What to measure - overview.

Putting aside all the hype and comparisons that the commercial solutions providers write, to differentiate their offerings from the competition, there are remarkably few attributes that affect the performance of a server. In no particular order these are:

- network performance
- disk I-O
- memory usage
- CPU utilisation
- hardware architecture

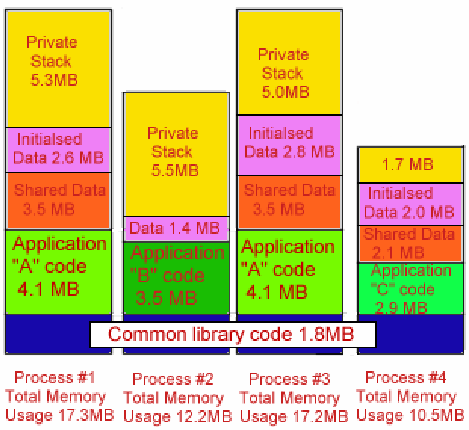
Network
The trickiest of these is network performance. Typically in any significantly sized organisation the complexity of the network will be greater than that of the servers. The reason being that apart from providing connectivity between one server and another, the network also connects users to the servers, printers, the outside world and any networked storage devices Therefore, it is often best to leave network operation and monitoring to the networking team. The technology, terminology, working practices and problems are quite different from those seen on servers and the experience needed to support computer networks make it a discipline in it's own right. For these reasons, I generally limit my network monitoring to the number and speed of packets in and out of a given network port. The pervasiveness of networks does mean that a network fault can look (to the users, at least) like a server problem and will often, if not always, be reported as "the system's slow/down again".

CPU
Traditionally CPU utilisation is seen as the main focus of performance monitoring. I think the reason is that until CPU speeds reached the gigaHertz levels, this was often the bottleneck. It is also one of the few attributes that can, on the face of it at least, be measured simply. This makes it an attractive target, especially on older systems. High CPU usage on it's own does not necessarily

mean you have a problem. For "batch" work (applications that don't have to wait for users input to execute, such as report generation) the server will consume as much processor resource as it can until it is limited either by it's own speed, or some other component of the server that limits it's ability to work any faster.

Memory

Memory usage is both easy to measure and hard to interpret. On the surface, most operating systems have metrics to tell you how much memory is allocated and how much is free. The simple approach is to watch these numbers change and if the amount of free memory drops to dangerously low levels AND the server appears to slow down, then it may be that more memory will help. The difficulty comes when you try to predict memory requirements to accommodate changes. Operating systems do not have a 1 to 1 mapping between demand and supply. Some pages can be shared among many copies of the same application, while other uses require unique allocations for each instance, as this diagram shows.



Here we have 4 processes. Process #1 and #3 are two instances of the same program, whereas #2 and #4 are unique. They all share code to perform common functions (opening files, writing data etc.) and some of them can share data that they only read (when a shared data page gets written to, it is immediately copied, so the reader doesn't get to see the modified version). The operating system will report the size of each process as shown in the diagram, for a total memory consumption of 57.2MB. However, this counts the library code 4 times when in fact all 4 processes access the same pages of memory. Likewise, the application "A" code used by processes #1 and #3 are double counted and the shared data segments (in orange) are also counted for each process. The actual memory requirements to run these 4 processes is 37.1MB. Each new instance of process #1 needs it's own stack and initialised data segments, but shares the library, application code and shared data that is already loaded.

Further, most systems will dynamically assign unused memory to dynamic system buffers, therefore showing very little unused memory. However system buffers are quickly deallocated when their memory needs to be assigned to running processes.

Disk

The only job your production servers have is to process data. That data is almost always stored on a disk or disk array at some point in it's life and most of the core business data is held in databases. The speed that the data can be read from and written to storage is critical in a servers ability to process it. While CPU speeds have shot up in the past decade and the amount of memory in a server has increased hugely, the speed of access to data on disks has remained pretty much static for the past 10 years (apart from advancements in solid-state disks). There have been some architectural improvements, such as smart arrays that achieve speed-ups by having massive memory caches but ultimately they still have to access mechanical devices to obtain data. An added complication for predictions is that as the rate of operations increase, the time taken to service them goes up, too.

Other factors

There are no particular performance metrics that show the effect of the architecture on a server. Some components of the architecture, such as system buses, have a marked effect on system performance but cannot be measured directly. Other aspects, such as the method used to connect disk arrays to the central processor or the bandwidth of a network connection will change the characteristics of the overall server and their impact can be measured indirectly. One major feature is the number of processors in a server and the performance increments that can be expected by adding more.

A note about disks

Whether you have a third party product, or use the system supplied tools to measure these attributes you should be careful about monitoring disks. It is quite normal to find many filesystems, or logical disks, on a single disk - although given the multiple layers of abstraction that are possible between these logical devices and the physical hardware, it is not always easy to spot which filesystems share the same hardware. The difficulty is fourfold:

- first of all, data rates to filesystems on common hardware all contribute to the overall performance of each other, so there are interdependencies that must be taken into account (see below).
- Secondly, some devices such as storage arrays will spread multiple filesystem across parts of many disks (disk striping) and name them as if they were independent. In fact you can get a situation where all your filesystems occupy stripes on all the disks in an array and therefore act as if they are all on one large spindle.
- The third point is again to do with disk arrays and is due to various levels of RAID that can be used. You may find that a physical disk containing your most frequently accessed data also contains a disk stripe that is another applications backup mirror and will therefore execute a large number of I-O operations at particular times. We will see the effect this can have in an example of a problem, later.
- Finally, look out for virtualisation. You may find virtual machines that are apparently independent but are actually sharing common hardware, such as disks and disk controllers.