AIM-7 Performance Automation

(available in PDF format as part of the install package)

Dec 13, 2012
Scott Norton

Prior to installing AIM-7 – *if needed*

```
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

# Update system if desired
cd /tmp
wget http://satserv.usa.hp.com/pub/bootstrap/bootstrap.sh
sh bootstrap.sh

# packages required to install AIM-7
yum -y install libaio-devel.x86_64                  # required
yum -y install automake-1.11.1-1.2.el6.noarch       # required
yum -y install make                                 # required
yum -y install libtool-2.2.6-15.5.el6.x86_64        # required

# Some useful tools
yum -y install numactl
yum -y install powertop
yum –y install perf
```

Installing AIM-7

- Remove old `reaim` directory contents (or save them to another directory):
  - `rm -rf /usr/local/share/reaim/*`
- Obtain the latest AIM-7 revision from:
  - http://mint-autotest.cce.hp.com/kernels/perf/
- Place it on your test machine in:
  - `/usr/local/share/reaim/.`
- Install
  - `cd  /usr/local/share/reaim`
  - `tar  xvjf  hp-osdl-aim-7.0.1.13.hp-rev7.tar.gz`
  - `./osdl-aim-7/setup`
- By default it will setup ram-based filesystems for use in `/t0,/t1,…,/t15`
  - If you want different mount points follow the instructions displayed at the end of the setup output.
- '`reaim`' and the new '`run_reaim`' script will be in `/usr/local/share/reaim`
  - They can be run from anywhere – you don't have to be in this directory

Kernel Boot Parameters

- Add the following to the boot options to your kernel(s) in `/etc/grub.conf`:
  - `ramdisk_size=131072`
- This value matches the ram-based filesystem sizes that are specified in the file:
  - `/usr/local/share/reaim/osdl-aim-7/Support_scripts/master_disk_list`
- The default `ramdisk_size` in Linux, if not specified, is `16384` (in K bytes) = `16M`
- If you don't do this you will see several "`WARNING:`" messages when the automated scripts create ram-based filesystems
  - With the current defaults in the automation scripts the AIM-7 workloads seem to be able to run fine without adding `ramdisk_size=131072` to `/etc/grub.conf` (but you will get a bunch of warnings)

AIM-7 Workloads

- AIM-7 has 14 workloads that we use:
  - all_utime
  - alltests
  - compute
  - custom
  - dbase
  - disk *(can't run with ramfs)*
  - five_sec
  - fserver
  - high_systime
  - long
  - new_dbase
  - new_fserver
  - shared
  - short
- Workloads are specified in files located in `/usr/local/share/reaim` such as:
  - `workfile.all_utime`
  - `workfile.dbase`
  - `workfile.fserver`

Running AIM-7: The Traditional Way

- Use the `reaim` command. For example:

```
– ./reaim -s100 -e1000 -t -j100 -i100 -y -f ./workfile.fserver
– ./reaim -s100 -e1000 -t -j100 -i100 -y -f ./workfile.dbase
```

- Options:
  - `-s<num>` is the starting number of users
  - `-e<num>` is the ending number of users
  - `-i<num>` is the increment in the number of users per run
  - `-j<num>` is jobs per user (which is `100` by default)
  - `-t` disables adaptive timer so that `–s` to `–e` will always be in `–i` increments
  - `-y` do not execute "sync“
- Results are output to stdout as well as to the following files:
  - `multiuser.ss` space separated “human readable” results
  - `reaim.csv` comma separated results to put into an Excel spreadsheet

Running AIM-7: The Traditional Way

```
[root@vlab380j reaim]# ./reaim -s100 -e1000 -t -j100 -i100 -y -f ./workfile.dbase

Using default config file 'reaim.config'
No logfile prefix specified, using default 'reaim'

Not executing sync during disk activity---faster execution.
DEBUG: Number of directories is 16
REAIM Workload
Times are in seconds - Child times from tms.cstime and tms.cutime

Num      Parent    Child    Child    Jobs per    Jobs/min/   Std_dev   Std_dev   JTI
Forked   Time      SysTime  UTime    Minute      Child       Time      Percent
100      4.86      2.04     56.11    122222.22   1222.22     0.62      13.93     86
200      9.75      3.91     112.68   121846.15   609.23      1.93      21.28     78
300      14.64     6.03     168.87   121721.31   405.74      0.93      6.62      93
400      19.51     8.07     225.31   121783.70   304.46      0.52      2.70      97
500      24.37     9.88     281.70   121871.15   243.74      0.28      1.17      98
600      29.18     11.66    337.69   122138.45   203.56      0.64      2.21      97
700      34.09     13.61    394.74   121971.25   174.24      0.35      1.04      98
800      39.00     15.79    451.29   121846.15   152.31      0.45      1.17      98
900      43.81     17.61    507.12   122026.93   135.59      0.45      1.04      98
1000     48.63     19.71    562.61   122146.82   122.15      0.50      1.05      98
Max Jobs per Minute 122222.2
```

Running AIM-7: The Traditional Way

ram-based FS vs. disk-based FS

- We typically run AIM-7 with a ram-based file system (unless we really want to test a specific disk device).
- To use a ram-based file system follow the example below:

```
    ./osdl-aim-7/Support_scripts/do_make_file_systems

    ./osdl-aim-7/Support_scripts/do_mount_file_systems
    ./osdl-aim-7/Support_scripts/do_clean_file_systems   # remove existing files

    ./reaim -s100 -e1000 -j100 -i100 -y -f ./workfile.fserver

    ./osdl-aim-7/Support_scripts/do_clean_file_systems   # remove existing files
    ./osdl-aim-7/Support_scripts/do_umount_file_systems
```

