL_NET_OUT_ERROR_PCT

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The percentage of outbound network errors to total outbound packet attempts during the interval. Outbound packet attempts include both packets successfully sent and those that encountered errors.

NCR

Currently on NCR systems, collisions are counted as outbound errors and the collision metrics are zero. This is true with all utilities that get their metrics from the kernel, such as netstat.

The percentage of outbound errors to total packets attempted to be transmitted should remain low.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

GBL_NET_OUT_PACKET

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of successful packets sent through all network interfaces during the last interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

For HP-UX 10.20 and earlier releases, this is the same as the sum of the “Opkts” column from the “netstat -i” command for a network device. See also netstat(1).

For HP-UX 11.0 and beyond, this metric will be the same as the sum of the “Outbound Unicast Packets” and “Outbound Non-Unicast Packets” values from the output of the "lanadmin" utility for the network interface. Remember that "lanadmin" reports cumulative counts. For this release and beyond, "netstat -i" shows network activity on the logical level (IP) only.

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS AIX NCR DEC

This is the same as the sum of the "Opkts" column from the "netstat -i" command for a network device. See also netstat(1).

Sinix

This is the same as the "(Total) output packets" column from the "netstat -i 5" form of the command. See netstat(1).

WinNT

For NBT connections, the packet size is defined as 1 Kbytes.

SunOS

On Sun systems, this metric is only available on 5.X or later.

GBL_NET_OUT_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of successful packets per second sent through the network interfaces during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

WinNT

For NBT connections, the packet size is defined as 1 Kbytes.

GBL_NET_PACKET_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

The number of successful packets per second (both inbound and outbound) for all network interfaces during the interval. Successful packets are those that have been processed without errors or collisions.

HP-UX

On HP-UX 11.0 and beyond for Glance and GPM, this metric is updated at the BYNETIF_INTERVAL time. On systems with large numbers of IP addresses, the BYNETIF_INTERVAL can be greater than the sampling interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

WinNT

For NBT connections, the packet size is defined as 1 Kbytes.

GBL_NFS_CALL

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of NFS calls the local system has made as either a NFS client or server during the interval.

This includes both successful and unsuccessful calls. Unsuccessful calls are those that cannot be completed due to resource limitations or LAN packet errors.

NFS calls include create, remove, rename, link, symlink, mkdir, rmdir, statfs, getattr, setattr, lookup, read, readdir, readlink, write, writecache, null and root operations.

GBL_NFS_CALL_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix

The number of NFS calls per second the system made as either a NFS client or NFS server during the interval.

Each computer can operate as both a NFS server, and as an NFS client.

This metric includes both successful and unsuccessful calls. Unsuccessful calls are those that cannot be completed due to resource limitations or LAN packet errors.

NFS calls include create, remove, rename, link, symlink, mkdir, rmdir, statfs, getattr, setattr, lookup, read, readdir, readlink, write, writecache, null and root operations.

GBL_NUM_CPU

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of CPUs physically on the system.

SunOS

This includes all CPUs, either online or offline. The commands psrinfo(1M) and psradm(1M) allow you to check or change the status of the system CPUs.

Sinix

This includes all CPUs, either online or offline.

GBL_NUM_DISK

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of disks on the system.

HP-UX

This is a count of the number of disks on the system that have ever had activity over the cumulative collection time.

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

"Disk” refers to a physical drive (that is, “spindle”), not a partition on a drive (unless the partition occupies the entire physical disk).

SunOS AIX DEC

Only local disk devices are counted in this metric.

SunOS

On Sun systems, this metric is only available on 4.1.X.

If a CD drive is powered off, or no CD is inserted in the CD drive at boottime, the operating system does not provide performance data for that device. This can be determined by checking the “by-disk” data when provided in a product. If the CD drive has an entry in the list of active disks on a system, then data for that device is being collected.

Sinix

Only local disk devices are counted in this metric. Virtual disks are not included.

GBL_NUM_LV

PLATFORMS: SunOS Sinix

SunOS

The sum of configured logical volumes.

Sinix

The number of configured virtual disks.

GBL_NUM_NETWORK

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX SunOS AIX NCR DEC

The number of Local Area Network (LAN) interfaces on the system. This includes the loopback interface. The “netstat -i” command also displays the list of network interfaces on the system.

Sinix

The number of network interfaces on the system. This includes the Local Area Network (LAN) interfaces, Serial Software interfaces such as SLIP or PPP and Wide Area Network interfaces (WAN) such as ISDN or X.25. The “netstat -i” command also displays the list of network interfaces on the system.

GBL_NUM_USER

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

HP-UX SunOS AIX NCR Sinix

The information for this metric comes from the utmp file which is updated by the login command. For more information, read the man page for utmp. Some applications may create users on the system without using login and updating the utmp file. These users are not reflected in this count.

This metric can be a general indicator of system usage. In a networked environment, however, users may maintain inactive logins on several systems.

WinNT

The information for this metric comes from the Server Sessions counter in the Performance Libraries Server object. It is a count of the number of users using this machine as a file server.

HP-UX

The number of users logged in at the time of the interval sample. This is the same as the command “who | wc -l”, which may exceed the value in the kernel parameter “maxusers”.

Levels of remote users that are close to the configured maximum value (npty) may cause problems because login attempts can fail when that limit is reached.

SunOS AIX NCR Sinix

The number of users logged in at the time of the interval sample. This is the same as the command “who | wc -l”.

GBL_OSKERNELTYPE_INT

PLATFORMS: HP-UX

This indicates the word size of the current kernel on the system. Some hardware can load the 64-bit kernel or the 32-bit kernel.

This metric is available on HP-UX 11.0 and beyond.

GBL_OSNAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

A string representing the name of the operating system.

HP-UX

This is the same as the output from the “uname -s” command. For example, “HP-UX”.

SunOS

This is the same as the output from the “uname -s” command. For example, “SunOS”.

AIX

This is the same as the output from the “uname -s” command. For example, “AIX”.

NCR

This is the same as the output from the “uname -s” command. For example, “UNIX_SV”.

Sinix

This is the output from the “uname -s” command without the trailing OS variant name letter. For example, “SINIX” is displayed for “SINIX-N” systems.

DEC

A string representing the name of the operating system. This is the same as the output from the “uname -s” command. For example, “OSF1”.

WinNT

For example, “NT”.

GBL_OSRELEASE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS Sinix DEC

The current release of the operating system. This is the same as the output from the “uname -r” command.

AIX

The current release of the operating system in the form “V.R”, where “V” is the system version (from “uname -v”) and “R” is the release (from “uname -r”). For example, “3.2”.

NCR

The current release of the operating system. This is the same as the “RELEASE” found in /etc/.relid.

WinNT

The current release of the operating system.

GBL_OSVERSION

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS AIX NCR Sinix DEC

A string representing the version of the operating system.

HP-UX

This is the same as the output from “uname -v”.

This is an alphabetic code indicating the number of users specified by the operating system license. The mapping is documented under the “uname(2)” command.

For the Series 700/800:

```
A => two-user system
B => 16-user system
C => 32-user system
D => 64-user system
E => 8-user system
U => unlimited-users system
```

SunOS

This is the same as the output from “uname -v”.

This string is limited to 20 characters, therefore the complete version name might be truncated.

NCR

This is the same as the “VERSION” found in /etc/.relid.

DEC

The current release of the operating system. This is the same as the output from the “uname -r” command.

WinNT

A string representing the service pack installed on the operating system.

GBL_OTHER_QUEUE

PLATFORMS: HP-UX SunOS AIX

The average number of processes or kernel threads blocked on other (unknown) activities during the interval.

This includes processes or kernel threads that were started and subsequently suspended before the midaemon was started and have not been resumed, or the block state is unknown.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on OTHER divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_PARTITION_SPACE_MIN

PLATFORMS: Win3X/95

The total free space on the partition having the least amount of free space, in megabytes and reported for local, fixed disk only. The value is sampled every 30 seconds and the minimum sample in the recording interval reported.

GBL_PRI_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on PRI (waiting for their priority to become high enough to get the CPU) during the interval.

To determine if the CPU is a bottleneck, compare this metric with GBL_CPU_TOTAL_UTIL. If GBL_CPU_TOTAL_UTIL is near 100 percent and GBL_PRI_QUEUE is greater than three, there is a high probability of a CPU bottleneck.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on PRI divided by the interval time.

For example, let's assume we're using a system with eight processors. We start eight CPU intensive processes that consume almost all of the CPU resources. The approximate values shown for the CPU related queue metrics would be:

```
GBL_RUN_QUEUE = 1.0
GBL_PRI_QUEUE = 0.1
GBL_CPU_QUEUE = 1.0
```

Assume we start an additional eight CPU intensive processes. The approximate values now shown are:

```
GBL_RUN_QUEUE = 2.0
GBL_PRI_QUEUE = 8.0
GBL_CPU_QUEUE = 9.0
```

At this point, we have sixteen CPU intensive processes running on the eight processors. Keeping the definitions of the three queue metrics in mind, the run queue is 2 (that is, 16 / 8); the pri queue is 8 (only half of the processes can be active at any given time); and the cpu queue is 9 (half of the processes waiting in the cpu queue that are ready to run, plus one for the active process).

This illustrates that the run queue is the average of the 1-minute load averages for all processors; the pri queue is the number of processes or kernel threads that are blocked on “PRI” (priority); and the cpu queue is the number of processes or kernel threads in the cpu queue that are ready to run, including the processes or kernel threads using the CPU.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_PROC_RUN_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The average run time, in seconds, for processes that terminated during the interval.

GBL_PROC_SAMPLE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of process data samples that have been averaged into global metrics that are based on process samples, such as GBL_ACTIVE_PROC.

GBL_QUEUE_HISTOGRAM

PLATFORMS: HP-UX

A bar chart of the processes waiting in queues.

Shows breakout of the components of processes waiting in queues.

```
The sum of processes waiting = GBL_RUN_QUEUE
                             + GBL_DISK_SUBSYSTEM_QUEUE
                             + GBL_MEM_QUEUE
                             + GBL_NETWORK_SUBSYSTEM_QUEUE
```

ASCII and binary files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of processes waiting on a character mode terminal display.

GBL_RDR_BYTE_RATE

PLATFORMS: Win3X/95

The number of kilobytes of data per second read and written through the Microsoft Network Client redirector (Windows 95 only).

GBL_RDR_REQUEST_RATE

PLATFORMS: Win3X/95

The number of requests for data per second to be read and written through the Microsoft Network Client redirector (Windows 95 only). (This value is not reported by the Microsoft Client for NetWare).

GBL_RUN_QUEUE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The average number of “runnable” processes or kernel threads over all processors during the interval. The value shown represents the average of the 1-minute load averages for all processors.

This metric is derived from a kernel variable (avenrun) which is calculated by summing the number of runnable processes or kernel threads for each processor and averaging the samples over the last minute. Processes or kernel threads marked “runnable” include:

```
* a process or kernel thread using
  the CPU at the time of the sample
* a process or kernel thread waiting
  for the CPU at the time of the
  sample
* a process or kernel thread paused
  on a “short disk wait” at the time
  of the sample (HP-UX 10.20 and
  11.0)
```

On HP-UX 10.20 and 11.0, this metric can include processes or kernel threads which are waiting for disk IO to complete. Because of that, it is not a reliable CPU bottleneck indicator.

On HP-UX 11i, this metric does not include processes or kernel threads which are waiting for disk IO to complete.

Several standard UNIX commands, such as uptime(1), display avenrun as the “1-minute Load Average."

For example, let's assume we're using a system with eight processors. We start eight CPU intensive processes that consume almost all of the CPU resources. The approximate values shown for the CPU related queue metrics would be:

```
GBL_RUN_QUEUE = 1.0
GBL_PRI_QUEUE = 0.1
GBL_CPU_QUEUE = 1.0
```

Assume we start an additional eight CPU intensive processes. The approximate values now shown are:

```
GBL_RUN_QUEUE = 2.0
GBL_PRI_QUEUE = 8.0
GBL_CPU_QUEUE = 9.0
```

At this point, we have sixteen CPU intensive processes running on the eight processors. Keeping the definitions of the three queue metrics in mind, the run queue is 2 (that is, 16 / 8); the pri queue is 8 (only half of the processes can be active at any given time); and the cpu queue is 9 (half of the processes waiting in the cpu queue that are ready to run, plus one for the active process).

This illustrates that the run queue is the average of the 1-minute load averages for all processors; the pri queue is the number of processes or kernel threads that are blocked on “PRI” (priority); and the cpu queue is the number of processes or kernel threads in the cpu queue that are ready to run, including the processes or kernel threads using the CPU.

To determine if the CPU is a bottleneck, examine this metric along with GBL_CPU_TOTAL_UTIL and GBL_PRI_QUEUE. If GBL_CPU_TOTAL_UTIL is near 100 percent and GBL_PRI_QUEUE is greater than three, there is a high probability of a CPU bottleneck.

SunOS AIX NCR Sinix DEC

The average number of “runnable” processes during the interval.

SunOS AIX

This metric is updated by the kernel every five seconds by counting the number of processes which are in the SRUN state at the time of update. Processes in the SRUN state are in memory, ready to run, and just waiting to get the CPU. It is an average over the number of times the kernel has updated the run queue length counter during the interval.

This is the same number reported as runq-sz by the “sar -q” command.

SunOS AIX NCR Sinix DEC

GBL_RUN_QUEUE is normally a very small number. Larger than normal values for this metric indicate CPU contention among processes. This CPU bottleneck is also normally indicated by 100 percent GBL_CPU_TOTAL_UTIL. It may be OK to have GBL_CPU_TOTAL_UTIL be 100 percent if no other processes are waiting for the CPU. However, if GBL_CPU_TOTAL_UTIL is 100 percent and GBL_RUN_QUEUE is greater than the number of processors, it indicates a CPU bottleneck.

WinNT

Approximately the average Processor Queue Length during the interval.

The Processor Queue reflects a count of process threads which are ready to execute. A thread is ready to execute (in the Ready state) when the only resource it is waiting on is the processor. The Windows NT operating system itself has many system threads which intermittently use small amounts of processor time. Several low priority threads intermittently wake up and execute for very short intervals. Depending on when the collection process samples this queue, there may be none or several of these low-priority threads trying to execute. Therefore, even on an otherwise quiescent system, the Processor Queue Length can be high. High values for this metric during intervals where the overall CPU utilization (gbl_cpu_total_util) is low do not indicate a performance bottleneck. Relatively high values for this metric during intervals where the overall CPU utilization is near 100% can indicate a CPU performance bottleneck.

GBL_SLEEP_QUEUE

PLATFORMS: HP-UX SunOS

The average number of processes or kernel threads blocked on SLEEP (waiting to awaken from sleep system calls) during the interval.

A process or kernel thread enters the SLEEP state by putting itself to sleep using system calls such as sleep, wait, pause, sigpause, sigsuspend, poll and select.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on SLEEP divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_STARTED_PROC

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The number of processes that started during the interval.

GBL_SUBPROCSAMPLEINTERVAL

PLATFORMS:

The SubProcSampleInterval parameter sets the internal sampling interval of process data. This option only changes the frequency of how often the operation system process table is scanned in order to accumulate process statistics during a log interval and does not change the logging interval for process data logging. If, for example, the CPU utilization is higher than expected (possibly due to a large operation system process table), you can decrease the utilization by increasing the sampling interval.

Note: Increasing the SUBPROC sample interval (SUBPROC can be used interchangeably with SUBPROCSAMPLEINTERVAL) parameter may decrease the accuracy of application data and process data since short-lived processes (those completing within a sample interval) cannot be captured and hence logged by scopeux.

To set process subintervals to 5 (default), 10, 15, 20, 30, or 60 seconds (these are the only values allowed), you will have to enter the SUBPROC or SUBPROCSAMPLEINTERVAL sample interval parameter in your parm file. You cannot input a value lower than 5. For example, to set the interval to 15 seconds, add one of the following lines in your parm file:

```
SUBPROC=15
  or
SUBPROCSAMPLEINTERVAL=15
```

Changes made to the parm file are logged every time MeasureWare Agent is restarted. To check changes made to the SUBPROC sample interval parameter in your parm file, you can use the following command:

utility -xs -D |grep -i sub 04/23/99 13:04 Process Collection Sample SubInterval 5 seconds -> 5 seconds 04/23/99 14:31 Process Collection Sample SubInterval 5 seconds -> 15 seconds 04/23/99 14:43 Process Collection Sample SubInterval 15 seconds -> 30 seconds

Specify the full pathname of the performance tool bin directory as needed.

You can also export the GBL_SUBPROCSAMPLEINTERVAL metric from the Configuration data.

GBL_SUSPENDED_PROCS

PLATFORMS: AIX

The average number of processes which have been either marked as should be suspended (SGETOUT) or have been suspended (SSWAPPED) during the interval. Processes are suspended when the OS detects that memory thrashing is occurring. The scheduler looks for processes that have a high repage rate compared with the number of major page faults the process has done and suspends these processes.

GBL_SVR_BYTE_RATE

PLATFORMS: Win3X/95

The number of kilobytes of data per second read and written through the Microsoft Network Server (Windows 95 only).

