HOWTO: Collect an AIX performance baseline

Contributed by Michael Felt

Use nmon as basis for performance analysis
Too often I visit
customers and "discover" that they have one of many condiftions that make analysis of performance very difficult. Generally there is either too little, too much, or (too) different data collected to do a proper analysis. At the end of this article I provide a script as an example of how performance data collection can be standardized. In between I discuss the reasoning for the number of collections, as well as number and duration of the samples.

Although many customers have adopted the use
of nmon for collecting performance statistics their
collection process is rarely standardized. When it is
standardized the intended detail of data is often lost in the reports because the defaults taken or chosen are not sized right.

History and Purpose of nmon tools

nmon (Nigel's monitor is the creation of Nigel Griffiths) is well known for it's interactive mode. nmon began as a AIX version of the command top. Nigel's goal was to integrate views of common performance statistics that otherwise required several programs such as sar, vmstat, iostat, netstat, nfsstat, entstat, etc.. That was not enough - he also wanted a way to be able to present the statistics in graphical form using a spreadsheet. With the assistance of a colleague - Stephen Atkins - nmon and nmon_analyzer were born. In short - nmon is the collector, and nmon_analyzer and
nmon_consolidator are the reporters.

nmon_analyzer is best at "bare metal" analysis and
has few limitations. Perhaps the only limitation is than an .nmon file can get too big to be
analyzed easily. Generally speaking, for nmon_analyzer any number of samples can be collected - nmon_analyzer takes it in stride. This is not the case for the "consolidators".

The "consolidators" (consolidator and trend) provide two ways to view data, The one to use is determined by the number of different hosts and starting time of the data. When the collections all start on the same day, at the same time, with the same interval and number of samples the statistics are literally consolidated, or stacked on top of each other and a performance consultant (forexample - you!) can view potential dependancies and or bottlenecks coming from the collective or group of machines rather than from a single machine. When the data is only from one host - but starting on different days at the same time with the same interval interval and sample count the graphs are not stacked but drawn in parallel showing differences and similiarities of the data collected. Small, but consistent change is seen in this "trending" view. Also, seeing when and how a sudden change from a long standing pattern appears may help determine the root cause of a sudden, unexpected, performance issue.

In short, there is a difference between the purpose and preferred data
collections for the analyzer and consolidator tools. I see the
nmon_analyzer as a "bare metal" analysis for a single system or virtual
machine while the nmon_consolidated views are for comparing multiple, related data collections.

NMON is a collector
... not a reporter

Remember that nmon is the collector and/or presentor of raw data. The performance analysis and reporting is done using the nmon_analyzer and nmon_consolidator tools. nmon_analyzer focus on presenting "bare metal" reports and consolidator tools analyse and report on multiple, related samples.

The design of the consolidators specifies that a maximum of 250 samples are compared. This may have initially been a spreadsheet limitation. Why is not relevant now. What is important is that it is a very real limitation and the consolidators compress data into fewer samples when this number is exceeded. The effect is that intended or expected detail is lost. When there are more than 250 samples the first pass of the consolidators is to compress (average) the data into fewer data points. As an example: a (common default configuration) collection of 24 hours with a sample of every 5 minutes (300 seconds) has 288 samples. nmon_analyzer - the bare metal view reports the data as is. The consolidators first reduce, compress, or average the data down to 144 data points (288 / 2) to get within the 250 sample limit. The number of points drawn and the implicit detail is lost. In other words - the details are no longer 5 minute averages, but 10 minute averages. Nothing wrong with 10 minute averages of course - for historical views - but collecting 5 minute samples is for 10 minute reporing is collecting twice as much raw data for very course averages. And if you are keeping data for the life of the application - lets say 3 to 5 years - that is a lot of wasted detail.

Historical and detailed collections
360 seconds as standard for 24 hour collection
Long term storage - 3 to 5 years

To get as much detail as possible in my historical data I use 6 minute samples (360) seconds. That is 10 samples per hour - 240 samples a day - and that works well in consolidated and trending views. These 24 hour samples I keep "forever" - for the life of the application and/or machine. However, I also want more detail for when "issues" crop up. For the detail I start additional collectors.

Multiple detailed views
Short term storage - 3 to 6 months

The finest detail I collect as part of a standard is 60 second samples. For other views, e.g., shift-like views I strech the sample time so that I have a whole shift or "work" period of an application. The total amount of time - divided by 240 gives me the number of seconds to sample. Initially I do not know when "shifts" start, nor do I know how long they will be. So the approach in the script is to create a "scatter" pattern of 4 and 8 hour collections (6x 4 hour collections, 3x 8 hour collections) so that I have many start moments. And for "day-night" reporting I start two more collections of 12 hours each at 6am and 6pm (0600 and 1800).

