

CMPS 261 Server Management - Module 4: Tuning and Configuration

Mark Voortman, Ph.D.

Course Modules

- [Module 0: Course Design](#)
- [Module 1: Introduction to Servers and Server Operating Systems](#)
- [Module 2: Getting Started with FreeBSD Server](#)
- [Module 3: Software Maintenance](#)
- [**Module 4: Tuning and Configuration**](#)
- [Module 5: Storage Management](#)
- [Module 6: Networking](#)
- [Module 7: Shell Scripting](#)
- [Module 8: Building a WordPress Server](#)

Module 4

- **Tuning and Configuration**
- Part A
 - Perform configuration and tuning analysis and adjustments
 - Computer perspective
 - Enterprise perspective
 - Immediate problem solving
 - Long term monitoring
 - Start and stop services
 - Set up scheduled tasks
- Part B
 - Understand system logging and how to use logging for investigation
 - Understand the boot process

Objectives

- Upon successful completion of this module, you should be able to:
 - Understand the need for configuration and tuning
 - Understand what services are and how to control them
 - Schedule tasks
 - Use the system logging facilities
 - Understand the boot process

PART A

Server Performance

- Need to assure that servers are performing adequately for required tasks
 - Is a server under stress from too much work?
 - What is nature of stress? Memory? CPU? Disk I/O?
 - What is the cause of the stress?
 - What will solve the problem?
 - Long term, what is capacity of server to handle growth?

Investigation Versus Monitoring

- Investigation
 - When a problem is observed, and we must find out the cause
 - Also termed troubleshooting
- Monitoring
 - Proactive observations of key performance metrics
- Tools we use for each are different

Investigation/Troubleshooting

- No one single defined script to follow
- What you do is dictated by circumstances
- Alerted to issue in various ways
 - Internal users report speed or access problems
 - Can't print, save files, or check email
 - Failures in applications
 - External users report access problems
 - Can't get access to web site

Investigation/Troubleshooting

- Problem could be network
- Problem could be in storage system
- Problem could be on different server
 - Database server issue could be exposed via an application server
- Problem could be database related
 - Performance problems caused by corrupt database indexes

Investigation/Troubleshooting Commands

- Our review will be restricted to those commands available in FreeBSD to check on performance.
- We will first review the commands, then conduct a demonstration.
- We recommend that you open the manual pages web site as we discuss each command.
 - <https://www.freebsd.org/cgi/man.cgi>

Investigation/Troubleshooting Commands

- [vmstat](#)
 - Report virtual memory statistics
 - Other information also displayed
 - Processes
 - Memory
 - Page
 - Disks
 - Faults
 - CPU

Investigation/Troubleshooting Commands

- [uptime](#)
 - Shows how long system has been running
 - Reports number of users logged in
 - System load value at 1, 5 and 15 minute periods
 - Remember two shell commands that are useful here:
 - who
 - w

Investigation/Troubleshooting Commands

- who -b
 - Reports when system was last booted

Investigation/Troubleshooting Commands

- [top](#)
 - Display and update information on top (CPU consuming) processes

Investigation/Troubleshooting Commands

- [ps](#)
 - Display information about active processes
 - By default, for the user running command

Investigation/Troubleshooting Commands

- [swapinfo](#)
 - Gives condition of swap space

Investigation/Troubleshooting Commands

- [iostat](#)
 - Report I/O statistics

Investigation/Troubleshooting Commands

- [systat](#)
 - Display system statistics of various kinds

Investigating/Troubleshooting Summary

- We have seen commands that let us look at conditions on a single computer
 - Current conditions, not history
- Many more utilities and scripts can be found to provide additional information
- Now let's turn our attention to longer term monitoring

Investigation Versus Monitoring

- Investigation
 - Looking at conditions on one server at a point in time
- Monitoring
 - Longer term observation, across the enterprise
 - Tools we use for each are different

Roll Your Own?

- Unix has a wealth of commands and utilities
- People can, and do, cobble together monitoring capabilities
- Example: Using the nc command

The nc Command

- [nc](#) is a Swiss Army knife TCP/UDP utility
- Could be used to send monitoring data from one server to a central repository
- Data could be processed from there
- Try it out
 - Set up the client/server model described in the man page

Reality Check

- While Unix/Linux are popular for servers, they are not exclusive
 - Most organizations have numerous servers, of many flavors
 - All should be monitored

The Monitoring Tool Solution

- Server monitoring is a well-served software category
 - Commercial
 - Open source
 - Web-based
- Constant monitoring, with alerts generated if problems occur
- Rich graphical interfaces to portray information
 - “A picture is worth a thousand words”

SolarWinds (Commercial)

Average Response Time & Packet Loss



Avg Resp Time



Packet Loss

Average CPU Load & Memory Utilization



Avg CPU Load



Memory Used

Node Details

| | |
|--------------------|--|
| Node Status | Node is Up |
| Polling IP Address | 10.199.14.30 |
| Dynamic IP | No |
| Machine Type | Windows 2012 R2 Server |
| DNS | lab-dmz-its-01.demo.lab |
| System Name | lab-dmz-its-01 |
| Description | Hardware: Intel(R) Family 6 Model 26 Stepping 5 ATX IT COMPATIBLE - Software: Windows Version 6.3 (Build 9600 Multiprocessor Free) |
| Location | Authn |
| Contact | |
| SystemObjectID | |
| Last Boot | Tuesday, December 15, 2015 9:56 AM |
| Software Version | 6.3 (Build 9600 Multiprocessor Free) |
| Software Image | Unknown |
| Hardware | Virtual hosted by lab-dmz-esp.demo.lab |
| No of CPUs | 2 |