- Note: files persist on a file system such as `ext4` with `ramdisk` between umount and (re)mount. This is not true for `ramfs`.
- Note: `ramfs` releases memory back to the system for reuse after umount, `ext4` with `ramdisk` does not

Running AIM-7: The Traditional Way

- If you are analyzing the performance of a specific workload at a specific user level load, the "Traditional Way" works just fine
- However, if you want to run
  - All or many workloads
  - Use ram-based file systems
  - Have the full range of data points from 10 users to 2000 users
  - Have averaged results instead of single data points

and then have to save away the results and do this for multiple kernels to compare can be a lot of manual work

Running AIM-7: `run_reaim` script

- The new `run_reaim` script will automate all of this. By default it:
  - Runs all workloads (except aim9)
  - Runs 10 to 90 users in increments of 10 and 100 to 2000 users in increments of 100
  - Uses a ram-based file system
  - Saves data in `<workload>.ssv` and `<workload>.csv` files in the `./results` directory (for example, `./results/compute.ssv` and `./results/compute.csv`)
  - Successive `run_reaim` runs add data to the end of the `.ssv` and `.csv` files
- Output from the `run_reaim` script (including the `reaim` command output) is redirected to `./results/run_reaim.out`
  - This file is also copied to `./results/run_reaim.out.$$` so future runs won't over-write your previous output.
- Look through the `./results/run_reaim.out` file for any errors after running the workloads

Running AIM-7: `run_reaim` script

- After running `run_reaim` the output directory `./results` looks like:

```
all_utime.csv  custom.ssv    fserver.csv       new_dbase.ssv         short.csv
all_utime.ssv  dbase.csv     fserver.ssv       new_fserver.csv       short.ssv
alltests.csv   dbase.ssv     high_systime.csv  new_fserver.ssv
alltests.ssv   disk.csv      high_systime.ssv  run_reaim.out
compute.csv    disk.ssv      long.csv          run_reaim.out.3868
compute.ssv    five_sec.csv  long.ssv          shared.csv
custom.csv     five_sec.ssv  new_dbase.csv     shared.ssv
```

  - The `.ssv` files are space separated values that are “human readable”
  - The `.csv` files are comma separated values that can be read into an Excel spreadsheet
    - More on this later in this slide set
- Each time `run_reaim` is run it will add data to the end of the `.csv` and `.ssv` files
  - When you want to start fresh, either remove all these files or specify the `-r` option to `run_reaim`

Running AIM-7: `run_reaim` script

Sample dbase.ssv output file

```
Title:    hydrazine0  DL980  80-core
Kernel:   hydrazine0  RHEL   2.6.32-279.el6.x86_64   HT-disabled   ramfs
Date:     Thu Nov  8 02:06:16 MST 2012
Command:  /usr/local/share/reaim/osdl-aim-7/reaim -s10 -e2000 -t -j100 -i[10|100] -y -f ./workfile.dbase
Workload: dbase
Time:     2 mins, 57 secs
                 Jobs/min/  Jobs/sec/  Time:   Time:  Time:   Time:           Running child time
Forks  Jobs/min     child      child    parent  childU childS  std_dev   JTI   :max  :min
10     99000.00   9900.00    165.00     0.60    4.28   0.22    0.07   87.00  0.60  0.44
20    237600.00  11880.00    198.00     0.50    8.65   0.48    0.02   96.00  0.50  0.44
30    187578.95   6252.63    104.21     0.95   13.14   0.64    0.14   78.00  0.95  0.44
40    440000.00  11000.00    183.33     0.54   17.74   0.87    0.02   94.00  0.53  0.44
50    550000.00  11000.00    183.33     0.54   22.67   1.11    0.02   95.00  0.54  0.46
60    548307.69   9138.46    152.31     0.65   27.71   1.31    0.05   91.00  0.65  0.49
70    540000.00   7714.29    128.57     0.77   32.40   1.59    0.06   91.00  0.77  0.46
80    505531.91   6319.15    105.32     0.94   36.93   1.74    0.11   87.00  0.94  0.46
90    712800.00   7920.00    132.00     0.75   41.90   2.04    0.07   89.00  0.75  0.51
100   813698.63   8136.99    135.62     0.73   46.86   2.33    0.03   95.00  0.73  0.60
200   886567.16   4432.84     73.88     1.34   93.68   4.73    0.04   96.00  1.34  1.10
300   909183.67   3030.61     50.51     1.96  140.62   7.20    0.06   96.00  1.96  1.65
400   876752.77   2191.88     36.53     2.71  187.50   9.35    0.08   96.00  2.71  2.16
500   931034.48   1862.07     31.03     3.19  234.80  11.71    0.11   96.00  3.19  2.53
600   909183.67   1515.31     25.26     3.92  281.97  13.99    0.15   95.00  3.92  2.70
700   932287.00   1331.84     22.20     4.46  329.00  16.34    0.14   96.00  4.45  3.73
800   939130.43   1173.91     19.57     5.06  376.05  18.97    0.19   96.00  5.06  3.78
900   936252.19   1040.28     17.34     5.71  423.26  21.19    0.19   96.00  5.71  4.73
1000  913846.15    913.85     15.23     6.50  470.60  23.67    0.23   96.00  6.49  5.15
1100  860869.57    782.61     13.04     7.59  518.45  25.68    0.28   96.00  7.59  6.08
1200  897733.00    748.11     12.47     7.94  566.01  28.19    0.29   96.00  7.93  6.35
1300  937135.92    720.87     12.01     8.24  612.61  31.01    0.30   96.00  8.24  5.77
1400  924000.00    660.00     11.00     9.00  660.13  33.43    0.36   95.00  9.00  6.66
1500  936908.52    624.61     10.41     9.51  707.30  35.70    0.38   95.00  9.51  7.63
1600  939130.43    586.96      9.78    10.12  754.62  38.29    0.35   96.00 10.11  8.33
1700  942857.14    554.62      9.24    10.71  801.78  40.33    0.42   95.00 10.70  8.36
1800  936252.19    520.14      8.67    11.42  849.15  42.93    0.46   95.00 11.42  8.72
1900  938154.61    493.77      8.23    12.03  897.33  45.33    0.43   96.00 12.03  9.36
2000  939873.42    469.94      7.83    12.64  944.26  47.93    0.50   95.00 12.64  8.05

Highest JPM: 942857.14  at  1700  users
*********************  Data Separator  *****************************
```

