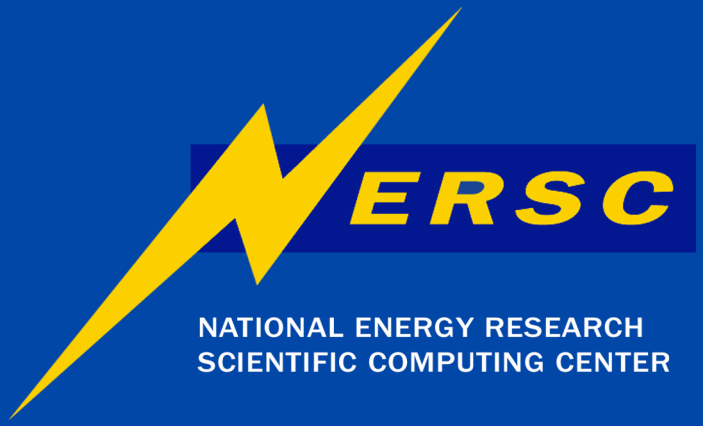


National Energy Research Scientific Computing Center (NERSC)

Detecting System Problems With Application Exit Codes

Nicholas P. Cardo
NERSC Center Division, LBNL
CUG 2008, Helsinki

The Big Problem

- System getting larger, more complicated to detect problems
- More difficult to detect node health issues
- Applications are scaling to new heights
- Need to detect problems before users
- Can we say.. “needle in a haystack”

Challenges

- How to detect a failure
- What is an application failure
- Redirected stdout/stderr, can't find error messages
- What is a system failure

```
$ acctrep -u cardo
                                                  TTY                                                    CPU Time
Command             Flags  User      Group      Maj/Min      Start Time          End Time          User   Systm    pid    ppid    Exit
-----------------   -----  --------  --------   -------  -------------------  -------------------  -----  -----  ------  ------  -----
 bash               -----  cardo     cardo      136/003  05/04/2008 23:46:12  05/06/2008 09:16:57      4      2    9380    9379      0
#sshd               FS---  cardo     cardo      ---/---  05/04/2008 23:46:11  05/06/2008 09:18:23      1      1    9379    9377      0
 xauth              -----  cardo     cardo      ---/---  05/05/2008 00:06:18  05/05/2008 00:06:18      0      0    9882    9881      0
 sh                 -----  cardo     cardo      ---/---  05/05/2008 00:06:18  05/05/2008 00:06:19      0      0    9881    9880      0
 sshd               F----  cardo     cardo      ---/---  05/05/2008 00:06:18  05/05/2008 00:06:19      0      0    9880       1      0
#sshd               FS---  cardo     cardo      ---/---  05/04/2008 22:23:25  05/13/2008 20:12:15      0      0    8600    8598  65280
 bash               ----X  cardo     cardo      ---/---  05/04/2008 22:23:26  05/13/2008 20:13:19      2      2    8601       1      1
 xauth              -----  cardo     cardo      136/001  05/05/2008 05:28:56  05/05/2008 05:28:59      0      0   31592   31591      0
 ls                 -----  cardo     cardo      136/001  05/05/2008 05:28:58  05/05/2008 05:31:35      2      0   31594   31593      0
 bash               F----  cardo     cardo      136/001  05/05/2008 05:28:58  05/05/2008 05:31:35      0      0   31593   31591      0
 tty                -----  cardo     cardo      136/001  05/05/2008 05:28:59  05/05/2008 05:28:59      0      0   31596   31595      0
 bash               F----  cardo     cardo      136/001  05/05/2008 05:28:59  05/05/2008 05:28:59      0      0   31595   31591      0
 hostname           -----  cardo     cardo      136/001  05/05/2008 05:28:59  05/05/2008 05:28:59      0      0   31598   31597      0
 bash               F----  cardo     cardo      136/001  05/05/2008 05:28:59  05/05/2008 05:28:59      0      0   31597   31591      0
```