GBL_SWAP_SPACE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS AIX NCR Sinix DEC WinNT

The total amount of potential swap space, in MB.

HP-UX

This is the sum of the device swap areas enabled by the swapon command, the allocated size of any file system swap areas, and the allocated size of pseudo swap in memory if enabled.

Note that this is potential swap space. Since swap is allocated in fixed (SWCHUNK) sizes, not all of this space may actually be usable. For example, on a 61MB disk using 2 MB swap size allocations, 1 MB remains unusable and is considered wasted space.

This is the same as (AVAIL: total) as reported by the “swapinfo -mt” command.

SunOS

This is the total amount of swap space available from the physical backing store devices (disks) plus the amount currently available from main memory.

This is the same as (used + available)/1024, reported by the “swap -s” command.

HP-UX SunOS AIX NCR Sinix DEC

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SWAP_SPACE_AVAIL_KB

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The total amount of potential swap space, in KB.

HP-UX

This is the sum of the device swap areas enabled by the swapon command, the allocated size of any file system swap areas, and the allocated size of pseudo swap in memory if enabled.

Note that this is potential swap space. Since swap is allocated in fixed (SWCHUNK) sizes, not all of this space may actually be usable. For example, on a 61MB disk using 2 MB swap size allocations, 1 MB remains unusable and is considered wasted space.

This is the same as (AVAIL: total) as reported by the “swapinfo -t” command.

SunOS

This is the total amount of swap space available from the physical backing store devices (disks) plus the amount currently available from main memory.

This is the same as (used + available)/1024, reported by the “swap -s” command.

HP-UX SunOS AIX NCR Sinix DEC

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SWAP_SPACE_FREE

PLATFORMS: Win3X/95

The amount of free virtual memory (Windows 3.11 only). This includes free heap memory in Windows 3.11 and free memory held by the DPMI host. Unlike the Windows Program Manager About box, it does not include allocated segments that are discardable.

GBL_SWAP_SPACE_RESERVED

PLATFORMS: HP-UX SunOS NCR Sinix DEC Win3X/95

HP-UX SunOS NCR Sinix DEC

The amount of swap space (in MB) reserved for the swapping and paging of programs currently executing. Process pages swapped include data (heap and stack pages), bss (data uninitialized at the beginning of process execution), and the process user area (uarea). Shared memory regions also require the reservation of swap space.

Swap space is reserved (by decrementing a counter) when virtual memory for a program is created, but swap is only used when a page or swap to disk is actually done or the page is locked in memory if swapping to memory is enabled. Virtual memory cannot be created if swap space cannot be reserved.

HP-UX

This is the same as (USED: total) as reported by the “swapinfo -mt” command.

SunOS

This is the same as used/1024, reported by the “swap -s” command.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SWAP_SPACE_USED

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC Win3X/95

HP-UX SunOS AIX NCR Sinix DEC

The amount of swap space (in MB) used.

HP-UX

“Used” indicates written to disk (or locked in memory), rather than reserved. swap space is allocated in SWCHUNK size increments (defaulting to 2 MB) which means that not all swap space may be used. For example, on a 61 MB disk using 2 MB swap size allocations, 1 MB remains unusable and is considered wasted space.

This is the same as (USED: total - reserve) as reported by the “swapinfo -mt” command.

SunOS

“Used” indicates amount written to disk (or locked in memory), rather than reserved. Swap space is reserved (by decrementing a counter) when virtual memory for a program is created.

This is the same as (bytes allocated)/1024, reported by the “swap -s” command.

Global swap space is tracked through the operating system. Device swap space is tracked through the devices. For this reason, the amount of swap space used may differ between the global and by-device metrics. Sometimes pages that are marked to be swapped to disk by the operating system are never swapped. The operating system records this as used swap space, but the devices do not, since no physical IOs occur. (Metrics with the prefix “GBL” are global and metrics with the prefix “BYSWP” are by device.)This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SWAP_SPACE_USED_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The percentage of swap space currently in use (has memory belonging to processes paged or swapped out onto it).

HP-UX

“Used” indicates written to disk or locked in memory, rather than reserved. Swap space is allocated in SWCHUNK size increments (defaulting to 2 MB) which means that not all swap space may be used. For example, on a 61 MB disk using 2 MB swap size allocations, 1 MB remains unusable and is considered wasted space.

When compared to the “swapinfo -mt” command results, this is calculated as:

```
Util = ((USED: total - reserve)
       / (AVAIL: total)) * 100
```

SunOS

Global swap space is tracked through the operating system. Device swap space is tracked through the devices. For this reason, the amount of swap space used may differ between the global and by-device metrics. Sometimes pages that are marked to be swapped to disk by the operating system are never swapped. The operating system records this as used swap space, but the devices do not, since no physical IOs occur. (Metrics with the prefix “GBL” are global and metrics with the prefix “BYSWP” are by device.)This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SWAP_SPACE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The percent of swap space available that was reserved by running processes in the interval.

On HP-UX systems, swap space must be reserved (but not allocated) before virtual memory can be created. If all of available swap is reserved, then no new processes or virtual memory can be created. Swap space locations are actually assigned (used) when a page is actually written to disk or locked in memory (pseudo swap in memory).

Note that available swap is only potential swap space. Since swap is allocated in fixed (SWCHUNK) sizes, not all of this space may actually be usable. For example, on 61 MB disk using 2 MB swap size allocations, 1 MB remains unusable and is considered wasted space. Consequently, 100 percent utilization on a single device is not always obtainable.

This is the same as (PCT USED: total) as reported by the “swapinfo -mt” command.

SunOS AIX NCR Sinix DEC

The percent of available swap space that was being used by running processes in the interval.

WinNT

This is the percentage of virtual memory, which is available to user processes, that is in use at the end of the interval. It is not an average over the entire interval. It reflects tha ratio of committed memory to the current commit limit. The limit may be increased by the operating system if the paging file is extended.

This is the same as (Committed Bytes / Commit Limit) * 100 when comparing the results to Performance Monitor.

HP-UX SunOS AIX NCR Sinix

This metric is a measure of capacity rather than performance. As this metric nears 100 percent, processes are not able to allocate any more memory and new processes may not be able to run. Very low swap utilization values may indicate that too much area has been allocated to swap, and better use of disk space could be made by reallocating some swap partitions to be user filesystems.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

GBL_SYSCALL_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The average number of system calls per second during the interval.

High system call rates are normal on busy systems, especially with IO intensive applications. Abnormally high system call rates may indicate problems such as a “hung” terminal that is stuck in a loop generating read system calls.

HP-UX

On HP-UX, system call rates affect the overhead of the midaemon.

Due to the system call instrumentation, the fork and vfork system calls are double counted. In the case of fork and vfork, one process starts the system call, but two processes exit.

Lightweight system calls, such as umask, do not show up in the GlancePlus System Calls display, but will get added to the global system call rates. If a process is being traced (debugged) using standard debugging tools (such as adb or xdb), all system calls used by that process will show up in the System Calls display while being traced.

Compare this metric to GBL_DISK_LOGL_IO_RATE to see if high system callrates correspond to high disk IO. GBL_CPU_SYSCALL_UTIL shows the CPU utilization due to processing system calls.

GBL_SYSTEM_ID

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT Win3X/95

HP-UX SunOS AIX NCR Sinix DEC WinNT

The system ID as defined in the parm file which specifies the name of the system that the data was collected on. If the ID is not specified in the parm file, the system ID defaults to the

HP-UX SunOS AIX NCR Sinix DEC

nodename displayed when you enter the command “uname -n”.

WinNT

name obtained from GetComputerName.

GBL_SYSTEM_UPTIME_HOURS

PLATFORMS: HP-UX SunOS Sinix

The time, in hours, since the last system reboot.

GBL_TERM_FIRST_RESP_DIST_1

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_10

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_2

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_3

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_4

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_5

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_6

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_7

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_8

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_DIST_9

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with first response times falling into the 10 ranges collected. GBL_TERM_FIRST_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the first response times for the interval.

GBL_TERM_FIRST_RESP_RANGE_1

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_2

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_3

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_4

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_5

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_6

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_7

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_8

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_FIRST_RESP_RANGE_9

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 first response time bins in seconds. This is for reporting of histograms showing the distribution of first response times.

GBL_TERM_IO_QUEUE

PLATFORMS: HP-UX

The average number of processes or kernel threads blocked on terminal IO (waiting for their terminal IO to complete) during the interval.

This is calculated as the accumulated time that all processes or kernel threads spent blocked on TERM (that is, terminal IO) divided by the interval time.

The Global QUEUE metrics, which are based on block states, represent the average number of process or kernel thread counts, not actual queues.

The Global WAIT PCT metrics, which are also based on block states, represent the percentage of all processes or kernel threads that were alive on the system.

No direct comparison is reasonable with the Application WAIT PCT metrics since they represent percentages within the context of a specific application and cannot be summed or compared with global values easily. In addition, the sum of each Application WAIT PCT for all applications will not equal 100% since these values will vary greatly depending on the number of processes or kernel threads in each application.

For example, the GBL_DISK_SUBSYSTEM_QUEUE values can be low, while the APP_DISK_SUBSYSTEM_WAIT_PCT values can be high. In this case, there are many processes on the system, but there are only a very small number of processes in the specific application that is being examined and there is a high percentage of those few processes that are blocked on the disk I/O subsystem.

GBL_TERM_RESP_DIST_1

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_10

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_2

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_3

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_4

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_5

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_6

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_7

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_8

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_DIST_9

PLATFORMS:

An array of 10 numbers showing the distribution of transactions done during the interval with response times falling into the 10 ranges collected. GBL_TERM_RESP_RANGE_(1-10) returns the limits for the 10 ranges. This presents a histogram distribution of the response times for the interval.

GBL_TERM_RESP_RANGE_1

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_2

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_3

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_4

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_5

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_6

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_7

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_8

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_TERM_RESP_RANGE_9

PLATFORMS:

An array of 10 numbers showing the upper limit of ranges for the 10 response time bins in seconds. This is for reporting of histograms showing the distribution of response times.

GBL_THRESHOLD_CPU

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The percent of CPU that a process must use to become interesting during an interval. The default for this threshold is “10.0” which means a process must have a value of at least 10.00% for PROC_CPU_TOTAL_UTIL to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_DISK

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

HP-UX

The rate (IOs/sec) of physical disk IOs that a process must generate to become interesting during an interval. The default for this threshold is “10.0” which means a process must have a value of at least 10.0 for PROC_DISK_PHYS_IO_RATE to exceed this threshold.

SunOS AIX

The rate of either block disk IOs or major faults that a process must generate to become interesting during an interval. Both block disk IOs and major faults are indicators of disk IO activity. The default for this threshold is “ 10.0” which means a process must have a value of at least 10.0 for PROC_DISK_BLOCK_IO_RATE or PROC_MAJOR_FAULT_RATE to exceed this threshold.

GBL_THRESHOLD_FIRSTRESP

PLATFORMS:

The number of seconds that the average transaction time to first response must exceed for a process to become interesting during an interval. The default for this threshold is “ 5.0” which means a process must have a PROC_TERM_FIRST_RESP value of 5.0 seconds or greater to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_NOKILLED

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

This is a flag specifying that terminating processes are not interesting. The flag is set by the THRESHOLD NOKILLED statement in the parm file. If this flag is set then the process will be logged only if it exceeds at least one of the thresholds. The default (blank) is for the flag to be turned off which means a terminating process will be logged in the interval it exits even if it did not exceed any thresholds during that interval. This is so that the death of a process is recorded even if it does not exceed any of the thresholds.

HP-UX

An exception to this is short-lived processes that are alive for less than one second. By default short-lived processes are not considered interesting, but there is a flag to turn on logging them, that is, THRESHOLD_SHORTLIVED.

GBL_THRESHOLD_NONEW

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

This is a flag specifying that newly created processes are not interesting. The flag is set by the THRESHOLD NONEW statement in the parm file. If this flag is set then the process will be logged only if it exceeds at least one of the thresholds. The default (blank) is for the flag to be turned off which means a new process to be logged in the interval created even if it did not exceed any thresholds during that interval. This is so that the existence of a process is recorded even if it does not exceed any of the thresholds.

HP-UX

An exception to this is short-lived processes that are alive for less than one second. By default short-lived processes are not considered interesting, but there is a flag to turn on logging them, that is, THRESHOLD_SHORTLIVED.

GBL_THRESHOLD_RESPONSE

PLATFORMS:

The number of seconds that the average transaction time to prompt must exceed for a process to become interesting during an interval.

The default for this threshold is " 30.0" which means a process must have a PROC_TERM_RESP value of 30.0 seconds or greater to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_SHORTLIVED

PLATFORMS: HP-UX

This is a flag to specify that short-lived processes are to be considered interesting. A short-lived process is one that runs for less than one second. By default, a short-lived process is not logged unless it exceeded any threshold during its short life. This is because on UNIX systems there are many of these short-lived processes that do not consume much of the system resources. Many are simply processes created to fork another process. Logging them can become quite expensive. The THRESHOLD SHORLIVED statement in the parm file can be used to turn logging of short-lived processes on.

GBL_THRESHOLD_TRANS

PLATFORMS:

The number of terminal transactions that a process must complete to become interesting during an interval. The default for this threshold is " 100" which means a process must have a value of at least 100 for PROC_TERM_TRAN to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

This metric is available on HP-UX 10.20.

GBL_THRESHOLD_WAIT_CPU

PLATFORMS: HP-UX

The percent of time that a process must spend waiting on the CPU to become interesting during an interval. The default for this threshold is " 100.0" which means a process must have a value of at least 100.0% for PROC_PRI_WAIT_PCT to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_WAIT_DISK

PLATFORMS: HP-UX

The percent of time that a process must spend waiting on the disk subsystem to become interesting during an interval. The disk subsystem includes disk transfers, memory cache, inode access, or CD-ROM disk transfers. The default for this threshold is " 100.0" which means a process must have a value of at least 100.0% for PROC_DISK_SUBSYSTEM_WAIT_PCT to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_WAIT_IMPEDE

PLATFORMS: HP-UX

The percent of time that a process must spend waiting on a semaphore to become interesting during an interval. The default for this threshold is " 100.0" which means a process must have a value of at least 100.0% for PROC_SEM_WAIT_PCT to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_THRESHOLD_WAIT_MEMORY

PLATFORMS: HP-UX

The percent of time that a process must spend waiting on access to main memory to become interesting during an interval. The default for this threshold is " 100.0" which means a process must have a value of at least 100.0% for PROC_MEM_WAIT_PCT to exceed this threshold.

All threshold values are supplied by the parm file. A process must exceed at least one threshold value in any given interval before it will be considered interesting and be logged.

GBL_TT_OVERFLOW_COUNT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

The number of new transactions that could not be measured because the Measurement Processing Daemon's (midaemon) Measurement Performance Database is full. If this happens, the default Measurement Performance Database size is not large enough to hold all of the registered transactions on this system. This can be remedied by stopping and restarting the midaemon process using the -smdvss option to specify a larger Measurement Performance Database size. The current Measurement Performance Database size can be checked using the midaemon -sizes option.

GBL_WEB_CACHE_HIT_PCT

PLATFORMS: WinNT

The ratio of cache hits to all cache requests during the interval. Cache hits occur when a file open, directory listing or service specific object request is found in the cache.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_CGI_REQUEST_RATE

PLATFORMS: WinNT

The number of CGI requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_CONNECTION_RATE

PLATFORMS: WinNT

The sum of the number of simultaneous connections to the HTTP, FTP or gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_FILES_RECEIVED_RATE

PLATFORMS: WinNT

The rate of files/sec received by the HTTP or FTP servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_FILES_SENT_RATE

PLATFORMS: WinNT

The rate of files/sec sent by the HTTP, FTP or gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_FTP_READ_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are received by FTP servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_FTP_WRITE_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are sent by FTP servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_GET_REQUEST_RATE

PLATFORMS: WinNT

The number of GET requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_GOPHER_READ_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are received by gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_GOPHER_WRITE_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are sent by gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_HEAD_REQUEST_RATE

PLATFORMS: WinNT

The number of HEAD requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_HTTP_READ_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are received by HTTP servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_HTTP_WRITE_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are sent by HTTP servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_ISAPI_REQUEST_RATE

PLATFORMS: WinNT

The number of ISAPI requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_LOGON_FAILURES

PLATFORMS: WinNT

The number of logon failures that have been made by the HTTP, FTP or gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_NOT_FOUND_ERRORS

PLATFORMS: WinNT

Number of requests that could not be satisfied by service because requested documents could not be found; typically reported as HTTP 404 error code to client.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_OTHER_REQUEST_RATE

PLATFORMS: WinNT

The number of OTHER requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_POST_REQUEST_RATE

PLATFORMS: WinNT

The number of POST requests being processed per second.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_READ_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are received by the HTTP, FTP or gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

GBL_WEB_WRITE_BYTE_RATE

PLATFORMS: WinNT

The byte rate in KB/second that data bytes are sent by the HTTP, FTP or gopher servers during the interval.