Running AIM-7: `run_reaim` script

Sample dbase.csv output file – to put in an excel spreadsheet

```
Title:,hydrazine0  DL980  80-core w/ ramfs
Kernel:,hydrazine0  RHEL  2.6.32-279.el6.x86_64  HT-disabled  ramfs
Date:,Thu Nov  8 02:06:16 MST 2012
Command:, /usr/local/share/reaim/osdl-aim-7/reaim -s10 -e2000 -j100 -t -i[10|100] -y -f ./workfile.dbase
Workload:,dbase
Time:,mins:,2,secs:,57
Forks,JPM,JPM_C,JPS_C,parent_tm,childU_tm,childS_tm,std_dev,JTI,max_c,min_c
10,99000.00,9900.00,165.00,0.60,4.28,0.22,0.07,87.00,0.60,0.44
20,237600.00,11880.00,198.00,0.50,8.65,0.48,0.02,96.00,0.50,0.44
30,187578.95,6252.63,104.21,0.95,13.14,0.64,0.14,78.00,0.95,0.44
40,440000.00,11000.00,183.33,0.54,17.74,0.87,0.02,94.00,0.53,0.44
50,550000.00,11000.00,183.33,0.54,22.67,1.11,0.02,95.00,0.54,0.46
60,548307.69,9138.46,152.31,0.65,27.71,1.31,0.05,91.00,0.65,0.49
70,540000.00,7714.29,128.57,0.77,32.40,1.59,0.06,91.00,0.77,0.46
80,505531.91,6319.15,105.32,0.94,36.93,1.74,0.11,87.00,0.94,0.46
90,712800.00,7920.00,132.00,0.75,41.90,2.04,0.07,89.00,0.75,0.51
100,813698.63,8136.99,135.62,0.73,46.86,2.33,0.03,95.00,0.73,0.60
200,886567.16,4432.84,73.88,1.34,93.68,4.73,0.04,96.00,1.34,1.10
300,909183.67,3030.61,50.51,1.96,140.62,7.20,0.06,96.00,1.96,1.65
400,876752.77,2191.88,36.53,2.71,187.50,9.35,0.08,96.00,2.71,2.16
500,931034.48,1862.07,31.03,3.19,234.80,11.71,0.11,96.00,3.19,2.53
600,909183.67,1515.31,25.26,3.92,281.97,13.99,0.15,95.00,3.92,2.70
700,932287.00,1331.84,22.20,4.46,329.00,16.34,0.14,96.00,4.45,3.73
800,939130.43,1173.91,19.57,5.06,376.05,18.97,0.19,96.00,5.06,3.78
900,936252.19,1040.28,17.34,5.71,423.26,21.19,0.19,96.00,5.71,4.73
1000,913846.15,913.85,15.23,6.50,470.60,23.67,0.23,96.00,6.49,5.15
1100,860869.57,782.61,13.04,7.59,518.45,25.68,0.28,96.00,7.59,6.08
1200,897733.00,748.11,12.47,7.94,566.01,28.19,0.29,96.00,7.93,6.35
1300,937135.92,720.87,12.01,8.24,612.61,31.01,0.30,96.00,8.24,5.77
1400,924000.00,660.00,11.00,9.00,660.13,33.43,0.36,95.00,9.00,6.66
1500,936908.52,624.61,10.41,9.51,707.30,35.70,0.38,95.00,9.51,7.63
1600,939130.43,586.96,9.78,10.12,754.62,38.29,0.35,96.00,10.11,8.33
1700,942857.14,554.62,9.24,10.71,801.78,40.33,0.42,95.00,10.70,8.36
1800,936252.19,520.14,8.67,11.42,849.15,42.93,0.46,95.00,11.42,8.72
1900,938154.61,493.77,8.23,12.03,897.33,45.33,0.43,96.00,12.03,9.36
2000,939873.42,469.94,7.83,12.64,944.26,47.93,0.50,95.00,12.64,8.05
High JPM:, 942857.14,1700,users,
*********,***********  Data Separator  *****************************
```

Running AIM-7: `run_reaim` options

```
Usage: run_reaim

        -k <kernel>      # name to appear on graph line
        -t <title>       # name to appear on graph title (i.e., system name/type)
        -d <log_dir>     # output logging directory (default: "./results")
        -c <cmd>         # reaim preface (for ex: -c "chrt --fifo 3")
        -r               # remove old logging directory files first
        -h               # help menu

        -ramfs           # use a ramfs for workloads
        -ext4_ramdisk    # use an ext4 ramdisk for workloads (default)
        -ext3_ramdisk    # use an ext3 ramdisk for workloads
        -ext2_ramdisk    # use an ext2 ramdisk for workloads
        -diskfs          # use a diskfs for workloads

        -avg <num>       # average <num> runs

        -perf            # use perf record/report - output is in ./results/<wkload>.perf

        -x1              # do *not* run 10-90     users option
        -x2              # do *not* run 100-1000  users option
        -x3              # do *not* run 1100-2000 users option
        -x4              # do *not* run $OPTS4    users option (currently unused)
        -x5              # do *not* run $OPTS5    users option (currently unused)
        -x6              # do *not* run $OPTS6    users option (currently unused)

        -<workload>      # do *not* run <workload>

        +<workload>      # only run <workload>

        -aim9only        # only run aim9

    -k <kernel> and -t <title> are required

Avail Workloads:  short       dbase         new_dbase   compute       all_utime
                  disk        five_sec      long        shared        alltests
                  custom      new_fserver   fserver     high_systime  aim9
```