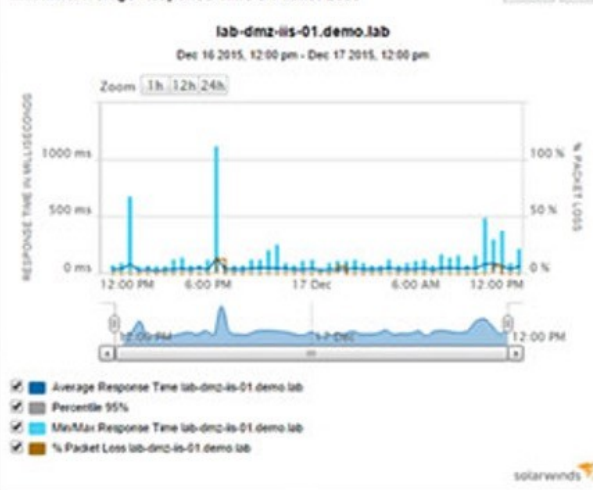
Interface Downtime

| STATUS | NAME |
|--------|---|
| Up | Intel(R) 82574L Gigabit Network Connection - Ethernet |

12/16/2015 23:58 12/17/2015 23:58

Unknown Up Down Warning Shutdown Unmanaged Unplugged Unreachable

Min/Max/Average Response Time & Packet Loss



Top 10 Monitored Processes by Physical Memory

| PROCESS NAME | APPLICATION NAME | NETWORK NODE | PHYSICAL MEMORY USED |
|--------------|------------------|-------------------------|----------------------|
| w3wp.exe | Microsoft IIS | lab-dmz-its-01.demo.lab | 385 MB 18.81% |
| w3wp.exe | Microsoft IIS | lab-dmz-its-01.demo.lab | 216 MB 10.56% |
| w3wp.exe | Microsoft IIS | lab-dmz-its-01.demo.lab | 157 MB 7.68% |
| w3wp.exe | Microsoft IIS | lab-dmz-its-01.demo.lab | 49 MB 2.39% |
| smss.exe | Microsoft IIS | lab-dmz-its-01.demo.lab | 0.1 MB 0.01% |

Management

| | |
|----------------------------|-------------------------|
| Real-Time Process Explorer | Service Control Manager |
| Real-Time Event Log Viewer | Reboot |

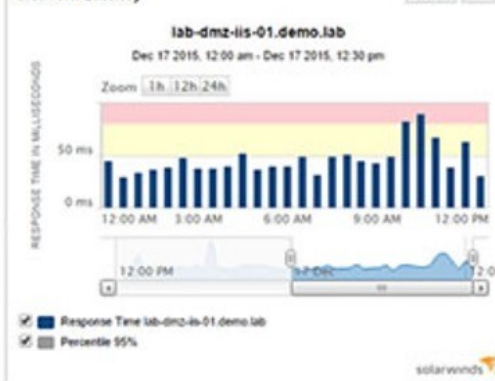
Disk Volumes

| VOLUME | SIZE | SPACE USED |
|-------------------|---------|-------------|
| C:\Label 262624F3 | 39.7 GB | 26.1 GB 66% |
| Physical Memory | 2.0 GB | 1.4 GB 70% |
| Virtual Memory | 2.4 GB | 1.7 GB 71% |