This metric is available only for Internet Information Server (IIS) 3.0 because IIS 3.0 uses the HTTP object. The GBL_WEB_* metrics are not available for IIS 4.0 because IIS 4.0 uses the Web Service object, not the HTTP object. There is a sample Extended Collection Builder policy that uses selected metrics from the Web Service object. This policy is provided with the MeasureWare Agent product.

INTERVAL

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of seconds in the measurement interval.

For the process data class this is the number of seconds the process was alive during the interval.

LV_DEVNAME_ALIAS

PLATFORMS: SunOS

The name of this volume group associated with a logical volume.

This metric is applicable only for the Veritas LVM.

LV_DIRNAME

PLATFORMS: HP-UX

The path name of this logical volume or volume/disk group.

On HP-UX 11i and beyond, data is available from VERITAS Volume Manager (VxVM). LVM (Logical Volume Manager) uses the terminology “volume group” to describe a set of related volumes. VERITAS Volume Manager uses the terminology “disk group” to describe a collection of VM disks. For additional information on VERITAS Volume Manager, see vxintro(1M).

For LVM logical volumes, this is the name used as a parameter to the lvdisplay(1M) command. For volume groups, this is the name used as a parameter to the vgdisplay(1M) command.

The entry referred to as the “/dev/vgXX/group” entry shows the internal resources used by the LVM software to manage the logical volumes.

LV_DIRNAME_ALIAS

PLATFORMS: SunOS

The absolute path name of this logical volume.

For example: Logical volume:
/dev/vx/dsk/<group_name>/<logical_volume>

This metric is applicable only for the Veritas LVM.

LV_GROUP_NAME

PLATFORMS: HP-UX SunOS

HP-UX

The name of this volume/disk group associated with a logical volume.

On HP-UX 11i and beyond, data is available from VERITAS Volume Manager (VxVM). LVM (Logical Volume Manager) uses the terminology “volume group” to describe a set of related volumes. VERITAS Volume Manager uses the terminology “disk group” to describe a collection of VM disks. For additional information on VERITAS Volume Manager, see vxintro(1M).

SunOS

The name of this volume group associated with a logical volume.

LV_LOGL_READ

PLATFORMS: HP-UX Sinix

The number of logical reads for the current logical volume during the interval.

LV_LOGL_WRITE

PLATFORMS: HP-UX Sinix

The number of logical writes to the current logical volume during the interval.

LV_READ_BYTE_RATE

PLATFORMS: HP-UX SunOS

The number of physical KB per second read from this logical volume during the interval.

Note that bytes read from the buffer cache are not included in this calculation.

SunOS

DiskSuite metadevices are not supported. This metric is reported as "na" for volume groups since it is not applicable.

LV_READ_RATE

PLATFORMS: HP-UX SunOS Sinix

HP-UX

The number of physical reads per second for this logical volume during the interval.

This may not correspond to the physical read rate from a particular disk drive since a logical volume may be composed of many disk drives or it may be a subset of a disk drive. An individual physical read from one logical volume may span multiple individual disk drives.

Since this is a physical read rate, there may not be any correspondence to the logical read rate since many small reads are satisfied in the buffer cache, and large logical read requests must be broken up into physical read requests.

SunOS

The number of physical reads per second for this logical volume during the interval.

This may not correspond to the physical read rate from a particular disk drive since a logical volume may be composed of many disk drives or it may be a subset of a disk drive. An individual physical read from one logical volume may span multiple individual disk drives.

Since this is a physical read rate, there may not be any correspondence to the logical read rate since many small reads are satisfied in the buffer cache, and large logical read requests must be broken up into physical read requests.

DiskSuite metadevices are not supported. This metric is reported as "na" for volume groups since it is not applicable.

Sinix

The number of physical reads per second for the current virtual disk during the interval.

Virtual disks function identically to traditional physical disks, but their relation to physical disks is determined from a mapping of a physical disk (or disks) to a virtual disk (vdisk(1M)). This is done by means of a virtual disk configuration file, /etc/dktab.

LV_SPACE_UTIL

PLATFORMS: SunOS Sinix HP-UX

Percentage of the logical volume file system space in use during the interval.

A value of "na" is displayed for volume groups and logical volumes which have no mounted filesystem.

LV_WRITE_BYTE_RATE

PLATFORMS: HP-UX SunOS

The number of KBs per second written to this logical volume during the interval.

SunOS

DiskSuite metadevices are not supported. This metric is reported as “na” for volume groups since it is not applicable.

LV_WRITE_RATE

PLATFORMS: HP-UX SunOS Sinix

HP-UX

The number of physical writes per second to this logical volume during the interval.

This may not correspond to the physical write rate to a particular disk drive since a logical volume may be composed of many disk drives or it may be a subset of a disk drive.

Since this is a physical write rate, there may not be any correspondence to the logical write rate since many small writes are combined in the buffer cache, and many large logical writes must be broken up.

SunOS

The number of physical writes per second to this logical volume during the interval.

This may not correspond to the physical write rate to a particular disk drive since a logical volume may be composed of many disk drives or it may be a subset of a disk drive.

Since this is a physical write rate, there may not be any correspondence to the logical write rate since many small writes are combined in the buffer cache, and many large logical writes must be broken up.

DiskSuite metadevices are not supported. This metric is reported as “na” for volume groups since it is not applicable.

Sinix

The number of physical writes per second to the current virtual disk during the interval.

Virtual disks function identically to traditional physical disks, but their relation to physical disks is determined from a mapping of a physical disk (or disks) to a virtual disk (vdisk(1M)). This is done by means of a virtual disk configuration file, /etc/dktab.

PROC_APP_ID

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The ID number of the application to which the process or kernel thread belonged during the interval.

SunOS AIX NCR Sinix DEC WinNT

The ID number of the application to which the process belonged during the interval. Application “other” always has an ID of 1. There can be up to 128 user-defined applications, which are defined in the parm file.

PROC_CPU_CSWITCH_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread spent in context switching during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_CSWITCH_UTIL

PLATFORMS: HP-UX

The percentage of time spent in context switching the current process or kernel thread during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_INTERRUPT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread spent processing interrupts during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_INTERRUPT_UTIL

PLATFORMS: HP-UX

The percentage of time that this process or kernel thread was in interrupt mode during the last interval. Interrupt mode means that interrupts were being handled while the process or kernel thread was loaded and running on the CPU. The interrupts may have been generated by any process, not just the running process, but they were handled while the process or kernel thread was running and may have had an impact on the performance of this process or kernel thread.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_NICE_TIME

PLATFORMS: HP-UX

The time, in seconds, that this niced process or kernel thread was using the CPU in user mode during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_NICE_UTIL

PLATFORMS: HP-UX

The percentage of time that this niced process or kernel thread was in user mode during the interval.

The NICE metrics include positive nice value CPU time only. Negative nice value CPU is broken out into NNICE (negative nice) metrics. Positive nice values range from 20 to 39. Negative nice values range from 0 to 19.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_NORMAL_TIME

PLATFORMS: HP-UX

The time, in seconds, that the selected process or kernel thread was in user mode at normal priority during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_NORMAL_UTIL

PLATFORMS: HP-UX

The percentage of time that this process or kernel thread was in user mode at a normal priority during the interval. “At a normal priority” means the neither rtprio or nice had been used to alter the priority of the process or kernel thread during the interval.

Normal priority user mode CPU excludes CPU used at real-time and nice priorities.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_REALTIME_TIME

PLATFORMS: HP-UX

The time, in seconds, that the selected process or kernel thread was in user mode at a realtime priority during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_REALTIME_UTIL

PLATFORMS: HP-UX

The percentage of time that this process or kernel thread was at a realtime priority during the interval. The realtime CPU is separated out to allow users to see the effect of using the realtime facilities to alter priority.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_SYSCALL_TIME

PLATFORMS: HP-UX

The time, in seconds, that this process or kernel thread spent executing system calls in system mode, excluding interrupt or context processing, during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_SYSCALL_UTIL

PLATFORMS: HP-UX

The percentage of the total CPU time this process or kernel thread spent executing system calls in system mode during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

PROC_CPU_SYS_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The CPU time in system mode in the context of the process or kernel thread during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC WinNT

The CPU time in system mode in the context of the process during the interval.

PROC_CPU_SYS_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The percentage of time that the CPU was in system mode in the context of the process or kernel thread during the interval.

SunOS AIX NCR Sinix DEC WinNT

The percentage of time that the CPU was in system mode in the context of the process during the interval.

Unlike the global and application CPU metrics, process CPU is not averaged over the number of processors on systems with multiple CPUs. Single-threaded processes can use only one CPU at a time and never exceed 100% CPU utilization.

High system mode CPU utilizations are normal for IO intensive programs. Abnormally high system CPU utilization can indicate that a hardware problem is causing a high interrupt rate. It can also indicate programs that are not using system calls efficiently.

A classic “hung shell” shows up with very high system mode CPU because it gets stuck in a loop doing terminal reads (a system call) to a device that never responds.

HP-UX

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

SunOS

Multi-threaded processes running on systems with multiple-CPUs can exceed 100% CPU utilization. The maximum percentage is 100% times the number of CPUs online.

PROC_CPU_TOTAL_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The total CPU time, in seconds, consumed by a process or kernel thread during the interval.

Total CPU time is the sum of the CPU time components for a process or kernel thread, including system, user, context switch, interrupt processing, realtime, and nice utilization values.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC WinNT

The total CPU time, in seconds, consumed by a process during the interval.

PROC_CPU_TOTAL_TIME_CUM

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The total CPU time consumed by a process or kernel thread over the cumulative collection time. CPU time is in seconds unless otherwise specified.

SunOS AIX NCR Sinix DEC WinNT

The total CPU time consumed by a process over the cumulative collection time. CPU time is in seconds unless otherwise specified.

HP-UX SunOS AIX NCR Sinix

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

DEC WinNT

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, whichever occurred last.

HP-UX SunOS AIX

```
This is calculated as
  PROC_CPU_TOTAL_TIME_CUM =
    PROC_CPU_SYS_MODE_TIME_CUM +
    PROC_CPU_USER_MODE_TIME_CUM
```

HP-UX

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_CPU_TOTAL_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The total CPU time consumed by a process or kernel thread as a percentage of the total CPU time available during the interval.

SunOS AIX NCR Sinix DEC WinNT

The total CPU time consumed by a process as a percentage of the total CPU time available during the interval.

Unlike the global and application CPU metrics, process CPU is not averaged over the number of processors on systems with multiple CPUs. Single-threaded processes can use only one CPU at a time and never exceed 100% CPU utilization.

HP-UX

Total CPU utilization is the sum of the CPU utilization components for a process or kernel thread, including system, user, context switch interrupts processing, realtime, and nice utilization values.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

SunOS

Multi-threaded processes running on systems with multiple-CPUs can exceed 100% CPU utilization. The maximum percentage is 100% times the number of CPUs online.

PROC_CPU_TOTAL_UTIL_CUM

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The total CPU time consumed by a process or kernel thread as a percentage of the total CPU time available over the cumulative collection time.

SunOS AIX NCR Sinix DEC WinNT

The total CPU time consumed by a process as a percentage of the total CPU time available over the cumulative collection time.

HP-UX SunOS AIX NCR Sinix

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

DEC WinNT

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, whichever occurred last.

Unlike the global and application CPU metrics, process CPU is not averaged over the number of processors on systems with multiple

CPUs. Single-threaded processes can use only one CPU at a time and never exceed 100% CPU utilization.

HP-UX

Total CPU utilization is the sum of the CPU utilization components for a process or kernel thread, including system, user, context switch, interrupt processing, realtime, and nice utilization values.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

SunOS

Multi-threaded processes running on systems with multiple-CPUs can exceed 100% CPU utilization. The maximum percentage is 100% times the number of CPUs online.

PROC_CPU_USER_MODE_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The time, in seconds, the process or kernel thread was using the CPU in user mode during the interval.

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC WinNT

The time, in seconds, the process was using the CPU in user mode during the interval.

PROC_CPU_USER_MODE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The percentage of time the process or kernel thread was using the CPU in user mode during the interval.

SunOS AIX NCR Sinix DEC WinNT

The percentage of time the process was using the CPU in user mode during the interval.

Unlike the global and application CPU metrics, process CPU is not averaged over the number of processors on systems with multiple CPUs. Single-threaded processes can use only one CPU at a time and never exceed 100% CPU utilization.

HP-UX

User CPU is the time spent in user mode at a normal priority, at real-time priority, and at a nice priority.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

On multi-processor systems, processes which have component kernel threads executing simultaneously on different processors could have resource utilization sums over 100%.

SunOS

Multi-threaded processes running on systems with multiple-CPUs can exceed 100% CPU utilization. The maximum percentage is 100% times the number of CPUs online.

PROC_DISK_BLOCK_IO

PLATFORMS: SunOS AIX NCR Sinix

The number of block IOs made by (or for) a process during the interval.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock

updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Block IOs refer to data transferred between disk and the file system buffer cache in block size chunks.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_IO_CUM

PLATFORMS: SunOS AIX NCR Sinix

The number of block IOs made by (or for) a process during its lifetime or over the cumulative collection time.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock

updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Block IOs refer to data transferred between disk and the file system buffer cache in block size chunks.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_IO_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block IOs per second made by (or for) a process during the interval.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Block IOs refer to data transferred between disk and the file system buffer cache in block size chunks.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_IO_RATE_CUM

PLATFORMS: SunOS AIX NCR Sinix

The average number of block IOs per second made by (or for) a process during its lifetime or over the cumulative collection time.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS

On SunOS 4.1.X, these are physical IOsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical IOsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Block IOs refer to data transferred between disk and the file system buffer cache in block size chunks.

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_READ

PLATFORMS: SunOS AIX NCR Sinix

The number of block reads made by a process during the interval.

SunOS

On SunOS 4.1.X, these are physical readsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_READ_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block reads per second made by (or for) a process during the interval.

SunOS

On SunOS 4.1.X, these are physical readsgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical readsgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_WRITE

PLATFORMS: SunOS AIX NCR Sinix

Number of block writes made by a process during the interval. Calls destined for NFS mounted files are not included.

SunOS

On SunOS 4.1.X, these are physical writesgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_BLOCK_WRITE_RATE

PLATFORMS: SunOS AIX NCR Sinix

The number of block writes per second made by (or for) a process during the interval.

SunOS

On SunOS 4.1.X, these are physical writesgenerated by file system access and network IOs generated by nfs access. If all accesses are satisfied out of the system's buffer cache, then this number will be zero. These IOs do not include raw disk access, but do include the physical IOs used by virtual memory to perform the file system access.

When a file is accessed, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

On SunOS 4.1.x, block IO counts are much higher because all file system IO is done through the buffer cache, not the virtual memory subsystem.

On Sun 5.X (Solaris 2.X or later), these are physical writesgenerated by file system access and do not include virtual memory IOs, or IOs relating to raw disk access. These are IOs for inode and superblock updates which are handled through the buffer cache. Because virtual memory IOs are not credited to the process, the block IOs tend to be much lower on SunOS 5.X than they are on SunOS 4.1.X systems.

When a file is accessed on SunOS 5.X or later, it is memory mapped by the operating system. Accesses generate virtual memory IOs. Reading a file generates block IOs as the file's inode information is cached. File writes are a combination of posting to memory mapped allocations (VM IOs) and posting updated inode information to disk (block IOs).

AIX

Note, when a file is accessed on AIX, it is memory mapped by the operating system, so accesses generate virtual memory IOs, not block IOs.

PROC_DISK_FS_IO

PLATFORMS: HP-UX

The number of file system disk IOs for a process during the interval.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access, or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical IOs in this category (they appear under virtual memory IOs).

PROC_DISK_FS_IO_RATE

PLATFORMS: HP-UX

The number of file system disk IOs per second for a process during the interval.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical IOs in this category (they appear under virtual memory IOs).

PROC_DISK_FS_READ

PLATFORMS: HP-UX

Number of file system physical disk reads made by a process or kernel thread during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical reads in this category. They appear under virtual memory reads.

"Disk" in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_FS_READ_RATE

PLATFORMS: HP-UX

The number of file system physical disk reads made by a process or kernel thread during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical reads generated by user file system access and do not include virtual memory reads, system reads (inode access), or reads relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical reads in this category. They appear under virtual memory reads.

"Disk" in this instance refers to any locally attached physical disk drives (that is, "spindles") that may hold file systems and/or swap. NFS mounted disks are not included in this list.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_FS_WRITE

PLATFORMS: HP-UX

Number of file system physical disk writes made by a process or kernel thread during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical writes generated by user file system access and do not include virtual memory writes, system writes (inode updates), or writes relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical writes in this category. They appear under virtual memory writes.

"Disk" in this instance refers to any locally attached physical disk drives (that is, "spindles") that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_FS_WRITE_RATE

PLATFORMS: HP-UX

The number of file system physical disk writes made by a process or kernel thread during the interval.

Only local disks are counted in this measurement. NFS devices are excluded.

These are physical writes generated by user file system access and do not include virtual memory writes, system writes (inode updates), or writes relating to raw disk access. An exception is user files accessed via the mmap(2) call, which does not show their physical writes in this category. They appear under virtual memory writes.