Running AIM-7: `run_reaim` examples

- Run with default options (all workloads, 10-2000 users, ext4 ram-based fs):

```
./run_reaim -k “my kernel” -t “my machine”
```

- Run all workloads with only 100 to 1000 users:

```
./run_reaim -k “my kernel” -t “my machine” -x1 -x3
```

- Run only the dbase and fserver workloads:

```
./run_reaim -k “my kernel” -t “my machine” +dbase +fserver
```

- Run all workloads except short and custom:

```
./run_reaim -k “my kernel” -t “my machine” -short -custom
```

- Run only the dbase workload with only 100 to 1000 users:

```
./run_reaim -k “my kernel” -t “my machine” -x1 -x2 +dbase
```

- Run all workloads with disk-based filesystems (as opposed to a ram-based fs)

```
./run_reaim -k “my kernel” -t “my machine” -diskfs
```

- Run all workloads using real-time scheduling:

```
./run_reaim -k “my kernel” -t “my machine” -c “chrt --fifo 3”
```

Running AIM-7: `run_reaim` examples

- At this point in the presentation show a live example of installing AIM-7 and using the `run_reaim` script…
  - `cd  /usr/local/share/reaim`
  - `tar  xvjf  hp-osdl-aim-7.0.1.13.hp-rev7.tar.gz`
  - `./osdl-aim-7/setup`
  - `./run_reaim –k “my kernel” –t “my machine” +dbase`
  - `more ./results/run_reaim.out`
  - `more ./results/dbase.ssv`
  - `more ./results/dbase.csv`

Running AIM-7: `run_reaim` script

Average option

- To run a workload multiple times and average each data point use the "`-avg <num>`" option
- Each individual run is still added to the end of `./results/<workload>.ssv` and `./results/<workload>.csv`
- Two new files are created that contain the averaged data from `<num>` runs:
  - `./results/<workload>.avg.ssv`
  - `./results/<workload>.avg.csv`
- As with the other results files, future "`-avg <num>`" runs add to the end of the existing `<workload>.avg.ssv` and `<workload>.avg.csv` files

Running AIM-7: run_reaim –avg <num>

Sample dbase.avg.ssv output file

```
Title:    hydrazine0  DL980  80-core w/ ramfs
Kernel:   average - hydrazine0  RHEL  2.6.32-279.el6.x86_64  HT-disabled  ramfs
Date:     Sat Nov 17 18:27:56 MST 2012
Command:  ./reaim -s10 -e2000 -t -j100 -i[10|100] -y -f ./workfile.dbase
Workload: dbase
Time:
Jobs           Avg       Median     High       Low H/L_Diff  H/L_Pct        Val1      Diff1    Pct1         Val2      Diff2    Pct2      ...    ...
----           ---       ------     ----       --- --------  -------        ----      -----    ----         ----      -----    ----      ---    ---
10       113464.18    126382.98   132000     63870    68130    51.61    63870.97  -49593.21 -43.70%     72439.02  -41025.16 -36.15%
20       198051.83    235270.59   252765    113142   139623    55.23   113142.86  -84908.97 -42.87%    127741.94  -70309.89 -35.50%
30       283815.23    330453.30   356400    147272   209128    58.67   147272.73 -136542.50 -48.10%    173009.71 -110805.52 -39.04%
40       388159.05    420827.58   484897    220000   264897    54.62   220000.00 -168159.05 -43.32%    240000.00 -148159.05 -38.16%
50       461464.75    480156.45   582352    270000   312352    53.63   270000.00 -191464.75 -41.49%    309375.00 -152089.75 -32.95%
60       571000.90    604067.80   698823    307241   391582    56.03   307241.38 -263759.52 -46.19%    318214.29 -252786.61 -44.27%
70       664880.87    784528.30   799615    270000   529615    66.23   270000.00 -394880.87 -59.39%    374594.59 -290286.28 -43.65%
80       753332.80    800067.91   896603    409655   486948    54.31   409655.17 -343677.63 -45.62%    552558.14 -200774.66 -26.65%
90       787283.25    810000.00   848571    540000   308571    36.36   540000.00 -247283.25 -31.40%    763714.29  -23568.96  -2.99%
100      778201.47    813698.63   836619    456923   379696    45.38   456923.08 -321278.39 -41.28%    751898.73  -26302.74  -3.37%
200      821775.11    845562.31   893233    686705   206528    23.12   686705.20 -135069.91 -16.43%    694736.84 -127038.27 -15.45%
300      854743.19    886567.16   904568    690697   213871    23.64   690697.67 -164045.52 -19.19%    778165.94  -76577.25  -8.95%
400      900132.93    906870.23   917374    854676    62698     6.83   854676.26  -45456.67  -5.04%    864000.00  -36132.93  -4.01%
500      916026.63    915256.41   928125    900000    28125     3.03   900000.00  -16026.63  -1.74%    905487.80  -10538.83  -1.15%
600      919634.54    918562.80   935433    904568    30865     3.29   904568.53  -15066.01  -1.63%    911508.95   -8125.59  -0.88%
700      927521.82    932287.00   942857    888461    54396     5.76   888461.54  -39060.28  -4.21%    905882.35  -21639.47  -2.33%
800      925719.50    932679.99   939130    842553    96577    10.28   842553.19  -83166.31  -8.98%    924513.62   -1205.88  -0.13%
900      930549.78    933799.83   942857    910732    32125     3.40   910732.54  -19817.24  -2.12%    912286.69  -18263.09  -1.96%
1000     933197.42    935435.39   942857    912442    30415     3.22   912442.40  -20755.02  -2.22%    919504.64  -13692.78  -1.46%
1100     941737.97    942857.14   945586    933428    12158     1.28   933428.57   -8309.40  -0.88%    936103.15   -5634.82  -0.59%
1200     937213.62    939130.43   946613    923316    23297     2.46   923316.06  -13897.56  -1.48%    928125.00   -9088.62  -0.96%
1300     939440.62    941133.80   944009    925899    18110     1.91   925899.28  -13541.34  -1.44%    934866.83   -4573.79  -0.48%
1400     936937.54    939662.21   945000    907860    37140     3.93   907860.26  -29077.28  -3.10%    933333.33   -3604.21  -0.38%
1500     940625.11    940865.89   945859    936908     8951     0.94   936908.52   -3716.59  -0.39%    936908.52   -3716.59  -0.39%
1600     937469.32    939131.35   943793    924513    19280     2.04   924513.62  -12955.70  -1.38%    928125.00   -9344.32  -0.99%
1700     937144.14    941099.72   942857    920510    22347     2.37   920510.48  -16633.66  -1.77%    929834.25   -7309.89  -0.78%
1800     936369.48    939956.22   942026    903040    38986     4.13   903040.54  -33328.94  -3.55%    934615.38   -1754.10  -0.18%
1900     939005.55    938935.11   942070    934271     7799     0.82   934271.52   -4734.03  -0.50%    935820.90   -3184.65  -0.33%
2000     940312.81    940617.58   944356    936908     7448     0.78   936908.52   -3404.29  -0.36%    937647.99   -2664.82  -0.28%

Highest JPM Average:   941737  at  1100 users
Highest JPM Median:    942857  at  1100 users
Highest JPM Single:    946613  at  1200 users
*********************  Data Separator  *****************************
```