Node Capacity Forecast

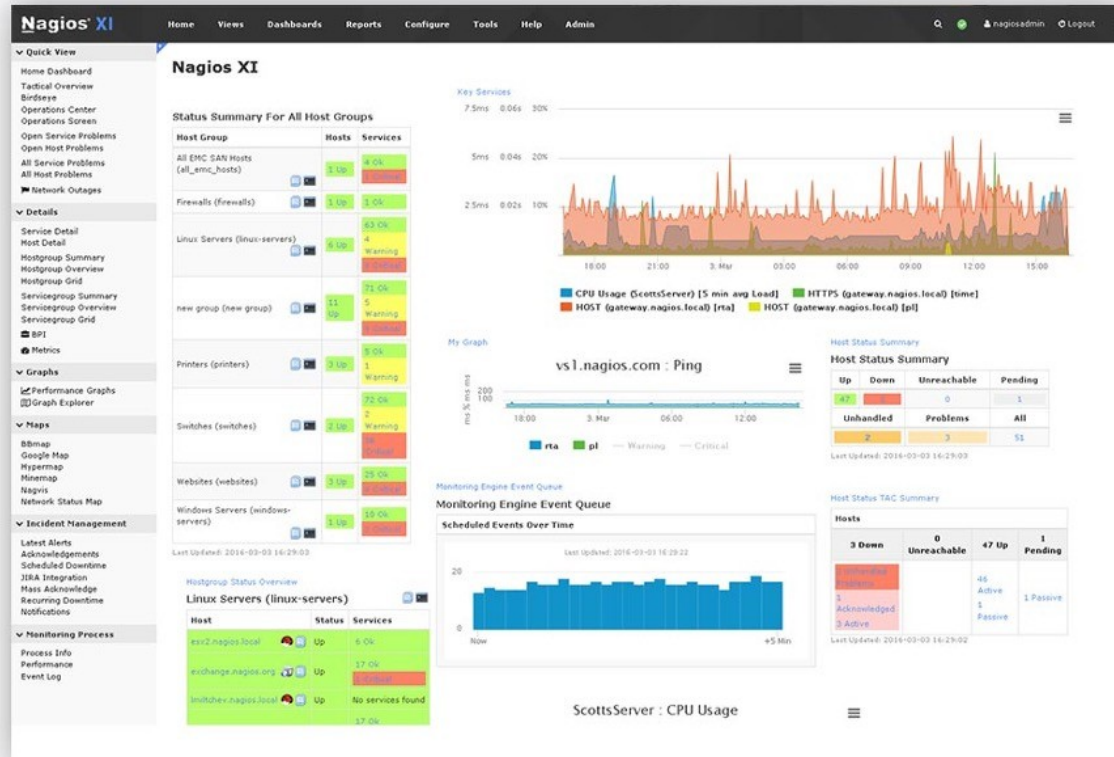
| RESOURCE | LAST 7 DAYS | WARNING | CRITICAL | AT CAPACITY |
|----------------------|---------------|-----------------|-----------------|------------------|
| CPU Load | Avg 7% 1% | >65% >1 year | >90% >1 year | >100% >1 year |
| Percent Memory Usage | Avg 60% 5% | >80% 2 weeks | >90% 3 weeks | >100% 4 weeks |

Network Latency



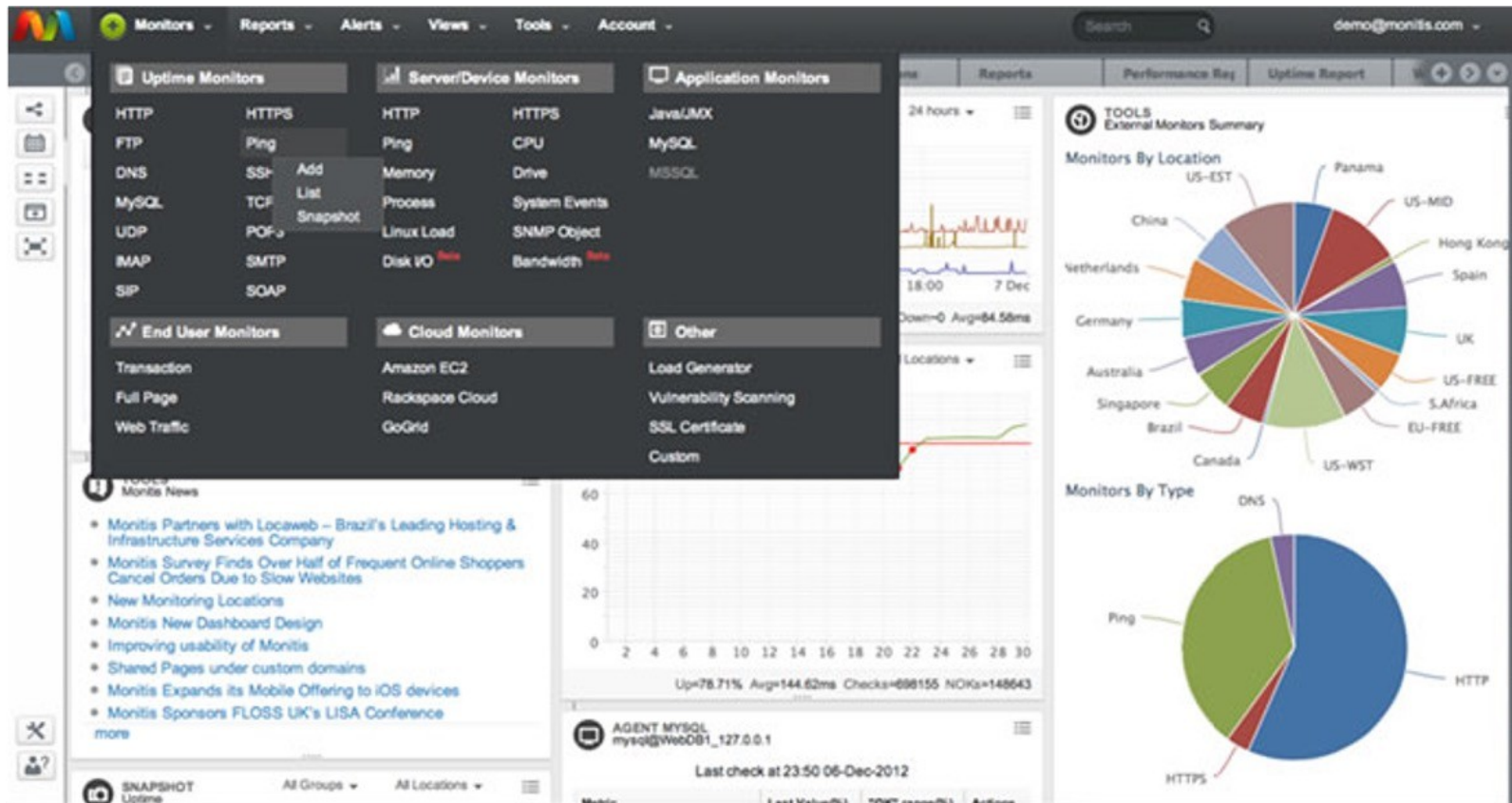
http://www.solarwinds.com/server-application-monitor?CMP=OTC-tad-dns-dns_serm-on-sam-PP

