Number of Active user sessions TTY and PTTY

Load is a measure of the amount of computational work, averaged for 1, 5 and 15 minutes. In Linux it is the number of processes in R and D. IT reflects the time to wait for a task to be done.
In multi-core, the load average should be divided by the number of cores.

System Uptime

Current hour

```
top - 12:44:26 up 40 min,  0 users,  load average: 0.01, 0.03, 0.00
Tasks:   2 total,   1 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.5 sy,  0.0 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1938.4 total,    741.5 free,    393.1 used,    853.8 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1069.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0    4112   3456   2900 S   0.0   0.2   0:00.03 bash
   12 root      20   0    6088   3204   2672 R   0.0   0.2   0:00.05 top
```

Total number of processes

Processes perform a mix of I/O-bound work (such as reading disks) and CPU-bound work (such as performing arithmetic operations).
The CPU is idle when a process performs I/O, so OSes switch to executing other processes during this time.
In addition, the OS allows a given process to execute for a very small amount of time, and then it switches over to another process. This is how OSes appear as if they were “multitasking”. Doing all this requires us to keep track of the “state” of a process. In Linux, a process may be in of these states:

- Runnable (R): A process in this state is either executing on the CPU, or it is present on the run queue, ready to be executed.
- Interruptible sleep (S): Processes in this state are waiting for an event to complete.
- Uninterruptible sleep (D): In this case, a process is waiting for an I/O operation to complete.
- Stopped (T): These processes have been stopped by a job control signal (such as by pressing Ctrl+Z) or because they are being traced.
- Zombie (Z): The kernel maintains various data structures in memory to keep track of processes. A process may create a number of child processes, and they may exit while the parent is still around. However, these data structures must be kept around until the parent obtains the status of the child processes. Such terminated processes whose data structures are still around are called zombies.

Processes in the D and S states are shown in “sleeping”, and those in the T state are shown in “stopped”. The number of zombies are shown as the “zombie” value.

Linux uses a “nice” value to determine the priority of a process. A process with a high “nice” value is “nicer” to other processes, and gets a low priority. Similarly, processes with a lower “nice” gets higher priority. As we shall see later, the default “nice” value can be changed. The time spent on executing processes with a manually set “nice” appear

Percentage of CPU time spent executing processes in user space

Percentage of CPU time spent executing processes in kernelspace

This is followed by `id`, which is the time the CPU remains idle. Most operating systems put the CPU on a power saving mode when it is idle

Next comes the `wa` value, which is the time the CPU spends waiting for I/O to complete.

```
top - 12:44:26 up 40 min,  0 users,  load average: 0.01, 0.03, 0.00
Tasks:   2 total,   1 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.5 sy,  0.0 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  1988.4 total,   741.5 free,   393.1 used,   853.8 buff/cache
MiB Swap:  1024.0 total,  1024.0 free,     0.0 used.  1069.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0    4112   3456   2900 S   0.0   0.2   0:00.03 bash
   12 root      20   0    6088   3204   2672 R   0.0   0.2   0:00.05 top
```

Interrupts are signals to the processor about an event that requires immediate attention. Hardware interrupts are typically used by peripherals to tell the system about events, such as a keypress on a keyboard. On the other hand, software interrupts are generated due to specific instructions executed on the processor. In either case, the OS handles them, and the time spent on handling hardware and software interrupts are given by `hi` and `si` respectively.

In a virtualized environment, a part of the CPU resources are given to each virtual machine (VM). The OS detects when it has work to do, but it cannot perform them because the CPU is busy on some other VM. The amount of time lost in this way is the “steal” time, shown as `st`.

The “memory” section shows information regarding the memory usage of the system.

The lines marked “Mem” and “Swap” show information about RAM and swap space respectively. Simply put, a swap space is a part of the hard disk that is used like RAM.

When the RAM usage gets nearly full, infrequently used regions of the RAM are written into the swap space, ready to be retrieved later when needed.

However, because accessing disks are slow, relying too much on swapping can harm system performance.

.

```
top - 12:44:26 up 40 min,  0 users,  load average: 0.01, 0.03, 0.00
Tasks:   2 total,   1 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.5 sy,  0.0 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1988.4 total,    741.5 free,    393.1 used,    853.8 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1069.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0    4112   3456   2900 S   0.0   0.2   0:00.03 bash
   12 root      20   0    6088   3204   2672 R   0.0   0.2   0:00.05 top
```

As you would naturally expect, the “total”, “free” and “used” values have their usual meanings.

The “avail mem” value is the amount of memory that can be allocated to processes without causing more swapping.

The Linux kernel also tries to reduce disk access times in various ways.

It maintains a “disk cache” in RAM, where frequently used regions of the disk are stored.

In addition, disk writes are stored to a “disk buffer”, and the kernel eventually writes them out to the disk.

The total memory consumed by them is the “buff/cache” value. It might sound like a bad thing, but it really isn’t — memory used by the cache will be allocated to processes if needed.

```
top - 12:44:26 up 40 min,  0 users,  load average: 0.01, 0.03, 0.00
Tasks:   2 total,   1 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.5 sy,  0.0 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1988.4 total,    741.5 free,    393.1 used,    853.8 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1069.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0    4112   3456   2900 S   0.0   0.2   0:00.03 bash
   12 root      20   0    6088   3204   2672 R   0.0   0.2   0:00.05 top
```

- PID This is the process ID, a unique positive integer that identifies a process.

- USER This is the "effective" username (which maps to a user ID) of the user who started the process. Linux assigns a real user ID and an effective user ID to processes; the latter allows a process to act on behalf of another user. (For example, a non-root user can elevate to root in order to install a package.)

- PR and NI The "NI" field shows the "nice" value of a process. The "PR" field shows the scheduling priority of the process from the perspective of the kernel. The nice value affects the priority of a process.

- VIRT, RES, SHR and %MEM These three fields are related with to memory consumption of the processes.
  - "VIRT" is the total amount of memory consumed by a process. This includes the program's code, the data stored by the process in memory, as well as any regions of memory that have been swapped to the disk.
  - "RES" is the memory consumed by the process in RAM
  - %MEM" expresses this value as a percentage of the total RAM available.
  - "SHR" is the amount of memory shared with other processes.

- S As we have seen before, a process may be in various states. This field shows the process state in the single-letter form.

- TIME+ This is the total CPU time used by the process since it started, precise to the hundredths of a second.

- COMMAND The COMMAND column shows the name of the processes.