Running AIM-7: `-perf` option

- The `-perf` option causes each workload to be run with:
  - `perf record -a -g -s`
- At the end of each workload run a perf report is generated with:
  - `perf report -n --stdio`
- Output files are located in the directory `./results/<workload>.perf`
- Three perf output files per workload run:
  - `<workload>.10_90.perf.<pid>.out          # 10-90 users`
  - `<workload>.100_1000.perf.<pid>.out       # 100-1000 users`
  - `<workload>.1100_2000.perf.<pid>.out      # 1100-2000 users`
- The `<pid>` is added to the filename so future runs will not overwrite previous data

Sysctl values

- Running the AIM-7 workloads requires some sysctl values to be increased depending on what your system currently has set
- The ones that we have found so far have been incorporated into `run_reaim` so you don't have to do it:
    - kernel.sem
    - kernel.shmmax
    - kernel.shmall
    - net.netfilter.nf_conntrack_max
    - net.ipv4.tcp_fin_timeout
    - net.ipv4.tcp_keepalive_time
    - net.ipv4.tcp_keepalive_intvl
    - net.ipv4.tcp_tw_recycle
    - net.ipv4.tcp_tw_reuse
- If `run_reaim` modifies any of these values it will be displayed at the beginning of the `./results/run_reaim.out` output file
- If you find others that need to be added please let us know

Different `reaim` run options

- If you want different run options to `reaim`, you can make a copy of `run_reaim` and modify the new copy.
- In your copy look for the following lines near the top of the file:

```
OPTS1="-s10   -e90   -t -j100 -i10  -y"
OPTS2="-s100  -e1000 -t -j100 -i100 -y"
OPTS3="-s1100 -e2000 -t -j100 -i100 -y“

# not defined now - user can modify as desired
OPTS4=""
OPTS5=""
OPTS6=""
```

- You can modify the above defines for your specific options (without changing the existing behavior). As an example, you can do something like:

```
OPTS4="-s2100  -e3000 -t -j100 -i100 -y"
```

to run 2100 to 3000 users since the current values stop at 2000 users

- Once modified you can run your script
  - Running with options `-x1 -x2 -x3` will cause it to run only with your new options

