**Content**

[1. Theory behind moniroting 2](#_Toc80363586)

[2. Tools 3](#_Toc80363587)

[3. Prometheus – Grafana – Alermanager – InfluxDB 4](#_Toc80363588)

1. Theory behind moniroting

**Pull vs push based moniroting solutions comparison**

**Pull**

* Monitoring solution does the work - The burden of gathering metrics is on the monitoring system itself
* Scrapes endpoints every x amount of seconds to add data
* Less „live“
* Can run from (almost) everywhere
* Access merics from web directory from endpoints
* Limited if using complicated netwroking and extensive firewalls
* Prometheus is a pull-based monitoring system

**Push**

* Clients do the work
* Event-based monitoring friendly
* More „live“
* Modular
* Less concert about firewalls

1. Metrics

For almost all monitoring solutions, including Prometheus, data for this metric is pulled from the **/proc/stat** file on the host itself

**node\_cpu\_seconds\_total** works as a counter — that is, it keeps track of how long the CPU spends in each mode, in seconds, and adds it to a persistent count. Counters might not seem especially helpful on their own, but combined with the power of math, we can actually get a lot of information out of it.

* idle - A computer processor is described as idle when it is not being used by any program The total of the user space percentage - us, the niced percentage - ni, and the idle percentage - id, should be close to 100%
* iowit – Time waiting for I/O
* irq - how much time the processor has spent servicing interrupts.
* nice - the priority level a user space process can be tweaked by adjusting its niceness. The ni stat shows how much time the CPU spent running user space processes that have been niced. On a system where no processes have been niced then the number will be 0.
* softirq – time fixing interrupts
* steal – if VM, the amount of time other VMS have stolen from you
* system - This is the amount of time that the CPU spent running the kernel. All the processes and system resources are handled by the Linux kernel. When a user space process needs something from the system, for example when it needs to allocate memory, perform some I/O, or it needs to create a child process, then the kernel is running. In fact the scheduler itself which determines which process runs next is part of the kernel. The amount of time spent in the kernel should be as low as possible.
* user - running user space processes. A user space program is any process that doesn't belong to the kernel. Shells, compilers, databases, web servers, and the programs associated with the desktop are all user space processes. If

1. Tools

* Prometheus
* Grafana
* InfluxDB
* OpsGenie
* New Relic
* Container monitoring
* Load testing ?!
* Nagios
* Service discovery (cunsul, zookeeper, nerve .. )
* Collectd – the node exporter for push monitoring systems
* Telegraf !

1. Prometheus – Grafana – Alermanager – InfluxDB