"Disk” in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_IO_CUM

PLATFORMS: HP-UX

The number of logical IOs made by (or for) a process or kernel thread over the cumulative collection time. NFS mounted disks are not included in this list.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

"Disk” refers to a physical drive (that is, “spindle”), not a partition on a drive (unless the partition occupies the entire physical disk).

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_IO_RATE_CUM

PLATFORMS: HP-UX

The average number of logical IOs per second made by (or for) a process or kernel thread over the cumulative collection time.

Only local disks are counted in this measurement. NFS devices are excluded.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

There are several reasons why logical IOs may not correspond with physical IOs. Logical IOs may not always result in a physical disk access, since the data may already reside in memory -- either in the buffer cache, or in virtual memory if the IO is to a memory mapped file. Several logical IOs may all map to the same physical page or block. In these two cases, logical IOs are greater than physical IOs.

The reverse can also happen. A single logical write can cause a physical read to fetch the block to be updated from disk, and then cause a physical write to put it back on disk. A single logical IO can require more than one physical page or block, and these can be found on different disks. Mirrored disks further distort the relationship between logical and physical IO, since physical writes are doubled.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_READ

PLATFORMS: HP-UX

The number of disk logical reads made by a process or kernel thread during the interval. Calls destined for NFS mounted files are not counted.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_READ_RATE

PLATFORMS: HP-UX

The number of logical reads per second made by (or for) a process or kernel thread during the interval. Calls destined for NFS mounted files are not counted.

Logical disk IOs are measured by counting the read system calls that are directed to disk devices. Also counted are read system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

"Disk” refers to a physical drive (that is, “spindle”), not a partition on a drive (unless the partition occupies the entire physical disk).

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_WRITE

PLATFORMS: HP-UX

Number of disk logical writes made by a process or kernel thread during the interval. Calls destined for NFS mounted files are not counted.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_LOGL_WRITE_RATE

PLATFORMS: HP-UX

The number of logical writes per second made by (or for) a process or kernel thread during the interval. NFS mounted disks are not included in this list.

“Disk” refers to a physical drive (that is, “spindle”), not a partition on a drive (unless the partition occupies the entire physical disk).

Logical disk IOs are measured by counting the write system calls that are directed to disk devices. Also counted are write system calls made indirectly through other system calls, including writev, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, send, sento, sendmsg, and ipcsend.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_PHYS_IO

PLATFORMS: HP-UX

The number of physical IOs made by (or for) a process during the interval.

PROC_DISK_PHYS_IO_CUM

PLATFORMS: HP-UX SunOS AIX

This is the total number of physical disk transfers done by a process since it was created. It is calculated as:

```
PROC_DISK_PHYS_IO_CUM = PROC_DISK_PHYS_IO_RATE_CUM
                        * PROC_RUN_TIME
```

PROC_DISK_PHYS_IO_RATE

PLATFORMS: HP-UX

The average number of physical disk IOs per second made by the process or kernel thread during the interval.

For processes which run for less than the measurement interval, this metric is normalized over the measurement interval. For example, a process ran for 1 second and did 50 IOs during its life. If the measurement interval is 5 seconds, it is reported as having done 10 IOs per second. If the measurement interval is 60 seconds, it is reported as having done 50/60 or 0.83 IOs per second.

"Disk" in this instance refers to any locally attached physical disk drives (that is, "spindles") that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_PHYS_IO_RATE_CUM

PLATFORMS: HP-UX

The number of physical disk IOs per second made by the selected process or kernel thread over the cumulative collection time.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

"Disk" in this instance refers to any locally attached physical disk drives (that is, "spindles") that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_SUBSYSTEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on the disk subsystem (waiting for its file system IOs to complete) during the interval. This includes time spent waiting in the DISK, INODE, CACHE, and CDFS wait states. It does not include processes doing raw IO to disk devices.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_DISK_SUBSYSTEM_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on the disk subsystem (waiting for its file system IOs to complete) during the interval. This includes time spent waiting in the DISK, INODE, CACHE, and CDFS wait states. It does not include processes doing raw IO to disk devices.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_DISK_SYSTEM_IO

PLATFORMS: HP-UX

Number of file system management physical disk IOs made for a process or kernel thread during the interval.

File system management IOs are the physical accesses required to obtain or update internal information about the file system structure (inode access). Accesses or updates to user data are not included in this metric.

"Disk" in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_SYSTEM_IO_RATE

PLATFORMS: HP-UX

The number of file system management physical disk IOs per second made for a process or kernel thread during the interval.

File system management IOs are the physical accesses required to obtain or update internal information about the file system structure (inode access). Accesses or updates to user data are not included in this metric.

"Disk" in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_VM_IO

PLATFORMS: HP-UX

The number of virtual memory IOs made for a process or kernel thread during the interval.

“Disk” in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_DISK_VM_IO_RATE

PLATFORMS: HP-UX

The number of virtual memory IOs per second made for a process or kernel thread during the interval.

“Disk” in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

PROC_GROUP_ID

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The effective group ID number of the process.

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

SunOS NCR Sinix DEC

The real group ID number of the process.

AIX

The effective group ID number of the process.

PROC_INTEREST

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

A field of flags indicating why the process was considered interesting enough to be logged. Scope determines the interest reason by comparing the activity of the process to the threshold criteria set in the parm file. New or Killed are treated differently, no matter what NONEW and NOKILLED options are set to, you may see an N or K flag if the process was interesting for another reason. This field consists of 12 independent columns. Each column contains a blank or a character representing a process INTEREST code as shown below.

```
Position  Char   Meaning
1         N      New Process
2         K      Killed (terminated) process
3         C      CPU percentage used exceeded threshold
```

HP-UX SunOS AIX NCR Sinix

```
4         D      Disk IOs exceeded threshold
```

HP-UX

```
5         P      Response-to-prompt time exceeded threshold
6         F      First-response time exceeded threshold
7         T      Transaction rate exceeded threshold
8         c      Wait for CPU percentage exceeded threshold
9         d      Wait for disk percentage exceeded threshold
10        m      Wait for memory percentage exceeded threshold
11        i      Wait for impede percentage exceeded threshold
12        blank  Special purpose field
```

SunOS AIX NCR Sinix WinNT

```
5         blank  Not Used
6         blank  Not Used
7         blank  Not Used
8         blank  Not Used
9         blank  Not Used
10        blank  Not Used
11        blank  Not Used
12        blank  Special purpose field
```

PROC_INTERVAL_ALIVE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The number of seconds that the process or kernel thread was alive during the interval. This may be less than the time of the interval if the process or kernel thread was new or died during the interval.

SunOS AIX NCR Sinix DEC WinNT

The number of seconds that the process was alive during the interval. This may be less than the time of the interval if the process was new or died during the interval.

PROC_IO_BYTE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The total number of KBs (unless otherwise specified) physically read or written by a process or kernel thread directly or indirectly during the interval.

Indirect IOs include paging and deactivation/reactivation activity done by the kernel on behalf of the process or kernel thread.

Direct IOs include disk, terminal, tape, and network IO, but exclude all NFS traffic.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC

The total number of physical IO KBs (unless otherwise specified) used by a process during the interval. IOs include disk, terminal, tape and network IO.

SunOS NCR Sinix

In general, counts in the MB ranges can be attributed to disk accesses and counts in the KB ranges can be attributed to terminal IO. This is useful when looking for processes with heavy disk IO activity. This may vary depending on the sample interval length.

SunOS

On Sun systems, this metric is only available on 5.X or later.

PROC_IO_BYTE_CUM

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The total number of KBs (unless otherwise specified) physically read or written by a process or kernel thread directly or indirectly over the cumulative collection time.

Indirect IOs include paging and deactivation/reactivation activity done by the kernel on behalf of the process or kernel thread.

Direct IOs include disk, terminal, tape, and network IO, but exclude all NFS traffic.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC

The total number of physical IO KBs (unless otherwise specified) used by a process over the cumulative collection time. IOs include disk, terminal, tape and network IO.

HP-UX DEC

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS

On Sun systems, this metric is only available on 5.X or later.

PROC_IO_BYTE_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The average number of KBs (unless otherwise specified) physically read or written by a process or kernel thread directly or indirectly during the interval.

Indirect IOs include paging and deactivation/reactivation activity done by the kernel on behalf of the process or kernel thread.

Direct IOs include disk, terminal, tape, and network IO, but exclude all NFS traffic.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC

The number of physical IO KBs per second used by a process during the interval. IOs include disk, terminal, tape and network IO.

SunOS NCR Sinix

In general, rates greater than 100 KB/second can be attributed to disk accesses and rates less than this can be attributed to terminal IO. This rule of thumb is useful when looking for processes with heavy disk IO activity. This rule varies depending on the sample interval length.

AIX

Certain types of disk IOs are not counted by AIX at the process level, so they are excluded from this metric.

SunOS

On Sun systems, this metric is only available on 5.X or later.

PROC_IO_BYTE_RATE_CUM

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

The average number of KBs per second physically read or written by a process or kernel thread directly or indirectly over the cumulative collection time.

Indirect IOs include paging and deactivation/reactivation activity done by the kernel on behalf of the process or kernel thread.

Direct IOs include disk, terminal, tape, and network IO, but exclude all NFS traffic.

HP-UX

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

On a threaded operating system, such as HP-UX 11.0 and beyond, process usage of a resource is calculated by summing the usage of that resource by its kernel threads. If this metric is reported for a kernel thread, the value is the resource usage by that single kernel thread. If this metric is reported for a process, the value is the sum of the resource usage by all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

SunOS AIX NCR Sinix DEC

The average number of physical IO KBs per second used by a process over the cumulative collection time. IOs include disk, terminal, tape and network IO.

HP-UX DEC

The cumulative collection time is defined from the point in time when either: a) the process or kernel thread was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS AIX NCR Sinix

The cumulative collection time is defined from the point in time when either: a) the process was first started, or b) the performance tool was first started, or c) the cumulative counters were reset (relevant only to GlancePlus), whichever occurred last.

SunOS NCR Sinix

In general, rates greater than 100 KB/second can be attributed to disk accesses and rates less than 100 KB/second can be attributed to terminal IO. This is useful when looking for processes with heavy disk IO activity. This may vary depending on the sample interval length.

SunOS

On Sun systems, this metric is only available on 5.X or later.

PROC_IPC_SUBSYSTEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on the InterProcess Communication (IPC) subsystems (waiting for its interprocess communication activity to complete) during the interval. This is the sum of processes or kernel threads in the IPC, MSG, SEM, PIPE, SOCKT (that is, sockets) and STRMS (that is, streams IO) wait states.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel

threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_IPC_SUBSYSTEM_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, the process or kernel thread was blocked on the InterProcess Communication (IPC) subsystems (waiting for its interprocess communication activity to complete) during the interval. This is the sum of processes or kernel threads in the IPC, MSG, SEM, PIPE, SOCKT (that is, sockets) and STRMS (that is, streams IO) wait states.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_LAN_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on LAN (waiting for IO over the LAN to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel

threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_LAN_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on LAN (waiting for IO over the LAN to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_MAJOR_FAULT

PLATFORMS: HP-UX SunOS AIX

HP-UX

Number of major page faults for this process or kernel thread during the interval.

Major page faults and minor page faults are a subset of vfaults (virtual faults). Stack and heap accesses can cause vfaults, but do not result in a disk page having to be loaded into memory.

SunOS AIX

Number of major page faults for this process during the interval.

PROC_MEM_RES

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The size (in KB) of resident memory for the process. This consists of text, data, stack, as well as the process' portion of shared memory regions (such as, shared libraries, text segments, and shared data).

Resident memory (RSS) is calculated as

```
RSS = sum of private region pages +
      (sum of shared region pages /
       number of references)
```

The number of references is a count of the number of attachments to the memory region. Attachments, for shared regions, may come from several processes sharing the same memory, a single process with multiple attachments, or combinations of these.

This value is only updated when a process uses CPU. Thus, under memory pressure, this value may be higher than the actual amount of resident memory for processes which are idle.

On HP-UX 10.20, the kernel instrumentation doubles the reported size of private regions. To compensate for this, the total reported RSS for each process is halved.

On HP-UX 11.0 and beyond, this metric accurately reports the resident memory for the process.

Note, a value of “na” may be shown for the swapper process.

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

SunOS AIX NCR Sinix DEC

The number of KBs of resident memory for the process. This consists of text, data, stack, as well as the process' portion of shared memory.

WinNT

The number of KBs in the working set of this process. The working set includes the memory pages touched recently by the threads of the process. If free memory in the system is above a threshold, then pages are left in the working set even if they are not in use. When free memory falls below a threshold, pages are trimmed from the working set, but not necessarily paged out to disk from memory. If those pages are subsequently referenced, they will be page faulted back into the working set. Therefore, the working set is a general indicator of the memory resident set size of this process, but it will vary depending on the overall status of memory on the system. Note that the size of the working set is often larger than the amount of pagefile space consumed (PROC_MEM_VIRT).

SunOS

A value of “na” is displayed when this information is unobtainable.

This information is not obtained for the fsflush, pageout and sched processes. It may also be unavailable for <defunct> processes.

AIX

This is the same as the RSS value shown by ps v.

Sinix

A value of “na” is displayed when this information is unobtainable.

This information is not obtained for the processes which belong to “System” priority class, such as pageout and fsflush processes. It may also not be available for <defunct> processes.

PROC_MEM_VIRT

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

This consists of the sum of the virtual set size of all private and shared memory regions used by this process. This metric is not affected by the reference count for those regions which are shared.

Note, a value of “na” may be shown for the swapper process.

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

SunOS AIX NCR Sinix DEC

The number of KBs of virtual memory allocated to the process. This consists of private text, private data, private stack and shared memory.

WinNT

The number of KBs the process has used in the paging file(s). Paging files are used to store pages of memory used by the process, such as local data, that are not contained in other files. Examples of memory pages which are contained in other files include pages storing a program's .EXE and .DLL files. These would not be kept in pagefile space. Thus, often programs will have a memory working set size (PROC_MEM_RES) larger than the size of its pagefile space.

SunOS

A value of “na” is displayed when this information is unobtainable.

This information is not obtained for the fsflush, pageout and sched processes. It may also not be available for <defunct> processes.

Sinix

A value of “na” is displayed when this information is unobtainable.

This information is not obtained for the processes which belong to “System” priority class, such as pageout and fsflush processes. It may also not be available for <defunct> processes.

PROC_MEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on memory (waiting for memory resources to become available) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_MEM_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on VM (waiting for virtual memory resources to become available) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_MINOR_FAULT

PLATFORMS: HP-UX SunOS AIX WinNT

HP-UX

Number of minor page faults for this process or kernel thread during the interval.

Major page faults and minor page faults are a subset of vfaults (virtual faults). Stack and heap accesses can cause vfaults, but do not result in a disk page having to be loaded into memory.

SunOS AIX WinNT

Number of minor page faults for this process during the interval.

PROC_NFS_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on NFS (waiting for network file system IO to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_NFS_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on NFS (waiting for its network file system IO to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel

threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_OTHER_IO_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on “other IO” during the interval.

Other IO” includes all IO directed at a device (connected to the local computer) which is not a terminal or LAN. Examples of “other IO” devices are local printers, tapes, instruments, and disks. Time waiting for character (raw) IO to disks is included in this measurement. Time waiting for file system buffered IO to disks will typically been seen as IO or CACHE wait. Time waiting for IO to NFS disks is reported as NFS wait.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_OTHER_IO_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on other IO during the interval.

Other IO" includes all IO directed at a device (connected to the local computer) which is not a terminal or LAN. Examples of "other IO" devices are local printers, tapes, instruments, and disks. Time waiting for character (raw) IO to disks is included in this measurement. Time waiting for file system buffered IO to disks will typically been seen as IO or CACHE wait. Time waiting for IO to NFS disks is reported as NFS wait.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_OTHER_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on other (unknown) activities during the interval.

This includes processes or kernel threads that were started and subsequently suspended before the midaemon was started and have not been resumed, or the block state is unknown.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on

Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_OTHER_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on other (unknown) activities during the interval.

This includes processes or kernel threads that were started and subsequently suspended before the midaemon was started and have not been resumed, or the block state is unknown.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_PARENT_PROC_ID

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The parent process' PID number.

HP-UX

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

PROC_PRI

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT

HP-UX

The dispatch priority of a process or kernel thread at the end of the interval.

Whenever the priority is changed for the selected process or kernel thread, the new value will not be reflected until the process or kernel thread is reactivated if it is currently idle (for example, SLEEPing).

The lower the value, the more the process or kernel thread is likely to be dispatched. Values between zero and 127 are considered to be “real-time” priorities, which the kernel does not adjust. Values above 127 are normal priorities and are modified by the kernel for load balancing. Some special priorities are used in the HP-UX kernel and subsystems for different activities. These values are described in /usr/include/sys/param.h. Priorities less than PZERO 153 are not signalable.

Note, many network-related programs such as inetd, biod, and rlogind run at priority 154 which is PPIPE. Just because they run at this priority does not mean they are using pipes. By examining the open files, you can determine if a process or kernel thread is using pipes.