Nagios (Commercial)



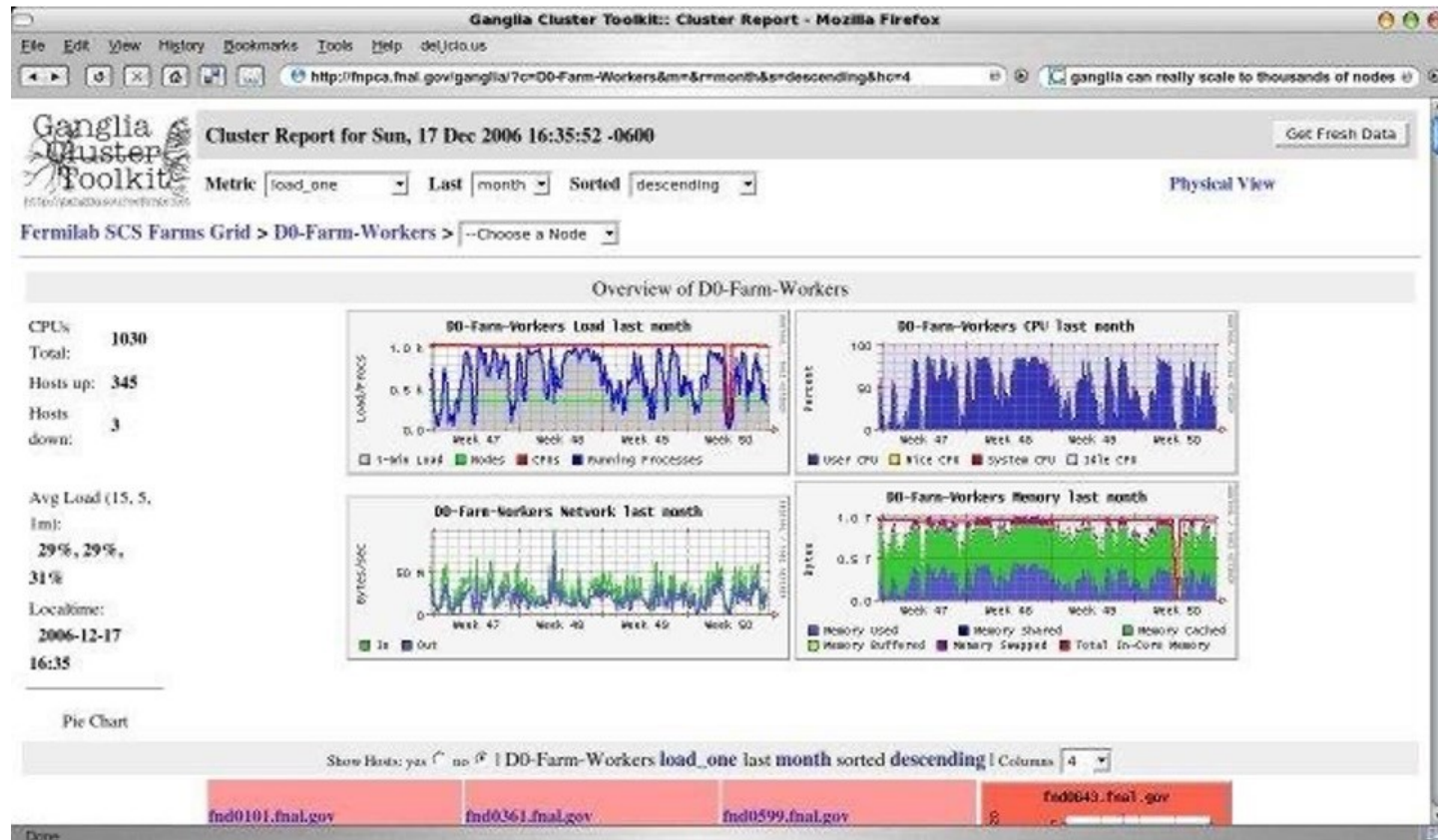
<https://www.nagios.com/products/nagios-xi/>

Monitor.US (Free/Commercial)