Compress early, Delete regularly

This is a lot of data - but the 4, 8 and even 12 hour collections are temporary. After 7, 14 and 21 days I compress them. After 90 to 180 days I delete them. The data is there because I might need it. In my opinioin, after three to six months their additional value is so dimished as to not need them. I only look at them when their are complaints NOW and/or for whatever reason an deep(er) analysis of an application (e.g., trending growth, (pending) capacity limitations of a whole system) is needed. Slow trend analysis can be made with the historical (6 minute) samples.

Show me the scripts!

There is a lot more I could say - and I shall in time - but for now I am going to stop and just share my script for nmon_setup.ksh (download link)

```
#!/usr/bin/ksh

# NMON_SETUP.KSH, v 1.0

# Copyright Michael Felt and AIXTOOLS.NET: 2013-2014

# All rights reserved

# Use and modification is permitted as long as origin is preserved


# NMON standarised setup for nmon collection

# TODO:

## provide arguments to set the constants

# -v volume group to create lvNMON

# -f alturnate filesystem mount-point

# -s initial size of lvNMON

## test for sufficient rights - currently executition as root is assumed


# run as root for now

# commands that do not need an argument

NMON1="fAdGKLMNOPVY^"

# arguments needing an argument

NMON2="-c 240 -I 10 -l 64 -NN -w 8 -Z 5"

# default filesystem name

FS=/var/nmon


# arguments that will be used in commands below

CMD="-${NMON1} ${NMON2} -m ${FS}"

LV=lvNMON

VG=rootvg

SZ=4G

US=root
```

```
H24='0 0'
H12='0 7,19'
H08='0 6,14,22'
H04='0 1,5,9,13,17,21'

S24="-s 360"
S12="-s 180"
S08="-s 120"
S04="-s 60"

mklv -t jfs2 -y ${LV} ${VG} ${SZ}
crfs -v jfs2 -d ${LV} -m ${FS} -A yes -u nmon

mount ${FS}
for i in 04 08 12 24
do
   mkdir -p ${FS}/$i
done

touch /var/spool/cron/crontabs/${US}

T=/tmp/cron.${US}.$$
crontab -l ${US} >${T}

print "\n##### add nmon monitors" >> ${T}
print ${H24} '* * *' /usr/bin/nmon -c240 ${S24} ${CMD}/24 >>  ${T}
print ${H12} '* * *' /usr/bin/nmon -c 240 ${S12} ${CMD}/12 >>  ${T}
print ${H08} '* * *' /usr/bin/nmon -c 240 ${S08} ${CMD}/08 >>  ${T}
print ${H04} '* * *' /usr/bin/nmon -c 240 ${S04} ${CMD}/04 >>  ${T}

print "\n##### nmon collections - compress on Sunday, cleanup 7th day of month" >> ${T}
print '0 2 * * *' "/usr/bin/find ${FS}/04 -name \"\\*.nmon\" -mtime +07 -exec bzip2 {} \"\\;\"" >>  ${T}
```

```
print '0 3 * * *' "/usr/bin/find ${FS}/08 -name \"\\*.nmon\" -mtime +14 -exec bzip2 {} \"\\;\""  >>  ${T}
print '0 4 * * *' "/usr/bin/find ${FS}/12 -name \"\\*.nmon\" -mtime +21 -exec bzip2 {} \"\\;\""  >>  ${T}
print '0 22 6 * *' "/usr/bin/find ${FS}    -name \"\\*.nmon\" -mtime +42 -exec bzip2 {} \"\\;\""  >>  ${T}
print '4 0 7 * *' "/usr/bin/find ${FS}/04 -name \"\\*.bz2\" -mtime +63 -exec rm -f {} \"\\;\""  >>  ${T}
print '8 0 7 * *' "/usr/bin/find ${FS}/08 -name\"\\*.bz2\" -mtime +198 -exec rm -f {} \"\\;\""  >>  ${T}
print '12 0 7 * *' "/usr/bin/find ${FS}/12 -name \"\\*.bz2\" -mtime +126 -exec rm -f {} \"\\;\""  >>  ${T}
print "\n##### nmon collections - move 24 hour collection to backup/archive - optional" >> ${T}
print '# 24 0 7 * *' "/usr/bin/find ${FS}/24 -mtime +126 -exec mv {} /backups/nmon/24 \"\\;\""  >>  ${T}

crontab ${T}
rm ${T}
```