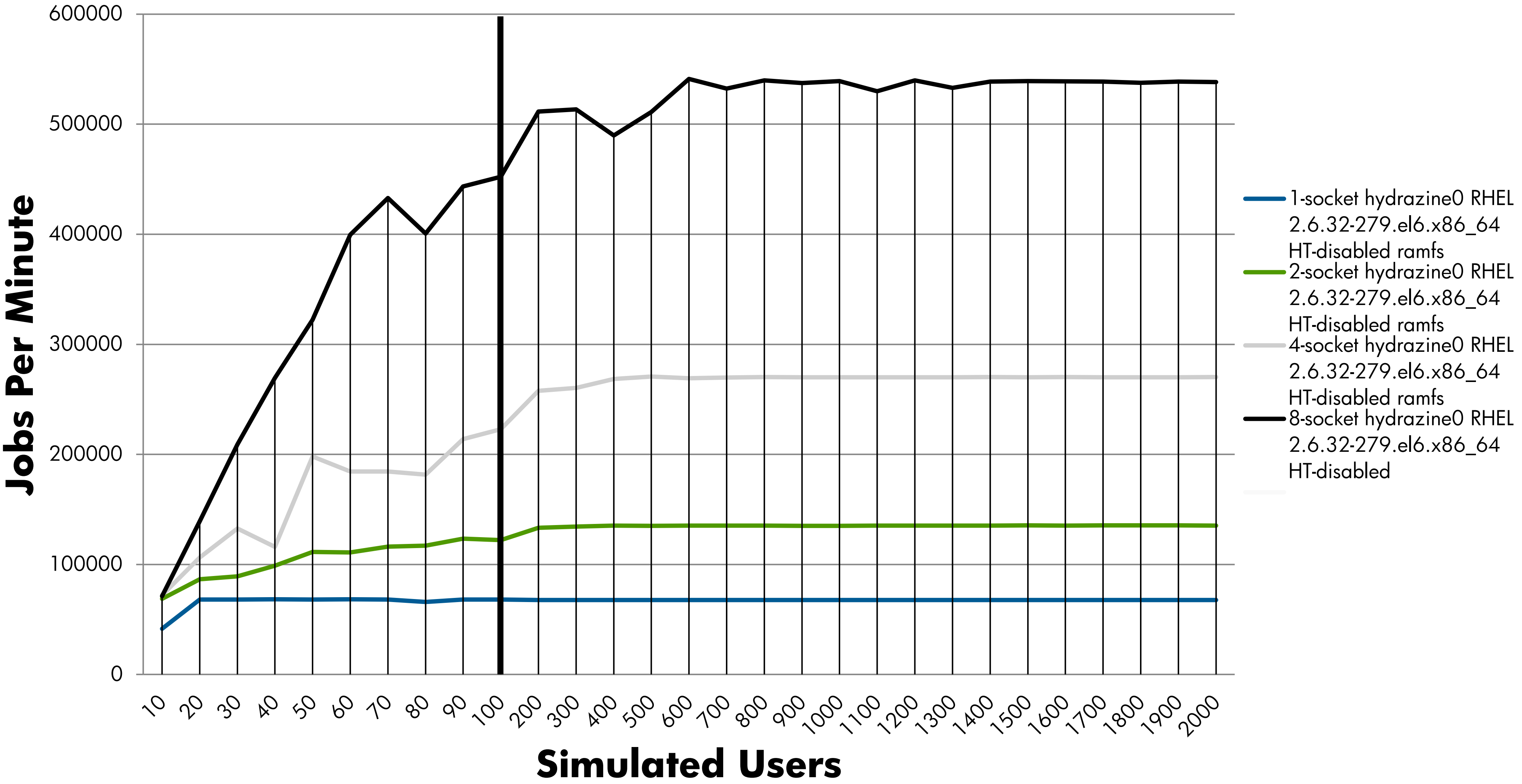
Graphing AIM-7 Output

- Each successive run of `run_reaim` adds data to the end of the `<workload>.ssv` and `<workload>.csv` files
- This is done so that `<workload>.csv` can be incorporated into an Excel spreadsheet to graph the results
  - The same is done with the `<workload>.avg.csv` file
- Graphed Excel spreadsheet templates are located at

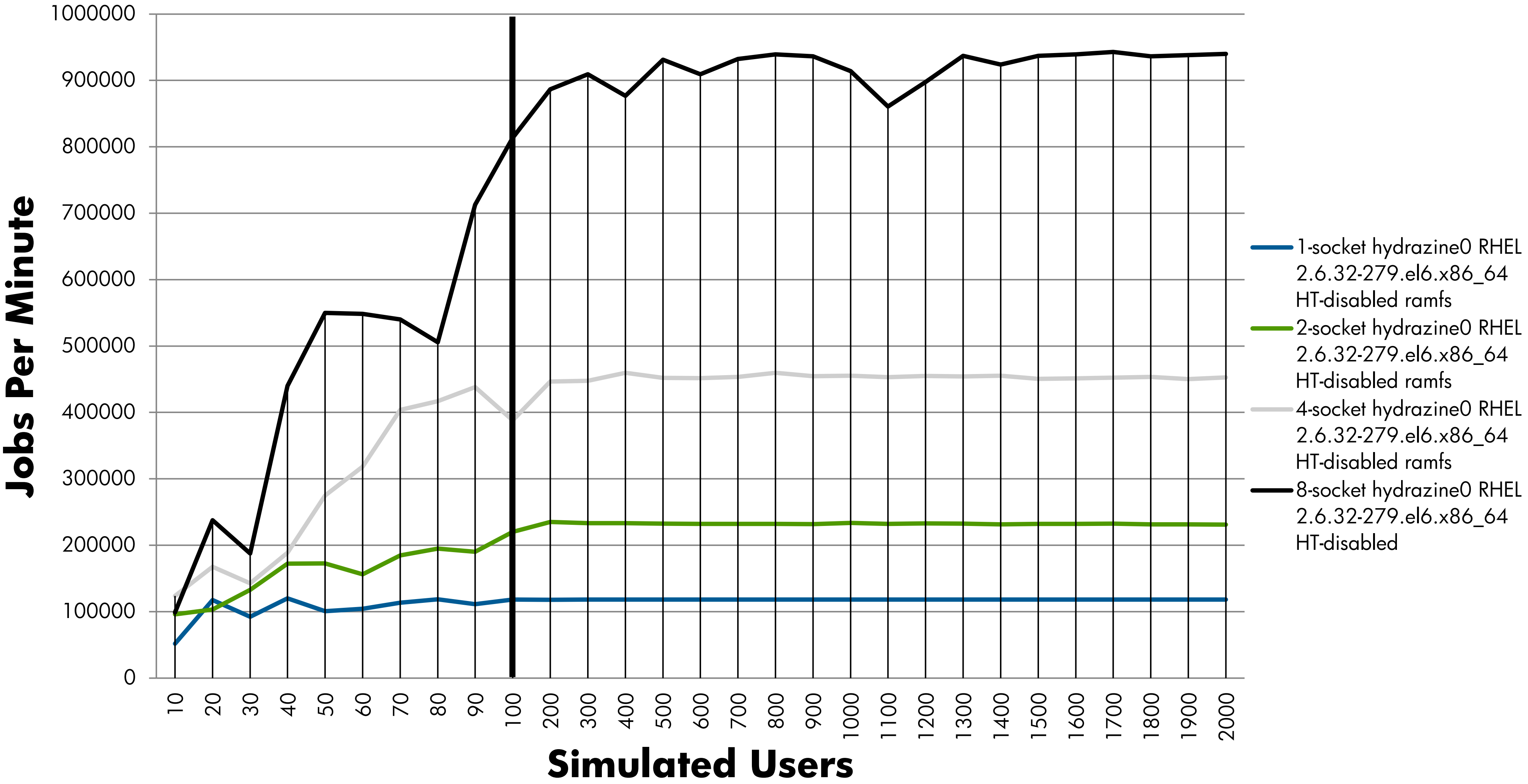
```
/usr/local/share/reaim/results/aim_graph_template.xlsm
/usr/local/share/reaim/results/aim_graph_average_template.xlsm
```