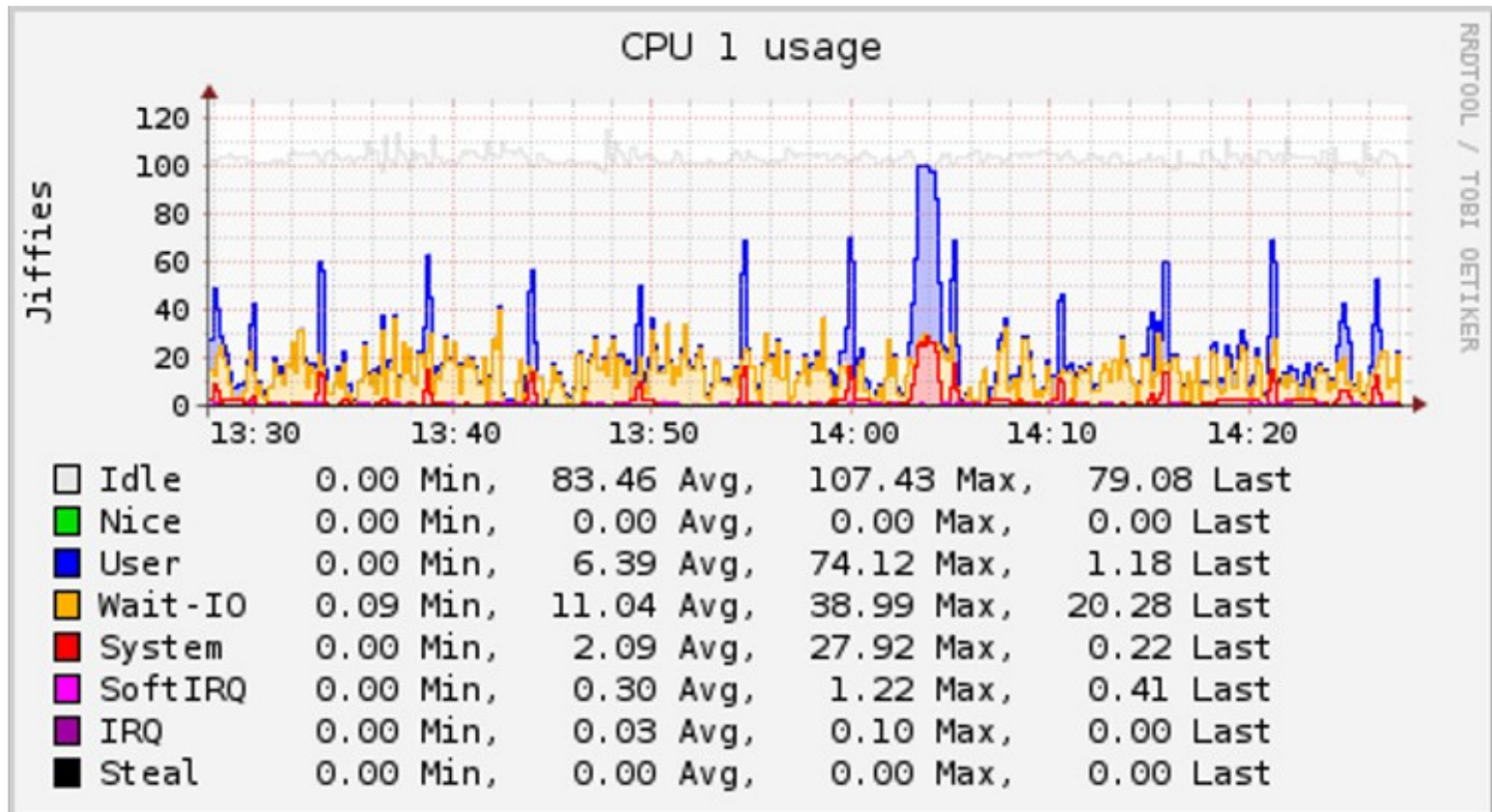
<http://www.monitis.com/server-monitoring>

Ganglia (Open Source)



<http://ganglia.info/>

CollectD (Open Source)