For HP-UX 10.0 and later releases, priorities between -32 and -1 can be seen for processes or kernel threads using the Posix Real-time Schedulers. When specifying a Posix priority, the value entered must be in the range from 0 through 31, which the system then remaps to a negative number in the range of -1 through -32. Refer to the rtsched man pages for more information.

On a threaded operating system, such as HP-UX 11.0 and beyond, this metric represents a kernel thread characteristic. If this metric is reported for a process, the value for its last executing kernel thread is given. For example, if a process has multiple kernel threads and kernel thread one is the last to execute during the interval, the metric value for kernel thread one is assigned to the process.

SunOS

The dispatch priority of a process at the end of the interval. The lower the value the more the process is likely to be dispatched.

On Sun Systems this metric is only available on 4.1.X.

AIX

The dispatch priority of a process at the end of the interval. The lower the value the more the process is likely to be dispatched. Values for priority range from 0 to 127. Processes running at priorities less than PZERO (40) are not signalable.

NCR Sinix

The global priority of a process at the end of the interval. This is used for process scheduling order. The higher the value the more likely the process is to be dispatched.

The global priorities are table driven. The tables can be customized on each system. For more information, see dispadmin(1M). The default priority ranges and their classes are 0 to 59 for “Timesharing” (TS), 60 to 99 for “System” (SYS) and 100 to 159 for “Real-Time” (RT). Process global priorities are also displayed with the “-c” option of the “ps” command.

“Timesharing” class priority is designed to provide good response time to interactive processes and good throughput to CPU-bound processes. The scheduler switches CPU allocation frequently enough to provide good response time, but not so frequently that it spends too much time doing the switching.

The “System” class uses a fixed-priority to run kernel processes such as servers and housekeeping processes like page daemon. The “System” class is reserved for use by the kernel; users cannot add or remove a process from the “System” class.

The “Real-Time” class uses a fixed-priority scheduling policy so that critical processes can run in predetermined order. “Real-Time” priorities never change except when a user requests a change.

WinNT

The current base priority of this process. The higher the value the more the process or thread is likely to be dispatched. Values for priority range from 0 to 31. Values of 16 and above are considered to be “realtime” priorities. Threads within a process can raise and lower their own base priorities relative to the process's base priority.

PROC_PRI_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time during the interval the process or kernel thread was blocked on priority (waiting for its priority to become high enough to get the CPU).

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_PRI_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on PRI (waiting for its priority to become high enough to get the CPU) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_PROC_ID

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX SunOS AIX NCR Sinix DEC

The PID number of a process, used by the kernel to identify a process.

WinNT

The process ID number of this process, used to uniquely identify it. Process numbers are reused, so they only identify a process for it's lifetime.

HP-UX

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

PROC_PROC_NAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

The process program name. It is limited to 16 characters.

HP-UX SunOS AIX NCR Sinix DEC

This is derived from the 1st parameter to the exec(2) system call.

HP-UX

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

WinNT

The “System Idle Process” is not reported by the MeasureWare Agent since Idle is a process that runs to occupy the processors when they are not executing other threads. Idle has one thread per processor.

PROC_REVERSE_PRI

PLATFORMS: SunOS NCR Sinix

SunOS NCR Sinix

The process priority in a range of 0 to 127, with a lower value interpreted as a higher priority. Since priority ranges can be customized, this metric provides a standardized way of interpreting priority that is consistent with other versions of UNIX. This is the same value as reported in the PRI field by the ps command when the -c option is not used.

SunOS

On Sun systems, this metric is only available on 5.X or later.

PROC_RUN_TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC WinNT

HP-UX

The elapsed time since a process or kernel thread started, in seconds.

This metric is less than the interval time if the process or kernel thread was not alive during the entire first or last interval.

On a threaded operating system such as HP-UX 11.0 and beyond, this metric is available for a process or kernel thread.

SunOS AIX NCR Sinix DEC WinNT

The elapsed time since a process started, in seconds.

This metric is less than the interval time if the process was not alive during the entire first or last interval.

PROC_SEM_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on semaphores (waiting on a semaphore operation to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_SEM_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on semaphores (waiting on a semaphore operation to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_SLEEP_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on SLEEP (waiting to awaken from sleep system calls) during the interval.

A process or kernel thread enters the SLEEP state by putting itself to sleep using system calls such as sleep, wait, pause, sigpause, sigsuspend, poll and select.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_SLEEP_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on SLEEP (waiting to awaken from sleep system calls) during the interval.

A process or kernel thread enters the SLEEP state by putting itself to sleep using system calls such as sleep, wait, pause, sigpause, sigsuspend, poll and select.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_STOP_HISTOGRAM

PLATFORMS: HP-UX

A bar chart of the process stop reasons.

Shows breakout of the percent of time processes were in different wait states.

```
The percent of time = PROC_CPU_TOTAL_UTIL_CUM
                    + PROC_PRI_WAIT_PCT
                    + PROC_DISK_SUBSYSTEM_WAIT_PCT
                    + PROC_MEM_WAIT_PCT
                    + PROC_LAN_WAIT_PCT
```

ASCII and binary files contain a line of ASCII characters that make up one row of a printed histogram. This can be a quick way to get a graphical view of the wait states on a character mode terminal display.

PROC_STOP_REASON

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX

A text string describing what caused the process or kernel thread to stop executing. For example, if the process is waiting for a CPU while higher priority processes are executing, then its block reason is PRI. A complete list of block reasons follows:

SunOS AIX NCR Sinix DEC

A text string describing what caused the process to stop executing. For example, if the process is waiting for a CPU while higher priority processes are executing, then its block reason is PRI. A complete list of block reasons follows:

HP-UX

```
String   Reason for Process Block
```

```
------------------------------------

CACHE   Waiting at the buffer cache
        level trying to lock down a
        buffer cache structure, or
        waiting for an IO operation
        to or from a buffer cache to
        complete. File system access
        will block on IO more often
        than CACHE on HP-UX 11.x.
        Processes using the file
        system to access block
        devices will block on CACHE
        on HP-UX 10.20.

CDFS    Waiting for CD-ROM file
        system node structure
        allocation or locks while
        accessing a CD-ROM device
        through the file system.

died    Process terminated during
        the interval.

DISK    Waiting for an IO operation
        to complete at the logical
        device manager or disk
        driver level.  Waits from
        raw disk IO and diagnostic
        requests can be seen here.
        Buffered IO requests can
        also block on DISK, but will
        more often be seen waiting
        on “IO”.  CDFS access will
        block on “CDFS”.  Virtual
        memory activity will block
        on “VM”.

GRAPH   Waiting for a graphics card
        or framebuf semaphore
        operation.

INODE   Waiting while accessing
        an inode structure.  This
        includes inode gets and
        waiting due to inode locks.

IO      Waiting for IO to local
        disks, printers, tapes, or
```

	instruments to complete (above the driver, but below the buffer cache). Both file system and raw disk access can block in this state. CDFS access will block on “CDFS”. Virtual memory activity will block on “VM”.
IPC	Waiting for a process or kernel thread event (that is, waiting for a child to receive a signal). This includes both inter and intra process or kernel thread operations, such as IPC locks, kernel thread mutexes, and database IPC operations. System V message queue operations will block on “MESG”, while semaphore operations will block on “SEM”.
JOBCL	Waiting for tracing or debug resume, or job control start. A background process incurs this block when attempting to write to a terminal set with “stty tostop”. On HP-UX 11i, scheduler activation threads (user threads) will show this block.
LAN	Waiting for a network IO completion. This includes waiting on the LAN hardware and low level LAN device driver. It does not include waiting on the higher level network software such as the streams based transport or NFS, which has its own stop state.
MESG	Waiting for a System V message queue operation such as msgrcv or msgsnd.

new	Process was created (via the fork/vfork system calls) during the interval.
NFS	Waiting for a Networked File System request to complete. This includes both NFS V2 and V3 requests. This does not include stops where kernel threads or deamons are waiting for a NFS event or request (such as biod or nfsd). These will block on SLEEP to show they are waiting for some activity.
NONE	Zombie process - waiting to die.
OTHER	The process was started before the midaemon was started and has not been resumed, or the block state is unknown.
PIPE	Waiting for operations involving pipes. This includes opening, closing, reading, and writing using pipes. Named pipes will block on PIPE.
PRI	Waiting because a higher priority process is running, or waiting for a spinlock or alpha semaphore.
RPC	Waiting for remote procedure call operations to complete. This includes both NFS and DCE RPC requests.
SEM	Waiting for a System V semaphore operation (such as semop, semget, or semctl) or waiting for a memory mapped file semaphore operation (such as msem_init or msem_lock).

SLEEP	Waiting because the process put itself to sleep using system calls such as sleep, wait, pause, sigpause, poll, sigsuspend and select. This is the standard stop reason for idle system daemons.
SOCKT	Waiting for an operation to complete while accessing a device through a socket. This is used primarily in networking code and includes all protocols using sockets (X25, UDP, TCP, and so on).
STRMS	Waiting for an operation to complete while accessing a “streams” device. This is the normal stop reason for kernel threads and daemons waiting for a streams event. This includes the network transport and pseudo terminal IO requests. For example, waiting for a read on a streams device or waiting for an internal streams synchronization.
SYSTM	Waiting for access to a system resource or lock. These resources include data structures from the LVM, VFS, UFS, JFS, and Disk Quota subsystems. “SYSTM” is the “catch-all” wait state for blocks on system resources that are not common enough or long enough to warrant their own stop state.
TERM	Waiting for a non-streams terminal transfer (tty or pty).
VM	Waiting for a virtual memory

```
            operation to complete, or
            waiting for free memory, or
            blocked while creating/
            accessing a virtual memory
            structure.
```

For a process or kernel thread currently running, the last reason it was stopped before obtaining the CPU is shown.

On HP-UX 11.0 and beyond, mikslp.text (located in /opt/perf/lib) contains the blocking functions and their corresponding block states for use by midaemon.

On a threaded operating system, such as HP-UX 11.0 and beyond, this metric represents a kernel thread characteristic. If this metric is reported for a process, the value for its last executing kernel thread is given. For example, if a process has multiple kernel threads and kernel thread one is the last to execute during the interval, the metric value for kernel thread one is assigned to the process.

SunOS 5.X

```
String   Reason for Process Block
------------------------------------

died     Process terminated during
         the interval.
new      Process was created (via the
         exec() system call) during
         the interval.
NONE     Process is ready to run.  It
         is not apparent that the
         process is blocked.
OTHER    Waiting for a reason not
         decipherable by the
         measurement software.
PMEM     Waiting for more primary
         memory.
PRI      Process is on the run queue.
SLEEP    Waiting for an event to
         complete.
TRACE    Received a signal to stop
         because parent is tracing
         this process.
ZOMB     Process has terminated and
         the parent is not waiting.
```

On SunOS 5.X, instead of putting the scheduler to sleep and waking it up, the kernel just stops and continues the scheduler as needed. This is done by changing the state of the scheduler to ws_stop, which is when you see the TRACE state. This is for efficiency and happens every clock tick so the “sched” process will always appear to be in a “TRACE” state.

SunOS 4.1.X

```
String    Reason for Process Block
------------------------------------

directed  Stopped or waiting for an
          event other than save
          flag, page wait, swap out
          or physical IO.
Disk      Waiting for physical IO.
NONE      Waiting for process
          creation.
PRI       Process is running.
VM        Waiting for a save flag,
          page wait, swap out or
          physical IO.
```

AIX

```
String    Reason for Process Block
------------------------------------

died    Process terminated during
        the interval.
LOCK    Waiting either for
        serialization or phys lock.
new     Process was created (via the
        exec() system call) during
        the interval.
NONE    Process is ready to run.  It
        is not apparent that the
        process is blocked.
OTHER   Waiting for a reason not
        decipherable by the
        measurement software.
PRI     Process is on the run queue.
SLEEP   Waiting for an event to
        complete.
TIMER   Waiting for the timer.
TRACE   Received a signal to stop
        because parent is tracing
        this process.
VM      Waiting for a virtual memory
        operation to complete.
ZOMB    Process has terminated and
        the parent is not waiting.
```

NCR Sinix

```
String    Reason for Process Block
------------------------------------

died     Process terminated during
         the interval.
new      Process was created (via
         the exec() system call)
         during the interval.
NONE     Process is ready to run.  It
         is not apparent that the
         process is blocked.
OTHER    Waiting for a reason not
         decipherable by the
         measurement software.
PMEM     Waiting for more primary
         memory.
PRI      Process is on the run queue.
SLEEP    Waiting for an event to
         complete.
TRACE    Received a signal to stop
         the process for tracing.
         This will occur when a
         process is stopped waiting
         on the tty device after
         having been backgrounded,
         or when the process is
         suspended by a debugger, or
         when a privileged process is
         accessing its proc structure
         to get process information.
ZOMB     Process has terminated and
         the parent is not waiting.
```

PROC_SYS_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on system resources during the interval.

These resources include data structures from the LVM, VFS, UFS, JFS, and Disk Quota subsystems. “SYSTM” is the “catch-all” wait state for blocks on system resources that are not common enough or long enough to warrant their own stop state.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_SYS_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on SYSTM (that is, system resources) during the interval.

These resources include data structures from the LVM, VFS, UFS, JFS, and Disk Quota subsystems. “SYSTM” is the “catch-all” wait state for blocks on system resources that are not common enough or long enough to warrant their own stop state.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_TERM_IO_WAIT_PCT

PLATFORMS: HP-UX

The percentage of time the process or kernel thread was blocked on terminal IO (waiting for its terminal IO to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation.

A percentage of time spent in a wait state is calculated as the time a kernel thread (or all kernel threads of a process) spent waiting in this state, divided by the alive time of the kernel thread (or all kernel threads of the process) during the interval.

If this metric is reported for a kernel thread, the percentage value is for that single kernel thread. If this metric is reported for a process, the percentage value is calculated with the sum of the wait and alive times of all of its kernel threads.

For example, if a process has 2 kernel threads, one sleeping for the entire interval and one waiting on terminal input for the interval, the process wait percent values will be 50% on Sleep and 50% on Terminal. The kernel thread wait values will be 100% on Sleep for the first kernel thread and 100% on Terminal for the second kernel thread.

For another example, consider the same process as above, with 2 kernel threads, one of which was created half-way through the interval, and which then slept for the remainder of the interval. The other kernel thread was waiting for terminal input for half the interval, then used the CPU actively for the remainder of the interval. The process wait percent values will be 33% on Sleen and 33% on Terminal (each one third of the total alive time). The kernel thread wait values will be 100% on Sleep for the first kernel thread and 50% on Terminal for the second kernel thread.

PROC_TERM_IO_WAIT_TIME

PLATFORMS: HP-UX

The time, in seconds, that the process or kernel thread was blocked on terminal IO (waiting for its terminal IO to complete) during the interval.

On a threaded operating system, such as HP-UX 11.0 and beyond, process wait time is calculated by summing the wait times of its kernel threads. If this metric is reported for a kernel thread, the value is the wait time of that single kernel thread. If this metric is reported for a process, the value is the sum of the wait times of all of its kernel threads. Alive kernel threads and kernel threads that have died during the interval are included in the summation. For multi-threaded processes, the wait times can exceed the length of the measurement interval.

PROC_THREAD_COUNT

PLATFORMS: HP-UX SunOS AIX WinNT

The total number of kernel threads for the current process.

PROC_TTY

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX AIX

The controlling terminal for a process. This field is blank if there is no controlling terminal. This is the same as the “TTY” field of the ps command.

SunOS NCR Sinix

The controlling terminal for a process. This field is blank if there is no controlling terminal.

The controlling terminal name is found by searching the directories provided in the /etc/ttysrch file. See man page ttysrch(4) for details. The matching criteria field ("M", "F" or "I" values) of the ttysrch file is ignored. If a terminal is not found in one of the ttysrch file directories, the following directories are searched in the order here: "/dev", "/dev/pts", "/dev/term" and "dev/xt". When a match is found in one of the "/dev" subdirectories, "/dev/" is not displayed as part of the terminal name. If no match is found in the directory searches, the major and minor numbers of the controlling terminal are displayed.

In most cases, this value is the same as the "TTY" field of the ps command.

DEC

The controlling terminal for a process. This field is blank if there is no controlling terminal.

The controlling terminal name is found by searching the directories provided in the /etc/ttysrch file. See man page ttysrch(4) for details.

In most cases, this value is the same as the "TTY" field of the ps command.

HP-UX

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

PROC_USER_NAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The login account of a process (from /etc/passwd).

If more than one account is listed in /etc/passwd with the same user ID (uid) field, the first one is used. If an account cannot be found that matches the uid field, then the uid number is returned. This would occur if the account was removed after a process was started.

HP-UX

This metric is specific to a process. If this metric is reported for a kernel thread, the value for its associated process is given.

RECORD_TYPE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

ASCII string that identifies the record. Possibilities include GLOB for global 5 minute detail, GSUM for global hourly summary, APPL for application 5 minute detail, ASUM for application hourly summary, CONF for configuration, TRAN for transaction tracker detail, TSUM for transaction tracker summary.

HP-UX SunOS AIX NCR Sinix WinNT

PROC for process 1 minute detail, DISK for disk device 5 minute detail, DSUM for disk device summary.

HP-UX

VOLS for logical volume disk detail, VSUM for logical volume disk summary.