  - There is a worksheet tab for each of the 14 workloads in the Excel template
- The following slides show some sample graphs after incorporating `<workload>.csv` files to the template

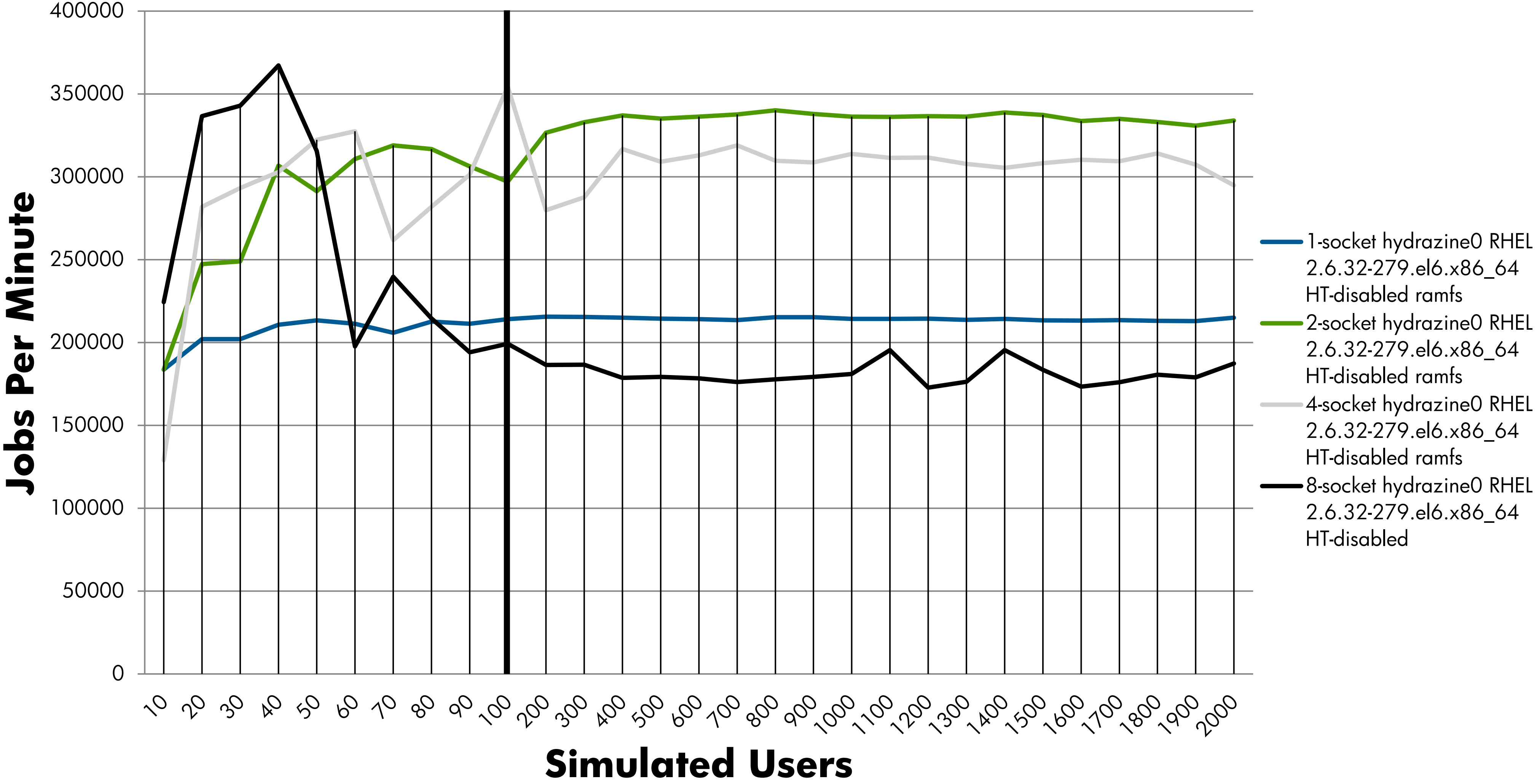
AIM7 – compute workload graph



AIM7 - dbase workload graph



AIM7 - fserver workload graph



Graphing AIM-7 Output

- You should be really explicit and thorough about what you specify with the "`-k <kernel>`" option as this is what will distinguish your excel graph lines
  - In the future you may want to manually merge data from different runs
  - I've learned to specify things like HT-enabled/HT-disabled, ramfs/diskfs, power_max/power_min, etc in the `-k <kernel>` option rather than the `-t <title>` option because of this
  - Naming is important!