<http://collectd.org/>

Glances (Open Source)

```
xps (Ubuntu 14.04 64bit / Linux 3.13.0-85-generic) - IP 192.168.0.6/24 Uptime: 1 day, 20:23:55

1.80/1.80GHz CPU [|||||] 100% user: 97.9% nice: 0.0% ctx_sw: 7605 MEM 26.8% active: 878M SWAP 7.6% LOAD 4-core
MEM [|||||] 26.8% system: 2.1% irq: 0.0% inter: 5015 total: 7.71G inactive: 1.50G total: 7.91G 1 min: 3.21
SWAP [||] 7.6% idle: 0.0% iowait: 0.0% sw_int: 1273 used: 2.06G buffers: 5.73M used: 612M 5 min: 1.86
free: 5.64G cached: 645M free: 7.31G 15 min: 1.17

NETWORK Rx/s Tx/s CONTAINERS 2 (served by Docker 1.11.1)
docker0 0b 0b
lo 392b 392b
h2c39a99 0b 0b
h610b701 0b 0b
wlan0 10.5Mb 860Kb

Name Status CPU% MEM /MAX IOR/s IOW/s Rx/s Tx/s Command
dbgrafana_grafana_1 Up 2 mins 0.0 5.94M 7.71G 0b 0b 0b 0b /run.sh
_bgrafana_influxdb_1 Up 2 mins 0.1 8.60M 7.71G 0b 0b 0b 0b /run.sh

TASKS 257 (780 thr), 5 run, 252 slp, 0 oth sorted automatically by cpu percent, flat view

CPU% MEM% VIRT RES PID USER NI S TIME+ R/s W/s Command
96.0 0.0 7.13M 100K 20888 nicolargo 0 R 0:03.50 0 0 stress --cpu 4 -t 30
91.8 0.0 7.13M 100K 20890 nicolargo 0 R 0:03.33 0 0 stress --cpu 4 -t 30
91.5 0.0 7.13M 100K 20891 nicolargo 0 R 0:03.26 0 0 stress --cpu 4 -t 30
86.4 0.0 7.13M 100K 20889 nicolargo 0 R 0:03.19 0 0 stress --cpu 4 -t 30
12.7 11.8 2.56G 933M 11378 nicolargo 0 S 2h21:43 0 1M /usr/lib/firefox/firefox
5.1 0.3 548M 23.4M 19899 nicolargo 0 R 0:07.27 0 0 python -m glances
4.3 2.1 2.04G 163M 3278 nicolargo 0 S 36:17.83 0 0 /usr/bin/gnome-shell
2.7 0.0 0 0 577 root 0 S 1:36.94 0 0 irq/59-lwlwifi
1.5 1.3 477M 100M 2141 root 0 S 17:18.96 0 0 /usr/bin/X :0 -background none -verbose -auth /var/run/gdm/aut
1.5 1.5 1.18G 122M 23657 nicolargo 0 S 0:07.93 0 0 /usr/bin/perl /usr/bin/shutter
0.6 0.2 914M 19.2M 19237 nicolargo 0 S 0:33.56 0 0 /usr/bin/python /usr/bin/terminator
0.6 0.3 606M 20.8M 2870 root 0 S 0:30.96 0 0 /usr/bin/docker daemon --raw-logs
0.6 0.1 358M 6.70M 3142 nicolargo 0 S 1:14.27 20K 0 /usr/bin/ibus-daemon --daemonize --xim
0.3 0.1 612M 11.1M 3381 nicolargo 19 S 0:44.80 0 0 /usr/lib/tracker/tracker-miner-fs
0.3 0.0 201M 1.16M 3227 nicolargo 0 S 0:19.28 2K 0 /usr/lib/ibus/ibus-engine-simple
0.3 0.3 1.18G 22.2M 4835 nicolargo 0 S 0:23.10 0 0 nautilus --new-window
0.3 0.0 410M 2.81M 1112 root 0 S 0:07.11 0 0 NetworkManager
0.3 0.3 649M 27.3M 3099 nicolargo 0 S 0:08.46 0 0 /usr/lib/x86_64-linux-gnu/bamf/bamfdaemon
0.3 0.0 0 0 79 root 0 S 0:20.63 0 0 kworker/3:1
0.3 0.0 88.4M 240K 2083 www-data 0 S 0:02.72 0 0 nginx: worker process
0.3 0.3 2.07G 25.6M 2194 rabbitmq 0 S 5:41.18 0 0 /usr/lib/erlang/erts-5.10.4/bin/beam.smp -W w -K true -A30 -P
0.3 0.0 0 0 16492 root 0 S 0:01.94 0 0 kworker/2:2
0.0 0.0 0 0 18 root 0 S 0:00.00 0 0 rcuob/1
0.0 0.1 999M 10.0M 3315 nicolargo 0 S 0:09.78 0 0 /usr/lib/gnome-online-accounts/goa-daemon
0.0 0.0 0 0 27916 root 0 S 0:00.00 0 0 irq/61-mei_me
0.0 0.0 0 0 39 root 0 S 0:00.70 0 0 ksoftirqd/3

Warning or critical alerts (last 4 entries)
2016-05-16 16:53:12 (ongoing) - CPU_USER (97.6): stress, stress, stress
2016-05-16 16:52:41 (0:00:18) - WARNING on MEM (76.0)
2016-05-16 16:52:01 (0:00:33) - CRITICAL on CPU_IOWAIT (Min:51.9 Mean:63.4 Max:77.1): bash, stress, firefox
2016-05-16 16:51:31 (0:00:30) - CRITICAL on CPU_USER (Min:80.5 Mean:95.8 Max:98.3): stress, stress, stress
```

<http://glances.readthedocs.io/en/latest/index.html>

Glances

- Install steps:
 - `pkg install py38-glances`
 - `pkg install py38-bottle`
- To run Glances:
 - `glances`
- To run and allow remote web access:
 - `glances -w -B {IP address of server}`
 - `glances -w -B 192.168.1.74`
 - Observe assigned port number
 - Use port number when browsing to site

Summary

- A rich set of software solutions are available for enterprise server monitoring
- Value of the comprehensive information gathered makes this a compelling option

Services

- Services are background applications that run without a user interface
- Provide vital functionality
 - Internal to the server operations
 - External access to server functionality
- Examples
 - Running apache web server
 - Accessing your system remotely via secure shell

Services

- A service is a specially designed application (or script) that silently performs its job
- Example:
 - Apache monitors TCP/IP traffic coming in to ports 80 and 443 (typically)
 - Apache responds to requests by sending out web pages, graphics, etc.
 - Services tend to silently monitor, waiting for requests
 - Services may need to start at boot time