Sinix

VOLS for virtual disk detail, VSUM for virtual disk summary.

TBL_BUFFER_CACHE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix Win3X/95

HP-UX

The size (in KB unless otherwise specified) of the file system buffer cache on the system. These buffers are used for all file system IO operations, as well as all other block IO operations in the system (exec, mount, inode reading, and some device drivers).

If dynamic buffer cache is enabled, the system allocates a percentage of available memory not less than dbc_min_pct nor more than dbc_max_pct, depending on the system needs at any given time. On systems with a static buffer cache, this value will remain equal to bufpages, or not less than dbc_min_pct nor more than dbc_max_pct.

SunOS

The size (in KB unless otherwise specified) of the file system buffer cache on the system.

This value is obtained by multiplying the system page size times the number of buffer headers (nbuf). For example, on a SPARCstation 10 the buffer size is usually (200 (page size buffers) * 4096 (bytes/page) = 800 KB).

NOTE: (for systems with VERITAS File System installed) Veritas implemented their Direct I/O feature in their file system to provide mechanism for bypassing the UNIX system buffer cache while retaining the on disk structure of a file system. The way in which Direct I/O works involves the way the system buffer cache is handled by the UNIX OS.

Once the VERITAS file system returns with the requested block, instead of copying the content to a system buffer page, it instead copies the block into the application's buffer space. That's why if you have installed vxfs on your system

The buffer cache is a memory pool used by the system to cache inode, indirect block and cylinder group related disk accesses. This is different from the traditional concept of a buffer cache that also holds file system data. On Solaris 2.X, as file data is cached, accesses to it show up as virtual memory IOs. File data caching occurs through memory mapping managed by the virtual memory system, not through the buffer cache. The “nbuf” value is dynamic, but it is very hard to create a situation where the memory cache metrics change, since most systems have more than adequate space for inode, indirect block, and cylinder group data caching. This cache is more heavily utilized on NFS file servers.

AIX

The size (in KB unless otherwise specified) of the file system buffer cache on the system. This cache is used for all block IO.

NCR

The size (in KB unless otherwise specified) of the file system buffer cache on the system.

This value is obtained by multiplying the system page size times the number of buffer headers (nbuf).

Sinix

The maximum amount of memory that can be allocated to the buffer cache.

On the systems with more than 800Mb of physical memory (GBL_MEM_PHYS) the buffer cache memory size is fixed to 8Mb.

On the systems with less than 800Mb but more that 160Mb of physical memory (GBL_MEM_PHYS) the amount of memory allocated to buffer cache (TBL_BUFFER_CACHE_USED) can vary between 1% of the physical memory and the top limit of 8Mb.

On the systems with less than 160Mb of physical memory (GBL_MEM_PHYS) the amount of memory allocated to buffer cache (TBL_BUFFER_CACHE_USED) can vary between 1% of the physical memory and TBL_BUFFER_CACHE_HWM.

TBL_BUFFER_CACHE_USED

PLATFORMS: HP-UX AIX NCR Sinix

HP-UX AIX NCR

The size (in KB unless otherwise specified) of the sum of the currently used buffers.

Sinix

The size (in KB unless otherwise specified) of the sum of the currently allocated buffers.

HP-UX AIX

This is normally greater than the amount requested due to internal fragmentation of the buffer cache. Since this is a cache, it is normal for it to be filled. The buffer cache is used to stage all block IOs to disk.

HP-UX

In a dynamic buffer cache configuration, this metric is always equal to TBL_BUFFER_CACHE_AVAIL. With dynamic buffer cache, the system allocates a percentage of available memory not less than dbc_min_pct nor more than dbc_max_pct, depending on the system needs at any given time.

On systems with a static buffer cache, this value will remain equal to bufpages, or not less than dbc_min_pct nor more than dbc_max_pct. With a static buffer cache, this metric shows the amount of memory within the configured size that is actually used.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_FILE_LOCK_AVAIL

PLATFORMS: HP-UX NCR Sinix

The configured number of file or record locks that can be allocated on the system. Files and/or records are locked by calls to lockf(2).

TBL_FILE_LOCK_USED

PLATFORMS: HP-UX SunOS NCR Sinix DEC

The number of file or record locks currently in use. One file can have multiple locks. Files and/or records are locked by calls to lockf(2).

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_FILE_LOCK_UTIL

PLATFORMS: HP-UX NCR Sinix

The percentage of configured file or record locks currently in use.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_FILE_TABLE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix

HP-UX AIX

The configured maximum number of the file table entries used by the kernel to manage open file descriptors.

HP-UX

This is the sum of the “nfile” and “file_pad” values used in kernel generation.

SunOS

The number of entries in the file cache. This is a size. All entries are not always in use. The cache size is dynamic.

Entries in this cache are used to manage open file descriptors. They are reused as files are closed and new ones are opened. The size of the cache will go up or down in chunks as more or less space is required in the cache.

AIX

The file table entries are dynamically allocated by the kernel if there is no entry available. These entries are allocated in chunks.

NCR

The number of entries in the file table. This number is dynamic on NCR. At boot time it is initialized to the NFILE parameter in the /etc/conf/cf.d/stune file. If number of open files on the system exceeds NFILE, the number is increased dynamiclly. There is no way to extract this metric from kernel memory, so it cannot be accurately displayed.

Sinix

The number of entries in the file table. The file table would be dynamically inreased by the kernel if the system were to run out of free entries.

TBL_FILE_TABLE_USED

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX AIX NCR DEC

The number of entries in the file table currently used by file descriptors.

SunOS

The number of file cache entries currently used by file descriptors.

Sinix

The number of entries in the file table currently used by file descriptors.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_FILE_TABLE_UTIL

PLATFORMS: HP-UX SunOS AIX Sinix

The percentage of file table entries currently used by file descriptors.

Sinix

Since the file table is dynamical SINIX systems always show 100% utilization.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_INODE_CACHE_AVAIL

PLATFORMS: HP-UX SunOS NCR Sinix DEC

HP-UX

The configured number of entries in the incore inode table on the system. The table contains recently closed inodes as well as open inodes.

Most file system activity is done through inodes which are stored on disk. The kernel keeps a memory cache of active and recently accessed inodes to reduce disk IOs. When a file is accessed through a pathname, the kernel converts the pathname to an inode number and obtains the inode information from the cache. If the inode entry is not in the cache, the inode is read from disk into the inode cache.

The number of used entries in the inode cache is usually at or near the capacity. This does not necessarily indicate that the size (configurable by the ninode parameter) is too small because the table contains recently used inodes and inodes referenced by entries in the directory name cache. When an inode cache entry is required, the kernel first tries to use a free entry. If one does not exist, inactive entries referenced by the directory name cache are used. If after freeing inode entries only referenced by the directory name cache does not create enough free space, the inode table can be then considered full causing the “inode: table is full” message to appear on the console. If this occurs, increase the ninode parameter. Low directory name cache hit ratios may also indicate an underconfigured inode cache (ninode).

The default formula for the ninode size is:

```
ninode = ((nproc+16+maxusers)+32+
         (2*npty)+(4*num_clients))
```

SunOS NCR Sinix DEC

The number of entries in the inode cache. This is a size. All entries are not always in use. The cache size is dynamic.

Entries in this cache are reused as files are closed and new ones are opened. The size of the cache will go up or down in chunks as more or less space is required in the cache.

Sinix

Note that this numbers apply only to the “ufs” file system and do not cover the Veritas (“vxfs”) file system.

Inodes are used to store information about files within the file system. Every file has at least two inodes associated with it (one for the directory and one for the file itself). The information stored in an inode includes the owners, timestamps, size, and an array of indices used to translate logical block numbers to physical sector numbers. There is a separate inode maintained for every view of a file, so if two processes have the same file open, they both use the same directory inode, but separate inodes for the file.

TBL_INODE_CACHE_USED

PLATFORMS: HP-UX SunOS NCR Sinix DEC

HP-UX

The number of “non-free” inodes currently used.

Since the inode table contains recently closed inodes as well as open inodes, the table often appears to be fully utilized. When a new entry is needed, one can usually be found by reusing one of the recently closed inode entries.

Most file system activity is done through inodes which are stored on disk. The kernel keeps a memory cache of active and recently accessed inodes to reduce disk IOs. When a file is accessed through a pathname, the kernel converts the pathname to an inode number and obtains the inode information from the cache. If the inode entry is not in the cache, the inode is read from disk into the inode cache.

The number of used entries in the inode cache is usually at or near the capacity. This does not necessarily indicate that the size (configurable by the ninode parameter) is too small because the table contains recently used inodes and inodes referenced by entries in the directory name cache. When an inode cache entry is required, the kernel first tries to use a free entry. If one does not exist, inactive entries referenced by the directory name cache are used. If after freeing inode entries only referenced by the directory name cache does not create enough free space, the inode table can be then considered full causing the “inode: table is full” message to appear on the console. If this occurs, increase the ninode parameter. Low directory name cache hit ratios may also indicate an underconfigured inode cache (ninode).

The default formula for the ninode size is:

```
ninode = ((nproc+16+maxusers)+32+
          (2*npty)+(4*num_clients))
```

SunOS NCR Sinix DEC

The number of inode cache entries currently in use.

Sinix

Note that this numbers apply only to the “ufs” file system and do not cover the Veritas (“vxfs”) file system.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_MSG_TABLE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The configured maximum number of message queues that can be allocated on the system. A message queue is allocated by a program using the msgget(2) call.

HP-UX AIX

Also refer to ipcs(1).

SunOS

InterProcess Communication facilities are dynamically loadable. If the amount available is zero, this facility was not loaded when data collection began, and its data is not obtainable. The data collector is unable to determine that a facility has been loaded once data collection has started. If you know a new facility has been loaded, restart the data collection, and the data for that facility will be collected. See ipcs(1) to report on interprocess communication resources.

TBL_MSG_TABLE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The percentage of configured message queues currently in use.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_PROC_TABLE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The configured maximum number of the proc table entries used by the kernel to manage processes. This number includes both free and used entries.

HP-UX

This is set by the NPROC value during system generation.

AIX

AIX has a “dynamic” proc table, which means the avail has been set higher than should ever be needed.

TBL_PROC_TABLE_UTIL

PLATFORMS: HP-UX SunOS NCR Sinix DEC

The percentage of proc table entries currently used by processes.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_SEM_TABLE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX SunOS AIX NCR DEC

The configured number of semaphore identifiers (sets) that can be allocated on the system.

SunOS

InterProcess Communication facilities are dynamically loadable. If the amount available is zero, this facility was not loaded when data collection began, and its data is not obtainable. The data collector is unable to determine that a facility has been loaded once data collection has started. If you know a new facility has been loaded, restart the data collection, and the data for that facility will be collected. See ipcs(1) to report on interprocess communication resources.

Sinix

The configured number of semaphores that can be allocated on the system. This limits both the number of semaphore identifiers (sets) and the number of semaphores.

TBL_SEM_TABLE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

HP-UX SunOS AIX NCR DEC

The percentage of configured semaphores identifiers currently in use.

Sinix

The percentage of configured semaphores currently in use.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TBL_SHMEM_TABLE_AVAIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The configured number of shared memory segments that can be allocated on the system.

SunOS

InterProcess Communication facilities are dynamically loadable. If the amount available is zero, this facility was not loaded when data collection began, and its data is not obtainable. The data collector is unable to determine that a facility has been loaded once data collection has started. If you know a new facility has been loaded, restart the data collection, and the data for that facility will be collected. See ipcs(1) to report on interprocess communication resources.

TBL_SHMEM_TABLE_UTIL

PLATFORMS: HP-UX SunOS AIX NCR Sinix DEC

The percentage of configured shared memory segments currently in use.

This metric is updated every 30 seconds or the sampling interval, whichever is greater.

TIME

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The local time of day for the start of the interval. The time is an ASCII field in hh:mm 24-hour format. This field will always contain 5 characters in ASCII files. The two subfields (hh, mm) will contain a leading zero if the value is less than 10. This metric is extracted from GBL_STATTIME, which is obtained using the time() system call at the start of the interval.

This field responds to language localization.

In binary files this field contains four byte size subfields. The most significant byte contains the hour, the next most significant byte contains the minute, then the seconds and finally the tenths of a second. The left two bytes can be isolated by dividing by 65536. HHMM = TIME/65536. Then HOUR = HHMM/256 and MINUTE = HHMM mod 256.

TTBIN_TRANS_COUNT_1

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_10

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_2

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_3

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_4

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_5

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_6

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_7

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_8

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_TRANS_COUNT_9

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions in this range during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_1

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_10

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_2

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_3

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_4

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_5

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_6

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_7

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_8

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TTBIN_UPPER_RANGE_9

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) for this bin.

There are a maximum of nine user defined transaction response time bins (TTBIN_UPPER_RANGE). The last bin, which is not specified in ttd.conf, is the overflow bin and will always have a value of -2 (overflow). It should be noted that the values specified in the ttd.conf file cannot exceed 2147483.6, which is the number of seconds in 24.85 days. If the user specifies any values greater than 2147483.6, the numbers reported for those bins or Service Level Objective (SLO) will be -2.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_ABORT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of aborted transactions during the last interval for this transaction.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_ABORT_WALL_TIME_PER_TRAN

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The average time, in seconds, per aborted transaction during the last interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_APP_NAME

PLATFORMS: HP-UX SunOS AIX Sinix WinNT

The registered ARM Application name.

TT_APP_TRAN_NAME

PLATFORMS: HP-UX

A concatenation of TT_APP_NAME and TT_NAME. This provides a way to uniquely identify a specific transaction. The field is limited to 60 characters.

TT_CLIENT_ADDRESS

PLATFORMS: HP-UX SunOS Sinix WinNT

The correlator address. This is the address where the child transaction originated.

TT_CLIENT_ADDRESS_FORMAT

PLATFORMS: HP-UX SunOS Sinix WinNT

The correlator address format. This shows the protocol family for the client network address. Refer to the ARM API Guide for the list and description of supported address formats.

TT_CLIENT_TRAN_ID

PLATFORMS: HP-UX SunOS Sinix WinNT

A numerical ID that uniquely identifies the transaction class in this correlator.

TT_COUNT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions during the last interval for this transaction.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_CPU_TOTAL_TIME_PER_TRAN

PLATFORMS: HP-UX

The average total CPU time, in seconds, consumed by each completed instance of the transaction during the interval.

Total CPU time is the sum of the CPU time components for a process or kernel thread, including system, user, context switch, interrupt processing, realtime, and nice utilization values.

Per-transaction performance resource metrics represent an average for all completed instances of the given transaction during the interval.

If there are no completed transaction instances during an interval, then there are no resources accounted, even though there may be in-progress transactions using resources which have not completed. Resource metrics for in-progress transactions will be shown in the interval after they complete (that is, after the process has called arm_stop).

If there is only one completed transaction instance during an interval, then the resources attributed to the transaction will represent the resources used by the process between its call to arm_start and arm_stop, even if arm_start was called before the current interval. Thus, the resource usage time or wall time per transaction can exceed the current collection interval time.

If there are several completed transaction instances during an interval for a given transaction, then the resources attributed to the transaction will represent an average for all completed instances during the interval. To obtain the total accumulated resource consumption for all completed transactions during an interval, multiply the resource metric by the number of completed transaction instances during the interval (TT_COUNT).

TT_DISK_LOGL_IO_PER_TRAN

PLATFORMS: HP-UX

The average number of logical IOs made by (or for) each completed instance of the transaction during the interval. NFS mounted disks are not included in this list.

“Disk” refers to a physical drive (that is, “spindle”), not a partition on a drive (unless the partition occupies the entire physical disk).

Logical disk IOs are measured by counting the read and write system calls that are directed to disk devices. Also counted are read and write system calls made indirectly through other system calls, including readv, recvfrom, recv, recvmsg, ipcrecvcn, recfrom, writev, send, sento, sendmsg, and ipcsend.

Per-transaction performance resource metrics represent an average for all completed instances of the given transaction during the interval.

If there are no completed transaction instances during an interval, then there are no resources accounted, even though there may be in-progress transactions using resources which have not completed. Resource metrics for in-progress transactions will be shown in the interval after they complete (that is, after the process has called arm_stop).

If there is only one completed transaction instance during an interval, then the resources attributed to the transaction will represent the resources used by the process between its call to arm_start and arm_stop, even if arm_start was called before the current interval. Thus, the resource usage time or wall time per transaction can exceed the current collection interval time.

If there are several completed transaction instances during an interval for a given transaction, then the resources attributed to the transaction will represent an average for all completed instances during the interval. To obtain the total accumulated resource consumption for all completed transactions during an interval, multiply the resource metric by the number of completed transaction instances during the interval (TT_COUNT).

TT_DISK_PHYS_IO_PER_TRAN

PLATFORMS: HP-UX

The average number of physical disk IOs per second made by each completed instance of the transaction during the interval.

For transactions which run for less than the measurement interval, this metric is normalized over the measurement interval. For example, a transaction ran for 1 second and did 50 IOs during its life. If the measurement interval is 5 seconds, it is reported as having done 10 IOs per second. If the measurement interval is 60 seconds, it is reported as having done 50/60 or 0.83 IOs per second.