Incorporating AIM-7 Output in Excel

- Instructions for Office 2007
  - Copy your `results` directory to a Windows machine (i.e., laptop or desktop)
  - Copy the `aim_graph_template.xlsm` template to the same directory
  - Open the Excel template: `aim_graph_template.xlsm`
    - If under the menu options (just above the opened file) you see:

      Security Warning some active content has been disabled Options…
      - Click the Options… box
      - A **"Security Alert"** dialog box will open
        - Select `"Enable this content"` in both the **"Macro"** and **"Data Connection"** sections
        - Click `"OK"`
        - Note: only need to do this when opening the file to update data, for viewing select `"Help protect me"`
  - Select the `"UPDATE"` worksheet at the bottom
  - Click on the `"Press to Update Data"` button
    - If any `.csv` file is missing the data in the tab for that workload will be empty – previous data is cleared
  - Save this file to a filename of your choice

Incorporating AIM-7 Output in Excel

- In the presentation let's do a live example with:
  - `./sample`: Walk through a sample `.xlsm` file
  - `./add_to_excel`: Demo incorporating `.csv` data into `aim_graph_template.xlsm`
  - `cp ./copy_from/* add_to_excel/.` and incorporate the new data
  - `./average`: Demo incorporating `.avg.csv` data into `aim_graph_average_template.xlsm`
  - `./average`: Demo changing line color and style from average data that was put in `aim_graph_average_template.xlsm` and `aim_graph_template.xlsm`
  - `./average`: Demo incorporating the `.csv` data used to create the average data into `aim_graph_template.xlsm`
- Note: the `.xlsm` templates are sensitive to the format of the `.csv` files – don't change these output files

How simple is it?

- To get the excel spreadsheet showing 1/2/4/8 socket DL980 scaling I wrote this script named `./my_per_socket_scr`:

```
./run_reaim -k “1-socket” -t “DL980” -c “numactl -m 0   -N 0“
./run_reaim -k “2-socket” -t “DL980” -c “numactl -m 0,1 -N 0,1"
./run_reaim -k “4-socket” -t “DL980” -c “numactl -m 0-3 -N 0-3"
./run_reaim -k “8-socket” -t “DL980”    # no need for numactl – run all
```

- Ran: `nohup ./my_per_socket_scr &`
- 12 hours later I copied the `./results` directory to my laptop
- Incorporated the `.csv` files into the Excel spreadsheet template
- All in all it took about 15-20 minutes of my time to write the script, start the run, copy the results and incorporate them into excel
- Obviously it will take longer when you are comparing different kernels that you will have to boot for each run!

How simple is it? The Traditional Way

- To do the previous slide's socket scaling runs the "Traditional Way" would mean:
  - 14 AIM-7 workloads times 3 `reaim` commands per workload = 42 manual `reaim` commands
  - times 4 different socket configurations = 168 manual `reaim` commands
  - times 2 to also run with diskfs = 336 manual `reaim` commands (data not previously shown)
  - times 2 to also run with HT-enabled = 672 manual `reaim` commands (data not previously shown)
- Additionally, after each `reaim` command completes you have to manually save away the `multiuser.ss` and `reaim.csv` data files so the next run doesn't overwrite the data
- Finally you get to figure out how to pull all of that data together and manually create graphs to display all the results
- That's a lot of manual work by hand…

How long does it take to run?

- All workloads:
  - 12-core DL380, RHEL 6.3, HT-disabled, ramfs: ~ 4 hrs
  - 12-core DL380, RHEL 6.3, HT-disabled, diskfs: ~ 4 hrs
  - 80-core DL980, RHEL 6.3, HT-disabled, ramfs: ~ 2.5 hrs
  - 80-core DL980, RHEL 6.3, HT-disabled, diskfs: ~ 4 to 5 hrs
- Individual workloads on an 80-core DL980, RHEL 6.3, HT-disabled, ramfs:

all_utime	~ 7 mins
alltests	~ 6 mins
compute	~ 5 mins
custom	~ 8 mins
dbase	~ 3 mins
disk *(no ramfs)*	~ 7 mins
five_sec	~ 14 mins
fserver	~ 13 mins
high_systime	~ 45 mins
long	~ 7 mins
new_dbase	~ 3 mins
new_fserver	~ 13 mins
shared	~ 5 mins
short	~ 2 mins

Backup Excel Slides

Manually Incorporating Output in Excel

- Instructions for Office 2007
  - Open the Excel template: `aim_graph_template.xlsm`
  - Click on the "Data" menu tab
  - For each worksheet tab (e.g., "all_utime", "compute", etc at the bottom)
    - Click on the worksheet tab you want
    - Select columns A-K, right-click, select "Clear Contents"
      - Click "Yes" on the pop-up dialog box
    - Click on cell A1
    - Click "From Text" on the Data menu tab – this will bring up an open dialog box
    - Navigate to the `.csv` file you want to incorporate and open the file
    - Click "Next" on first Import Wizard box ("Delimited" should be selected)
    - Click "Comma" on the second Import Wizard box, followed by "Next"
    - Click "Finish" on the third Import Wizard box ("General" should be selected)
    - In the Import Data box that appears, make sure it states "=A1", click "OK"