Managing Services

- System maintenance and trouble resolution may require interacting with services
- Services can crash or run amok
- [service](#) command available to interact with services

List Enabled Services

- `service -e` will display enabled services
- Try it out
 - `service -e`

Interacting With Services

- service command has four main options
 - start – to start the service
 - stop – to stop the service
 - restart – to stop and restart
 - status – to display status
- Try it out
 - `service devd status`

Interacting With Services

- Enabled service list does not show running services
- Try it out
 - `service sshd stop` (be careful not to lock yourself out)
 - `service sshd status`
 - `service -e`

Monitoring Services

- monit is a tool that can be used to monitor services
 - Other aspects of a system also
- `pkg install monit`
 - Will require configuration before running
 - See <https://mmonit.com/monit/>

Monit

```
root@CMPS261Server:/usr/home/admin1 # monit summary  
Monit 5.20.0 uptime: 1h 42m
```

| Service Name | Status | Type |
|---------------|---------|---------|
| CMPS261Server | Running | System |
| sshd | Running | Process |
| cron | Running | Process |
| ntpd | Running | Process |
| public | UP | Network |

monit can be run from command line

Monit

Monit Service Manager

Monit is running on CMPS261Server and monitoring:

| System | Status | Load | CPU | Memory | Swap |
|----------------------|---------|----------------------|----------------|----------------|------------|
| <u>CMPS261Server</u> | Running | [0.82] [0.70] [0.54] | 0.1%us, 0.0%sy | 3.4% [67.7 MB] | 0.0% [0 B] |
| Process | Status | Uptime | CPU Total | Memory Total | |
| <u>sshd</u> | Running | 39m | 0.0% | 0.4% [7.6 MB] | |
| <u>cron</u> | Running | 2h 9m | 0.0% | 0.1% [2.5 MB] | |
| <u>ntpd</u> | Running | 2h 9m | 0.0% | 0.5% [10.2 MB] | |
| Net | Status | | Upload | Download | |
| <u>public</u> | UP | | 72 B/s | 93 B/s | |

monit can also be run remotely via browser

Monit

Process status

| Parameter | Value | |
|---|--------------------|---|
| Name | sshd | |
| Pid file | /var/run/sshd.pid | |
| Status | Running | |
| Monitoring status | Monitored | |
| Monitoring interval | Port response time | 6.936 ms to localhost:22 type TCP/IP protocol SSH |
| On reboot | Data collected | Sun, 05 Feb 2017 12:22:33 |
| | Existence | If doesn't exist then restart |
| | Port | If failed [localhost]:22 type TCP/IP protocol SSH with timeout 5 s then restart |
| <div>Start service</div> <div>Stop service</div> <div>Restart service</div> <div>Disable monitoring</div> | | |

Services can be remotely managed

Summary

- Service applications provide critical functionality in the server environment
- Administrators will need to know how to interact with services
- Will also need to know how to set up new services
 - Discussed as we move forward

cron

- cron daemon is standard tool for running scheduled tasks
 - Starts when system boots
 - Runs while system is up
- cron reads configuration files
 - Contains lists of commands to run
 - When to run them
 - Commands are executed by sh
 - Virtually anything you can do by hand, can be done on schedule

cron service

- Check status of cron service:
 - service cron status
 - This service can only be checked by root
 - The `crond.pid` file is protected, which is why

cron Configuration Files

- cron controlled by one or two configuration files
 - System configuration file is `/etc/crontab`
 - User configuration file
 - Each user can have one, or no, configuration file
 - Location of user config files is `/var/cron/tabs`
 - Supplements system cron

cron Configuration File Contents

- Each line has run time parameters
 - * matches everything
 - Single integer matches exactly
 - Two integers separated by dash is range
 - Comma separated list of integers or ranges
 - */integer is that dimension, every interval

cron Configuration File Contents

```
# /etc/crontab - root's crontab for FreeBSD
#
# $FreeBSD$
#
SHELL=/bin/sh
PATH=/etc:/bin:/sbin:/usr/bin:/usr/sbin
#
#minute hour    mday    month    wday    who      command
#
*/5      *        *        *        *        root     /usr/libexec/atrun
```

Cron overview

cron Configuration File Contents

```
#minute hour    mday    month    wday    who  command
20      1         *        *        *       root find /tmp -atime +3 -type f -exec rm -f {} \;
```

Example task

User crons

- `crontab` command used to manage user crons
- Can be executed by root, or by a user (for their cron)
- `crontab -e`: Creates or edits a user cron
- `crontab -l`: Lists contents of a user cron
- `crontab -r`: Removes a user cron
- If running from root, always use `-u` to specify user

Controlling cron Access

- Two files control who can edit their crons:
 - `/var/cron/allow`: List of users allowed to use crontab
 - `/var/cron/deny`: List of users prohibited from using crontab
 - If any name appears in `allow`, then all allowed names must be included, even `root`
 - Entries are simply user names, one per line
- Where crons are stored:
 - `/var/cron/tabs`

cron Example

- Let's add a cron to write the date and time to a file
- Look at contents of system cron