“Disk” in this instance refers to any locally attached physical disk drives (that is, “spindles”) that may hold file systems and/or swap. NFS mounted disks are not included in this list.

Since this value is reported by the drivers, multiple physical requests that have been collapsed to a single physical operation (due to driver IO merging) are only counted once.

Per-transaction performance resource metrics represent an average for all completed instances of the given transaction during the interval.

If there are no completed transaction instances during an interval, then there are no resources accounted, even though there may be in-progress transactions using resources which have not completed. Resource metrics for in-progress transactions will be shown in the interval after they complete (that is, after the process has called arm_stop).

If there is only one completed transaction instance during an interval, then the resources attributed to the transaction will represent the resources used by the process between its call to arm_start and arm_stop, even if arm_start was called before the current interval. Thus, the resource usage time or wall time per transaction can exceed the current collection interval time.

If there are several completed transaction instances during an interval for a given transaction, then the resources attributed to the transaction will represent an average for all completed instances during the interval. To obtain the total accumulated resource consumption for all completed transactions during an interval, multiply the resource metric by the number of completed transaction instances during the interval (TT_COUNT).

TT_FAILED

PLATFORMS: HP-UX SunOS AIX Sinix WinNT

The number of Failed transactions during the last interval for this transaction name.

TT_INFO

PLATFORMS: HP-UX SunOS AIX Sinix WinNT

The registered ARM Transaction Information for this transaction.

TT_NAME

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The registered transaction name for this transaction.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_NUM_BINS

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of distribution ranges.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_SLO_COUNT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The number of completed transactions that violated the defined Service Level Objective (SLO) by exceeding the SLO threshold time during the interval.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_SLO_PERCENT

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The percentage of transactions which violate service level objectives.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_SLO_THRESHOLD

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The upper range (transaction time) of the Service Level Objective (SLO) threshold value. This value is used to count the number of transactions that exceed this user-supplied transaction time value.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_TERM_TRAN_1_HR_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

For this transaction name, the number of completed transactions calculated to a 1 hour rate. For example, if you completed five of these transactions in a 5 minute window, the rate is 60 transactions per hour.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_TRAN_1_MIN_RATE

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

For this transaction name, the number of completed transactions calculated to a 1 minute rate. For example, if you completed five of these transactions in a 5 minute window, the rate is one transaction per minute.

SunOS

On Sun systems, this metric is only available on 5.X or later.

TT_TRAN_ID

PLATFORMS: HP-UX SunOS AIX Sinix WinNT

The registered ARM Transaction ID for this transaction class as returned by arm_getid(). A unique transaction id is returned for a unique application id (returned by arm_init), tran name, and meta data buffer contents.

TT_UNAME

PLATFORMS: HP-UX SunOS AIX Sinix WinNT

The registered ARM Transaction User Name for this transaction.

If the arm_init function has NULL for the appl_user_id field, then the user name is blank. Otherwise, if “*” was specified, then the user name is displayed.

For example, to show the user name for the armsample1 program, use:

```
appl_id = arm_init(“armsample1”,“*”,0,0,0);
```

To ignore the user name for the armsample1 program, use:

```
appl_id = arm_init(“armsample1”,NULL,0,0,0);
```

TT_USER_MEASUREMENT_AVG
TT_USER_MEASUREMENT_AVG_2

PLATFORMS: HP-UX SunOS Sinix WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

HP-UX WinNT

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

SunOS Sinix

If the measurement type is a counter, this metric returns the average counter differences over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_AVG_2

PLATFORMS: HP-UX SunOS Sinix WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

HP-UX WinNT

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

SunOS Sinix

If the measurement type is a counter, this metric returns the average counter differences over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_AVG_3

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

TT_USER_MEASUREMENT_AVG_4

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

TT_USER_MEASUREMENT_AVG_5

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

TT_USER_MEASUREMENT_AVG_6

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the average counter differences of the transaction or transaction instance during the last interval. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this returns the average of the values passed on any ARM call for the transaction or transaction instance during the last interval.

TT_USER_MEASUREMENT_COUNT
TT_USER_MEASUREMENT_COUNT_2

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_COUNT_2

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_COUNT_3

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_COUNT_4

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_COUNT_5

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_COUNT_6

PLATFORMS: HP-UX WinNT

This returns the total number of times the associated user defined metric (UDM) was sampled during the last interval.

TT_USER_MEASUREMENT_MAX TT_USER_MEASUREMENT_MAX_2

PLATFORMS: HP-UX SunOS Sinix WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MAX_2

PLATFORMS: HP-UX SunOS WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MAX_3

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MAX_4

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MAX_5

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MAX_6

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the highest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the highest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN
TT_USER_MEASUREMENT_MIN_2

PLATFORMS: HP-UX SunOS Sinix WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN_2

PLATFORMS: HP-UX SunOS WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN_3

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN_4

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN_5

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns "na".

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_MIN_6

PLATFORMS: HP-UX WinNT

If the measurement type is a numeric or a string, this metric returns “na”.

If the measurement type is a counter, this metric returns the lowest measured counter value over the life of the transaction or transaction instance. The counter value is the difference observed from a counter between the start and the stop (or last update) of a transaction.

If the measurement type is a gauge, this metric returns the lowest value passed on any ARM call over the life of the transaction or transaction instance.

TT_USER_MEASUREMENT_NAME TT_USER_MEASUREMENT_NAME_2

PLATFORMS: HP-UX SunOS Sinix WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_USER_MEASUREMENT_NAME_2

PLATFORMS: HP-UX SunOS WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_USER_MEASUREMENT_NAME_3

PLATFORMS: HP-UX WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_USER_MEASUREMENT_NAME_4

PLATFORMS: HP-UX WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_USER_MEASUREMENT_NAME_5

PLATFORMS: HP-UX WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_USER_MEASUREMENT_NAME_6

PLATFORMS: HP-UX WinNT

The name of the user defined transactional measurement. The length of the string complies with the ARM 2.0 standard, which is 44 characters long (there are 43 usable characters since this is a NULL terminated character string).

TT_WALL_TIME_PER_TRAN

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The average transaction time, in seconds, during the last interval for this transaction.

SunOS

On Sun systems, this metric is only available on 5.X or later.

YEAR

PLATFORMS: HP-UX SunOS AIX NCR Sinix WinNT Win3X/95

The year, including the century, the data in this record was captured. This metric will contain 4 digits, such as 1997.

Glossary

alarm

A signal that an event has occurred. The signal can be either a notification or an automatically triggered action. The event can be a pre-defined threshold that is exceeded, a network node in trouble, and so on. Alarm information can be sent to a PerfView analysis system and to OpenView and IT/Operations. Alarms can also be identified in historical log file data.

alarmgen (alarm generator)

The service that handles the communication of alarm information. It consists of the agdbserver and the agdb database that it manages. The agdb database contains of list of PerfView analysis notes (if any) to which alarms are communicated, and various on/off flags that are set to define when and where the alarm information is sent.

alarmdef file

The file containing the alarm definitions in which alarm conditions are specified.

alert

A message sent when alarm conditions or conditions in an IF statement have been met.

analysis software

Analysis software analyzes system performance data.

The optional PerfView product provides a central window from which you can monitor, manage, and troubleshoot the performance of all networked systems in your computing environment, as well as analyze historical data from MeasureWare Agent systems. With PerfView, you view incoming alarms from all monitored systems on your network and view graphs of a systems' performance data to help you diagnose and resolve performance problems quickly.

application

A user-defined group of related processes or program files. Applications are defined so that performance software can collect performance metrics for and report on the combined activities of the processes and programs.

available memory

Available memory is that part of physical memory not allocated by the kernel. This includes the buffer cache, user allocated memory, and free memory.

backtrack

Backtracking allows the large data structures used by the Virtual Memory Manager (VMM) to be pageable. It is a method of safely allowing the VMM to handle page faults within its own critical sections of code.

Examples of backtracking are:

- A process page faults.
- The VMM attempts to locate the missing page via its External Page table (XPT).
- The VMM page faults due to the required XPT itself having been paged out.
- The VMM safely saves enough information on the stack to restart the process at its first fault.
- Normal VMM pagein/out routines are used to recover the missing XPT.
- The required XPT is now present, so the missing page is located and paged-in.
- The process continues normal execution at the original page fault.

bad call
A failed NFS server call. Calls fail due to lack of system resources (lack of virtual memory) and network errors.

biod
A daemon process responsible for asynchronous block IO on the NFS client. It is used to buffer read-ahead and write-behind IOs.

block IO
Buffered reads and writes. Data is held in the buffer cache, then transferred in fixed-size blocks. Any hardware device that transmits and receives data in blocks is a block-mode device. Compare with character mode.

block IO buffer
A buffer used to store data being transferred to or from a block-mode device through file system input and output, as opposed to character-mode or raw-mode devices.

block IO operation
Any operation being carried out on a block-mode device (such as read, write, or mount).

block size
The size of the primary unit of information used for a file system. It is set when a file system is created.

blocked state
The reason for the last recorded process block. Also called blocked-on state.

bottleneck
A situation that occurs when a system resource is constrained by demand that exceeds its capability. The resource is said to be "bottlenecked." A bottleneck causes system performance to degrade. A primary characteristic of a bottleneck is that it does not occur in all resources at the same time; other resources may instead be underutilized.

buffer
A memory storage area used to temporarily hold code or data until used for input/output operations.

buffer cache
An area of memory that mediates between application programs and disk drives. When a program writes data, it is first placed in the buffer cache, then delivered to the disk at a later time. This allows the disk driver to perform IO operations in batches, minimizing seek time.

buffer header
Entries used by all block IO operations to point to buffers in the file system buffer cache.

buffer pool
See buffer cache.

bytes in/out (DCE)
The number of bytes passed and returned as arguments by a DCE Remote Procedure

Call.

cache
See buffer cache.

cache efficiency
The extent to which buffered read and read-ahead requests can be satisfied by data already in the cache.

cache hit
Read requests that are satisfied by data already in the buffer cache. See also cache efficiency.

character mode
The mode in which data transfers are accomplished byte-by-byte, rather than in blocks. Printers, plotters, and terminals are examples of character-mode devices. Also known as raw mode. Compare with block IO.

child process
A new process created at another active process' request through a fork or vfork system call. The process making the request becomes the parent process.

client
A system that requests a service from a server. In the context of diskless clusters, a client uses the server's disks and has none of its own. In the context of NFS, a client mounts file systems that physically reside on another system (the Network File System server).

clock hand algorithm
The algorithm used by the page daemon to scan pages.

clock hand cycle
The clock hand algorithm used to control paging and to select pages for removal from system memory. When page faults and/or system

demands cause the free list size to fall below a certain level, the page replacement algorithm starts the clock hand and it cycles through the page table.

cluster

One or more work stations linked by a local area network (LAN) but having only one root file system.

cluster server process

(CSPs). A special kernel process that runs in a cluster and handles requests from remote cnodes.

cnode

The client on a diskless system. The term cnode is derived from "client node."

collision

Occurs when the system attempts to send a packet at the same time that another system is attempting a send on the same LAN. The result is garbled transmissions and both sides have to resubmit the packet. Some collisions occur during normal operation.

context switch

The action of the dispatcher (scheduler) changing from running one process to another. The scheduler maintains algorithms for managing process switching, mostly directed by process priorities.

CPU

Central Processing Unit. The part of a computer that executes program instructions.

CPU queue

The average number of processes in the "run" state awaiting CPU scheduling, which includes processes short waited for IOs. This is calculated from GBL-RUN-QUEUE and the number of times this metric is updated. This is also a measure of how busy the system's CPU resource is.

cyclical redundancy check

(CRC). A networking checksum protocol used to detect transmission errors.

cylinder

The tracks of a disk accessible from one position of the head assembly.

cylinder group

In the filesystem, a collection of cylinders on a disk drive grouped together for the purpose of localizing information.

The filesystem allocates inodes and data blocks on a per-cylinder-group basis.

daemon

A process that runs continuously in the background but provides important system services.

data class

A particular category of data collected by a data collection process. Single-instance data classes, such as the global class, contain a single set of metrics that appear only once in any data source. Multiple-instance classes, such as the application class, may have many occurrences in a single data source, with the same set of metrics collected for each occurrence of the class.

data locality

The location of data relative to associated data. Associated data has good data locality if it is located near one another, because accesses are limited to a small number of pages and the data is more likely to be in memory. Poor data locality means associated data must be obtained from different data pages.

data point

A specific point in time displayed on a performance graph where data has been summarized every five, fifteen, or thirty minutes, or every hour, two hours or one day.

data segment

A section of memory reserved for storing a process' static and dynamic data.

data source

A data source consists of one or more classes of data in a single scopeux or DSI log file set. For example, the default MeasureWare Agent data source, SCOPE, is a scopeux or scopeNT log file set consisting of global data. See also repository server.

data source integration (DSI)

Enables MeasureWare Agent to receive and log data from external sources such as applications, databases, networks, and other operating systems.

deactivated pages out

Pages from deactivated process regions that are moved from memory to the swap area. These pages are swapped out only when they are needed by another active process.

When a process becomes reactivated, the pages are moved from the swap area back to memory.

default

An option that is automatically selected or chosen by the system.

deferred packet

A deferred packet occurs when the network hardware detects that the LAN is already in use. Rather than incur a collision, the outbound packet transmission is delayed until the LAN is available.

device driver

A collection of kernel routines and data structures that handle the lowest levels of input and output between a peripheral device and executing processes. Device drivers are part of the kernel.

device file
A special file that permits direct access to a hardware device.

device swap space
Space devoted to swapping.

directory name lookup cache
The directory name lookup cache (DNLC) is used to cache directory and file names. When a file is referenced by name, the name must be broken into its components and each component's inode must be looked up. By caching the component names, disk IOs are reduced.

diskless cluster server
A system that supports disk activity for diskless client nodes.

diskless file system buffer
A buffer pool that is used only by the diskless server for diskless cluster traffic.

dispatcher
A module of the kernel responsible for allocating CPU resources among several competing processes.

DSI log file
A log file, created by MeasureWare Agent’s DSI (data source integration) programs, that contains self-describing data..

empty space
The difference between the maximum size of a log file and its current size.

error (DCE)
Unsuccessful client requests of a DCE server. Causes include datacomm errors, client call rejects and other execution errors that result in a failed request.

error (LAN)
Unsuccessful transmission of a packet over a local area network (LAN). Inbound errors are typically checksum errors. Outbound errors are typically local hardware problems.

exec fill page
When a process is 'EXECed' the working segments of the process are marked as copy on write. Only when segments change are they copied into a separate segment private to the process that is modifying the page.

extract program
The MeasureWare Agent’s program that allows you to extract data from raw or previously extracted log files, summarize it, and write it to extracted log files. It also lets you export data for use by analysis programs and other tools.

extracted log file
A MeasureWare Agent log file containing a user-defined subset of data extracted (copied) from raw or previously extracted log file. It is

formatted for optimal access by PerfView. Extracted log files are also used for archiving performance data.

file IO

IO activity to a physical disk. It includes file system IOs, system IOs to manage the file system, both raw and block activity, and excludes virtual memory management IOs.

file lock

A file lock guarantees exclusive access to an entire file, or parts of a file.

file system

The organization and placement of files and directories on a hard disk. The file system includes the operating system software's facilities for naming the files and controlling access to these files.

file system activity

Access calls (read, write, control) of file system block IO files contained on disk.

file system swap

File system space identified as available to be used as swap. This is a lower performance method of swapping as its operations are processed through the file system.

file table

The table contains inode descriptors used by the user file descriptors for all open files. It is set to the maximum number of files the system can have open at any one time.

fork

A system call that enables a process to duplicate itself into two identical processes - a parent and a child process. Unlike the vfork system call, the child process produced does not have access to the parent process' memory and control.

free list

The system keeps a list of free pages on the system. Free list points to all the pages that are marked free.

free memory

Memory not currently allocated to any user process or to the kernel.

GlancePlus

An online diagnostic tool that displays current performance data directly to a user terminal or workstation. It is designed to assist you in identifying and troubleshooting system performance problems as they occur.

global

A qualifier implying the whole system. Thus "global metrics" are metrics that describe the activities and states of each system. Similarly, application metrics describe application activity; process metrics describe process activity.

global log file
The raw log file, `logglob`, where the scopeNT collector places summarized measurements of the system-wide workload.

idle biod
The number of inactive NFS daemons on a client.

idle
The state in which the CPU is idle when it is waiting for the dispatcher (scheduler) to provide processes to execute.

inode
A reference pointer to a file. This reference pointer contains a description of the disk layout of the file data and other information, such as the file owner, access permissions, and access times. Inode is a contraction of the term 'index node'.

inode cache
An in memory table containing up-to-date information on the state of a currently referenced file.

interarrival time
The time between client requests of a DCE server.

interesting process
A filter mechanism that allows the user to limit the number of process entries to view. A process becomes interesting when it is first created, when it ends, and when it exceeds user-defined thresholds for CPU use, disk use, response time, and so on.

interrupt
High priority interruptions of the CPU to notify it that something has happened. For example, a disk IO completion is an interrupt.

intervals
Specific time periods during which performance data is gathered.

ioctl
A system call that provides an interface to allow processes to control IO or pseudo devices.