Summary

- Scheduling operations are a powerful way to have a system do the work for you
- Virtually any command can be executed
- Each user can have their own cron
 - Supplements, not replaces, main system cron

PART B

Syslog and Log Files

- Application and service log files are a prime source of key information about system performance
- Log files are text files
- Log management tools automatically rotate, compress, and remove log files
- Logging can be per machine, or centralized

Log File Locations

- Log files can be anywhere on the system
 - Decision made by application author
- Often, the location will be /var/log
- syslogd daemon is the standard logging facility
 - Applications can do their own logging thing
 - Look at /etc/syslog.conf for logging configuration

Syslog Configuration

- `/etc/syslog.conf` contains logging configuration
- Can be a guide to which log file pertains to a feature
- `security.*` `/var/log/security`
- `auth.info;authpriv.info` `/var/log/auth.log`
- `mail.info` `/var/log/maillog`
- `lpr.info` `/var/log/lpd-errs`
- `ftp.info` `/var/log/xferlog`
- `cron.*` `/var/log/cron`

Log File Location

- Finding log files made more difficult by non-standard naming
 - Example names (in /var/log):
 - auth.log
 - bsdinstall_log
 - cron
 - dmesg.today
 - lpd-errs
 - maillog
 - sendmail.st
 - xferlog

Log File Rotation

- Log files are often appended to
- This can result in log files that grow excessively large and contain obsolete information
- A cron process runs periodically that rotates log files and deletes ones no longer needed
- See `/etc/newsyslog.conf` for list
- Logs can be rotated based on size, point in time, or both

Centralized Logging

- FreeBSD allows writing logging information to a central server
- A FreeBSD server would be set up to perform this duty
- It's `syslog.conf` would contain the lists of clients sending it logging info
- Each sending server would then be configured to point to the logging server

Varying Levels of Access

- Some logs readable by all
- All logs readable by super user

Working With Log Files

- Since log files are (mostly) text files, you can handle them as such
 - head
 - tail
 - grep
 - cat
- Log files can get large
- Example:
 - `grep error /var/log/messages`

Working With Log Files

- Tool named `colorize` will display logs in color
 - `pkg install colorize`

Colorize Example Output

```
$ grep error /var/log/messages | colorize
Feb  5 10:20:23 CMPS261Server kernel: module_register_init: MOD_LOAD (vesa, 0xffffffff8101c970, 0) error 19
Feb  5 10:20:23 CMPS261Server sshd[609]: error: Bind to port 22 on :: failed: Address already in use.
Feb  5 10:20:23 CMPS261Server sshd[609]: error: Bind to port 22 on 0.0.0.0 failed: Address already in use.
Feb  6 18:29:51 CMPS261Server kernel: module_register_init: MOD_LOAD (vesa, 0xffffffff8101c970, 0) error 19
Feb  6 18:29:53 CMPS261Server sshd[609]: error: Bind to port 22 on :: failed: Address already in use.
Feb  6 18:29:53 CMPS261Server sshd[609]: error: Bind to port 22 on 0.0.0.0 failed: Address already in use.
```


Writing to the System Log

- `logger` command writes to messages log
- Useful in scripts
- `$ logger Check this out!!`
- `$ tail /var/log/messages`
- Last line should look like this:
 - `Feb 7 00:29:51 CMPS261Server admin1: Check this out!!`
- Try this yourself!

Summary

- Logs are one of the main diagnostic tools for the system administrator

System Start-Up

- Reference materials
 - [FreeBSD handbook](#)
 - [init man page](#)

System Start-Up

- The starting of a computer system is a well-orchestrated set of tasks
- Important to understand how it works
 - May be a need to intervene to fix problems
 - System might not boot up normally

The Boot Process

- The power on process executes code in the BIOS to find the startup code on the disk
- The master boot record or GUID partition table
- This executes boot0, which only knows to execute boot1
- boot1 knows only to find boot2 and execute it
- boot2 executes the loader process
 - Starts with `/boot/loader.rc`
 - Gets information from `/boot/defaults/loader.conf` and `/boot/loader.conf`
- This essentially loads the kernel and begins running it

System Initialization

- The kernel continues start-up by running the init process
- init process critical to successful operation
 - If init dies, system will reboot
- init runs `/etc/rc` which contains start-up commands
- `/etc/rc.conf` is a configuration file with some editable settings
 - DHCP versus fixed IP
 - Start up of services
 - Host name of computer

Intervening in the Start-Up

- There are numerous locations where edits can be made to control start-up
- Must tread very carefully here
- Unix is resilient, and will at least partially boot
- Can utilize recovery mode
 - Boot into single user mode
 - Stop at boot step to manually enter commands

Looking at Boot-Up Messages

- Use dmesg command to display messages
 - Message will indicate status of interactions with hardware devices

Shutting Down

- Shutdown is as ordered as start-up
- `/etc/rc.shutdown` is run
- TERM signals sent to all processes
 - If they don't shut down, then KILL signal is sent
- To immediately power off:
 - `shutdown -p now`
 - `poweroff`

Practice Project

- Install and configure [Nextcloud](#) in your jail
- [Follow the Nextcloud tutorial](#)
- Will need to apply all knowledge learned so far

THANK YOU!