IO done
The Virtual Memory Management (VMM) system reads and writes from the disk and keeps track of how many IOs are completed by the system. Since IOs are asynchronous, they are not completed immediately.

Sometimes IOs done can be higher than IO starts, since some of the IOs that are started in the previous interval can be completed.

IO start
The Virtual Memory Management (VMM) system reads and writes from the disk and keeps track of how many IOs are started by the system. Since IOs are async, they are not completed immediately.

InterProcess Communication (IPC)
Communication protocols used between processes.

kernel

The core of the operating system. It is the code responsible for managing the computer's resources and performing functions such as allocating memory. The kernel also performs administrative functions required for overall system performance.

kernel table

An internal system table such as the Process Table or Text Table. A table's configured size can affect system behavior.

last measurement reset

When you run a performance product, it starts collecting performance data. Cumulative metrics begin to accumulate at this time. When you reset measurement to zero, all cumulative metrics are set to zero and averages are reset so their values are calculated beginning with the next interval.

load average

A measure of the CPU load on the system. The load average is defined as an average of the number of processes running and ready to run, as sampled over the previous one-minute interval of system operation. The kernel maintains this data.

lock miss

The Virtual Memory Management (VMM) system locks pages for synchronization purposes. If the lock has to be broken for any reason that is considered a lock miss. Usually this is a very small number.

logappl (application log file)

The raw log file that contains summary measurements of processes in each user-defined application.

logdev (device log file)

The raw log file that contains measurements of individual device (disk, logical volume, network interface) performance.

logglob (global log file)

The raw log file that contains measurements of the system-wide, or global, workload.

logindex

The raw log file that contains information required for accessing data in the other log files.

logproc (process log file)

The raw log file that contains measurements of selected interesting processes.

logtran (transaction log file)

The raw log file that contains measurements of Transaction Tracker data.

log files

Performance measurement files that contain either raw or extracted log file data.

logical IO
A read or write system call to a file system to obtain data. Because of the effects of buffer caching, this operation may not require a physical access to the disk if the buffer is located in the buffer cache.

macro
A group of instructions that you can combine into a single instruction for the application to execute.

major fault
A page fault requiring an access to disk to retrieve the page.

measurement interface
A set of proprietary library calls used by the performance applications to obtain performance data.

memory pressure
A situation that occurs when processes are requesting more memory space than is available.

memory swap space
The part of physical memory allocated for swapping.

memory thrashing
See thrashing.

message buffer pool
A cache used to store all used message queue buffers on the system.

message queue
The messaging mechanism allows processes to send formatted data streams to arbitrary processes. A message queue holds the buffers from which processes read the data.

message table
A table that shows the maximum number of message queues allowed for the system.

metric
A specific measurement that defines performance characteristics.

midaemon
A process that monitors system performance and creates counters from system event traces that are read and displayed by performance applications.

minor fault
A page fault that is satisfied by a memory access (the page was not yet released from memory).

mount/unmount
The process of adding or removing additional, functionally-independent file systems to or from the pool of available file systems.

NFS call
A physical Network File System (NFS) operation a system has received or processed.

NFS client
A node that requests data or services from other nodes on the network.

NFS Logical IO
A logical I/O request made to an NFS mounted file system.

NFS-mounted
A file system connected by software to one system but physically residing on another system's disk.

NFS IO
A system count of the NFS calls.

NFS server
A node that provides data or services to other nodes on the network.

NFS transfer
Transfer of data packets across a local area network (LAN) to support Network File System (NFS) services.

Network Node Manager (NNM)
A network management application that provides the network map used by PerfView.

network time
The amount of time required for a particular network request to be completed.

nice
Altering the priority of a time-share process, using either the nice/renice command or the nice system call. High nice values lessen the priority; low nice values increase the priority.

node
A computing resource on a network, such as a networked computer system, hub, or bridge.

normal CPU
CPU time spent processing user applications which have not been real-time dispatched or niced.

operations
A DCE server's implementation of a procedure or function that provides remote services to clients.

outbound read/write
The designation used when a local process requests a read from or write to a remote system via NFS.

o/f (overflow)
This designates that the measurement software has detected a number that is too large to fit in the available space.

packet
A unit of information that is transferred between a server and a client over the LAN.

packet in/out

A request sent to the server by a client is an "in" packet. A request sent to a client by the server is an "out" packet.

page

A basic unit of memory. A process is accessed in pages (demand paging) during execution.

page fault

An event recorded when a process tries to execute code instructions or to reference a data page not resident in a process' mapped physical memory. The system must page-in the missing code or data to allow execution to continue.

page freed

When a paging daemon puts a page in the free list, it is considered as page freed.

page reclaim

Virtual address space is partitioned into segments, which are then partitioned into fixed size units called pages. There are usually two kinds of segments: persistent segments, and working segments.

Files containing data or executable programs are mapped into persistent segments. A persistent segment (text) has a permanent storage location on disk so the Virtual Memory Manager writes the page back to that location when the page has been modified and it is no longer kept in real memory. If the page has not changed, its frame is simply reclaimed.

page scan

The clock hand algorithm used to control page and to select pages for removal from system memory. It scans pages to select pages for possible removal.

page steal

Occurs when a page used by a process is taken away by the Virtual Memory Management system.

page in/page out

Moving pages of data from virtual memory (disk) to physical memory (page in) or vice versa (page out).

pagedaemon

A system daemon responsible for writing parts of a process' address space to secondary storage (disk) to support the paging capability of the virtual memory system.

pagein routine

A kernel routine that brings pages of a process' address space into physical memory.

pageout routine

A kernel routing that executes when physical memory space is scarce, and the pagedaemon is activated to remove the least-needed pages from memory by writing them to swap space or to the file system.

page request
A page fault that has to be satisfied by accessing virtual memory.

page space
The area of a disk or memory reserved for paging out portions of processes or swapping out entire processes. Also known as swap space.

parm file
The file containing the parameters used by MeasureWare Agent's scopeNT data collector to customize data collection. Also used to define your applications.

performance distribution range
An amount of time that you define with the range= keyword in the Transaction Tracker configuration file, ttd.conf.

perflbd.rc
The configuration file that contains entries for one or more data sources, each of which represents a scopeNT or DSI data source. See also repository server.

perfstat
The script used for viewing the status of all Hewlett-Packard performance products on your system.

PerfView
A tool that provides integrated performance management for multi-vendor distributed networks. Uses a single workstation to monitor environment performance on networks that range in size from tens to thousands of nodes.

pfaults
Most resolvable pfaults (protection faults) are caused by copy on writes (for example, writing to private memory segments). Most other pfaults are protection violations (for example, writing to a read-only region) and result in SIGBUS. See **mprotect(2)**.

physical IO
A input/output operation where data is transferred from memory to disk or vice versa. Physical IO includes file system IO, raw IO, system IO, and virtual memory IO.

physical memory
The actual hardware memory components contained within your computer system.

PID
A process identifier - a process' unique identification number that distinguishes it from all other processes on the system. PPID is a parent process identifier - the process identifier of a process that forked or vforked another process.

pipe
A mechanism that allows a stream of data to be passed between read and write processes.

priority
The number assigned to a PID that determines its importance to the CPU scheduler.

proc table
The process table that holds information for every process on the system.

process
The execution of a program file. This execution can represent an interactive user (processes running at normal, nice, or real-time priorities) or an operating system process.

process block
A process block occurs when a process is not executing because it is waiting for a resource or IO completion.

process deactivation
A technique used for memory management. Process deactivation marks pages of memory within a process as available for use by other more active processes. A process becomes a candidate for deactivation when physical memory becomes scarce or when a system starts thrashing.

Processes are reactivated when they become ready to run.

process state
Different types of tasks executed by a CPU on behalf of a process. For example: user, nice, system and interrupt.

pseudo terminal (pty)
A software device that operates in pairs. Output directed to one member of the pair is sent to the input of the other member. Input is sent to the upstream module.

queue
A waiting line in which unsatisfied requests are placed until a resource becomes available.

queue (DCE)
A line of client Remote Procedure Call requests waiting for service from a DCE server.

raw IO
Unbuffered input/output that transfers data directly between a disk device and the user program requesting the data. It bypasses the file system's buffer cache. Also known as character mode. Compare with block mode.

raw log file
A MeasureWare Agent file into which scopeNT logs collected data. It contains summarized measurements of system data. See logglob, logappl, logproc, logdev, logtran, and logindx.

read byte rate
The rate of kilobytes per second the system sent or received doing read operations.

read rate
The number of NFS and local read operations per second a system has processed. Read operations consist of getattr, lookup, readlink, readdir, null, root, statfs, and read.

Read/write Qlen
The number of pending NFS operations.

read/write system call
A request that a program uses to tell the kernel to perform a specific service on the program's behalf. When the user requests a read, a read system call is activated. When the user requests a write, a write system call is activated.

real time
The actual time in which an event takes place.

real-time cpu
Time the CPU spent executing processes that have a real-time priority.

rejects (DCE)
DCE client Remote Procedure Calls rejected by the DCE server because all threads were busy and the threadpool queue was full.

remote swapping
Swapping that uses swap space from a pool located on a different system's swap device. This type of swapping is often used by diskless systems that swap on a server machine.

repeat time
An action that can be selected for performance alarms. Repeat time designates the amount of time that must pass before an activated and continuing alarm condition triggers another alarm signal.

repository server
A server that provides data to the alarm generator and the PerfView analysis product. There is one repository for each data source configured in the perflbd.rc configuration file. A default repository server, provided at start up, contains a single data source consisting of a scopeNT log file set.

reserved swap space
Area set aside on your disk for virtual memory.

resident buffer
Data stored in physical memory.

resident memory
Information currently loaded into memory for the execution of a process.

resident set size
The amount of physical memory a process is using. It includes memory allocated for the process' data, stack, and text segments.

resize
Changing the overall size of a raw log file.

response time
The time spent to service all NFS operations.

response time (DCE)
The total time a DCE client Remote Procedure Call takes to execute on a DCE server. This time includes queuing time, marshalling/unmarshalling time and manager time.

roll back
Deleting one or more days worth of data from a raw log file with the oldest data deleted first. Roll backs are performed when a raw log file exceeds its maximum size parameter.

rxlog
The default extract log file created when data is extracted from raw log files.

SCOPE
The MeasureWare Agent default data source that contains a scopeux or scopeNT global log file set.

scopeux
The MeasureWare Agent data collector program that collects performance data and writes (logs) it to raw log files for later analysis or archiving.

scopeNT
The MeasureWare Agent data collector program that collects performance data and writes (logs) it to raw log files for later analysis or archiving.

scopeux log files
The raw log files that are created by the scopeux collector: logglob, logappl, logproc, logdev, logtran, and logindx.

scopeNT log files
The raw log files that are created by the scopeNT collector: logglob, logappl, logproc, logdev, logtran, and logindx.

semaphore
Special types of flags used for signaling between two cooperating processes. They are typically used to guard critical sections of code that modify shared data structures.

semaphore table
Maximum number of semaphores currently allowed for the system.

service level objective
A definable level of responsiveness for a transaction. For example, if you decide that all database updates must occur within 2 seconds, set the Service Level Objective (SLO) for that transaction as slo=2.

service level agreement
A document prepared for a business critical application that explicitly defines the service level objectives that IT (Information Technology) is expected to deliver to users. It specifies what the users can expect in terms of system response, quantities of work, and system availability.

shared memory
System memory allocated for sharing data among processes. It includes shared text, data and stack.

shared memory pool
The cache in which shared memory segments are stored.

shared memory segment
A portion of a system's memory dedicated to sharing data for several processes.

shared memory table
A list of entries that identifies shared memory segments currently allocated on your system.

shared text segment
Code shared between several processes.

signal
A software event to notify a process of a change. Similar to a hardware interrupt.

sleeping process
A process that either has blocked itself or that has been blocked, and is placed in a waiting state.

socket operation
A process that creates an endpoint for communication and returns a descriptor for use in all subsequent socket-related system calls.

start of collection
When you run a performance product, it starts collecting performance data.

summary data
The time period represented in one data point of a performance measurement. Summary levels can be five minutes, one hour, and one day.

swap
A memory management technique used to shuttle information between the main memory and a dedicated area on a disk (swap space). Swapping allows the system to run more processes than could otherwise fit into the main memory at a given time.

swap in/out
Moving information between the main memory and a dedicated (reserved) area on a disk. "Swapping in" is reading in to virtual memory; "swapping out" is reading out from virtual memory.

swap space
The area of a disk or memory reserved for swapping out entire processes or paging out portions of processes. Also known as page space.

system call

A command that a program uses to tell the kernel to perform a specific service on the program's behalf. This is the user's and application programmer's interface to the kernel.

system code

Kernel code that is executed through system calls.

system CPU

Time that the CPU was busy executing kernel code. Also called kernel mode.

system disk

Physical disk IO generated for file system management. These include inode access, super block access and cylinder group access.

system interrupt handling code

Kernel code that processes interrupts.

terminal transaction

A terminal transaction occurs whenever a read is completed to a terminal device or MPE message file. On a terminal device, a read is normally completed when the user presses the return or the enter key. Some devices such as serial printers may satisfy terminal reads by returning hardware status information. Several metrics are collected to characterize terminal transactions.

The FIRST_RESPONSE_TIME metric measures the time between the completion of the read and the completion of the first write back to that device. This metric is most often quoted in bench marks as it yields the quickest response time. For transactions which return a large amount of data to the terminal, such as reading an electronic mail message, the time to first response may be the best indicator of overall system responsiveness.

The RESPONSE_TIME_TO_PROMPT metric measures the time between the completion of the read and the posting of the next read. It is the amount of time that a user must wait before being able to enter the next transaction. This response time includes the amount of time it took to write data back to the terminal as a result of the transaction. The response time to prompt is the best metric for determining the limits of transaction throughput.

The THINK_TIME metric measures the time between posting a read and its completion. It is a measure of how much time the user took to examine the results of the transaction and then complete entering the next transaction. Transaction metrics are expressed as average times per transaction and as total times in seconds. Total times are calculated by multiplying the average time per transaction times the number of transactions completed.

Terminal transactions can be created by interactive or batch processes that do reads to terminal devices or message files. Reads to terminal devices or message files done by system processes will not be counted as transactions.

text segment

A memory segment that holds executable program code.

thrashing

A condition in which a system is spending too much time swapping data in and out, and too little time doing useful work. This is characteristic of situations in which either too many page faults are being created or too much swapping is occurring. Thrashing causes the system's performance to degrade and the response time for the interactive users to increase.

threadpool queue

A queue of requests waiting for an available server thread.

threshold

Numerical values that can be set to define alarm conditions. When a threshold is surpassed, an alarm is triggered.

trap

Software interrupt that requires service from a trap handler routine. An example would be a floating point exception on a system that does not have floating point hardware support. This requires the floating point operations to be emulated in the software trap handler code.

transaction

Some amount of work performed by a computer system on behalf of a user. The boundaries of this work are defined by the user.

Transaction Tracker

The technology used by MeasureWare Agent that lets information technology (IT) managers measure end-to-end response time of business application transactions.

trap handler code

Traps are measured when the kernel executes the code in the trap handler routine. For a list of trap types, refer to the file `/usr/include/machine/trap.h`.

ttd.conf

The Transaction Tracker configuration file where you define each transaction and the information to be tracked for each transaction, such as transaction name, performance distribution range, and service level objective.

unmount/mount

The process of removing or adding functionally-independent file systems from or to the root file system.

update interval

The interval of time between updates of the metrics that display in a report window or graph.

user CPU

Time that the CPU was busy executing user code. This includes time spent executing non-kernel code by daemon processes. It does not include CPU time spent executing system calls, context switching, or interrupt handling.

user disk

Physical disk IO generated by accessing the file system.

user code

Code that does not perform system calls.

utility program

A MeasureWare Agent program that lets you check parm file and alarmdef file syntax, resize log files, scan log files for information, and obtain alarm information from historical log file data.

vfault CPU

CPU time spent handling page faults.

vfaults

A vfault (virtual fault) is the mechanism that causes paging. Accessing an unmapped valid page causes a resolvable vfault. Accessing an illegal address results in a SIGSEGV.

vfork

A version of the fork system call that spawns a child process that is capable of sharing code and data with its parent process.

virtual memory

Secondary memory that exists on a portion of a disk or other storage device. It is used as an extension of the primary physical memory.

virtual memory IO

The virtual memory reads or writes from the disk for memory mapped files, and for paging out pages from paging area (swap area). Since all the files are memory mapped, all the reads or writes are virtual memory reads or writes as well. The computational memory of the processes that are changing is paged out if necessary to the swap area and read or written from there again.

write byte rate

The rate of kilobytes per second the system sent or received during write operations.

write rate

The number of NFS and local write operations the local machine has processed per second. Write operations include setattr, writecache, create, remove, rename, link, symlink, mkdir, rmdir, and write.

X-Axis

The horizontal scale on a graph.

Y-Axis

The vertical scale on a graph.

zero fill page

When pages are requested by the processes they are usually allocated by the Virtual Memory Management system and filled with zeros.