It is more than just a number

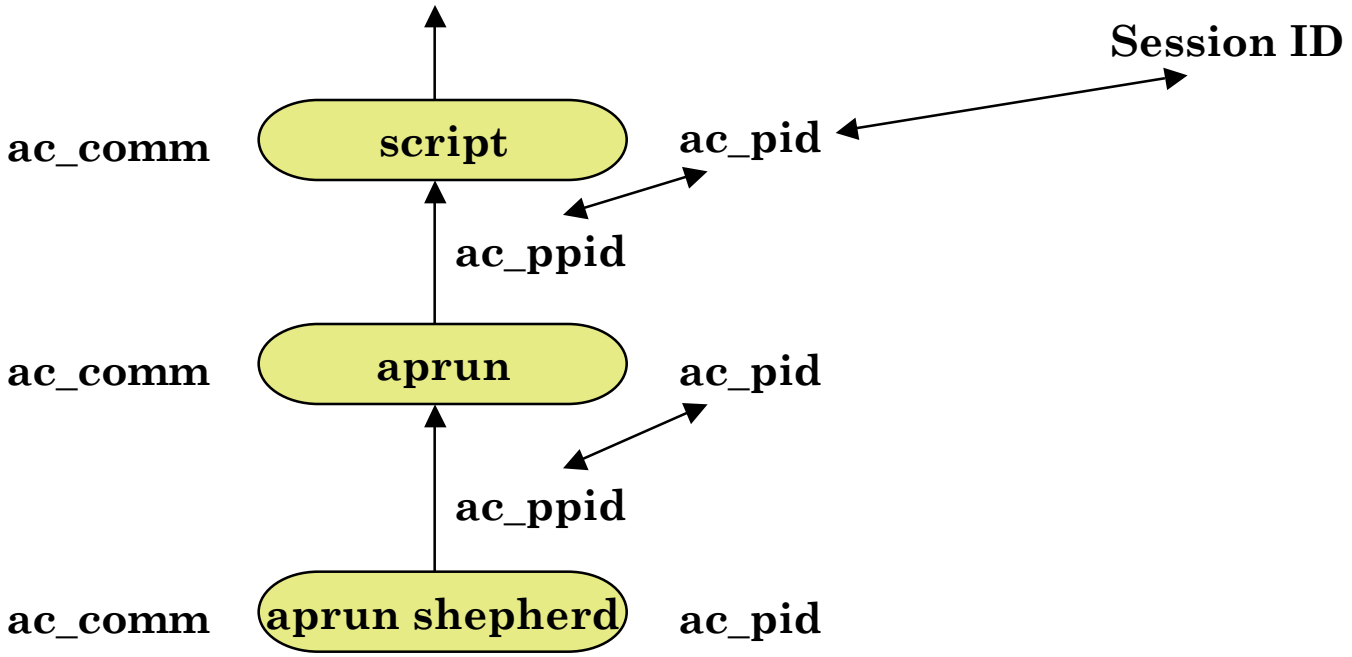
```
char  ac_flag;     /* Flags              */
char  ac_version;  /* ACCT_VERSION       */
__u16 ac_tty;      /* Control Terminal   */
__u32 ac_exitcode; /* Exitcode           */
__u32 ac_uid;      /* Real User ID       */
__u32 ac_gid;      /* Real Group ID      */
__u32 ac_pid;      /* Process ID         */
__u32 ac_ppid;     /* Parent Process ID  */
__u32 ac_btime;    /* Creation Time      */
#ifdef __KERNEL__
__u32 ac_etime;    /* Elapsed Time       */
#else
float  ac_etime;   /* Elapsed Time       */
#endif
comp_t ac_utime;   /* User Time          */
comp_t ac_stime;   /* System Time        */
comp_t ac_mem;     /* Avg Memory Usage   */
comp_t ac_io;      /* Chars Transferred  */
comp_t ac_rw;      /* Blocks Read/Write  */
comp_t ac_minflt;  /* Minor Pagefaults   */
comp_t ac_majflt;  /* Major Pagefaults   */
comp_t ac_swaps;   /* Number of Swaps    */
char  ac_comm[ACCT_COMM]; /* Command     */
```

BSD v3 Structure

For Tracing

Lots of Good Stuff

Process Tree



Job Exit Classifications

- SUCCESS: All apruns within a single batch job completed with an exit code of 0. No further analysis required.
- WALLTIME: The batch job exceeded its requested wallclock time limit.
- WIDTH: The width parameter for aprun exceeds the mppwidth request.
- NODEFAIL: The application aborted due to a node failure.
- UNEXBUFFER: The application requires a larger MPICH_UNEXBUFFERSIZE.
- ENOENT: The aprun command could not locate the application to launch.
- LIBSMA: Shared memory library error.
- SIGTERM: The batch job was killed.
- NOTRACE: The processing of accounting data could not match an aprun command to the batch job.
- UNKNOWN: None of the other conditions could be identified.
- NOAPRUN: The batch did not execute aprun.
- ATOMIC: For a brief time, shmem atomic operations were disabled. This identified applications that killed due to the attempted use of shmem atomic operations.
- QUOTA: The user exceeded their disk quota.

Error Messages

- WALLTIME: “PBS: job killed: walltime”
- WIDTH: “exceeds confirmed width”
- NODEFAIL: “Received node failed or halted event”
- UNEXBUFFER: “MPIDI_PortalsU_Request_PUPE(605):”
- ENOENT: “No such file or directory” and “aprun: file * not found”
- LIBSMA: “LIBSMA ERROR:”
- SIGTERM: “aprun: Sending caught Terminated signal to application”

Root Cause

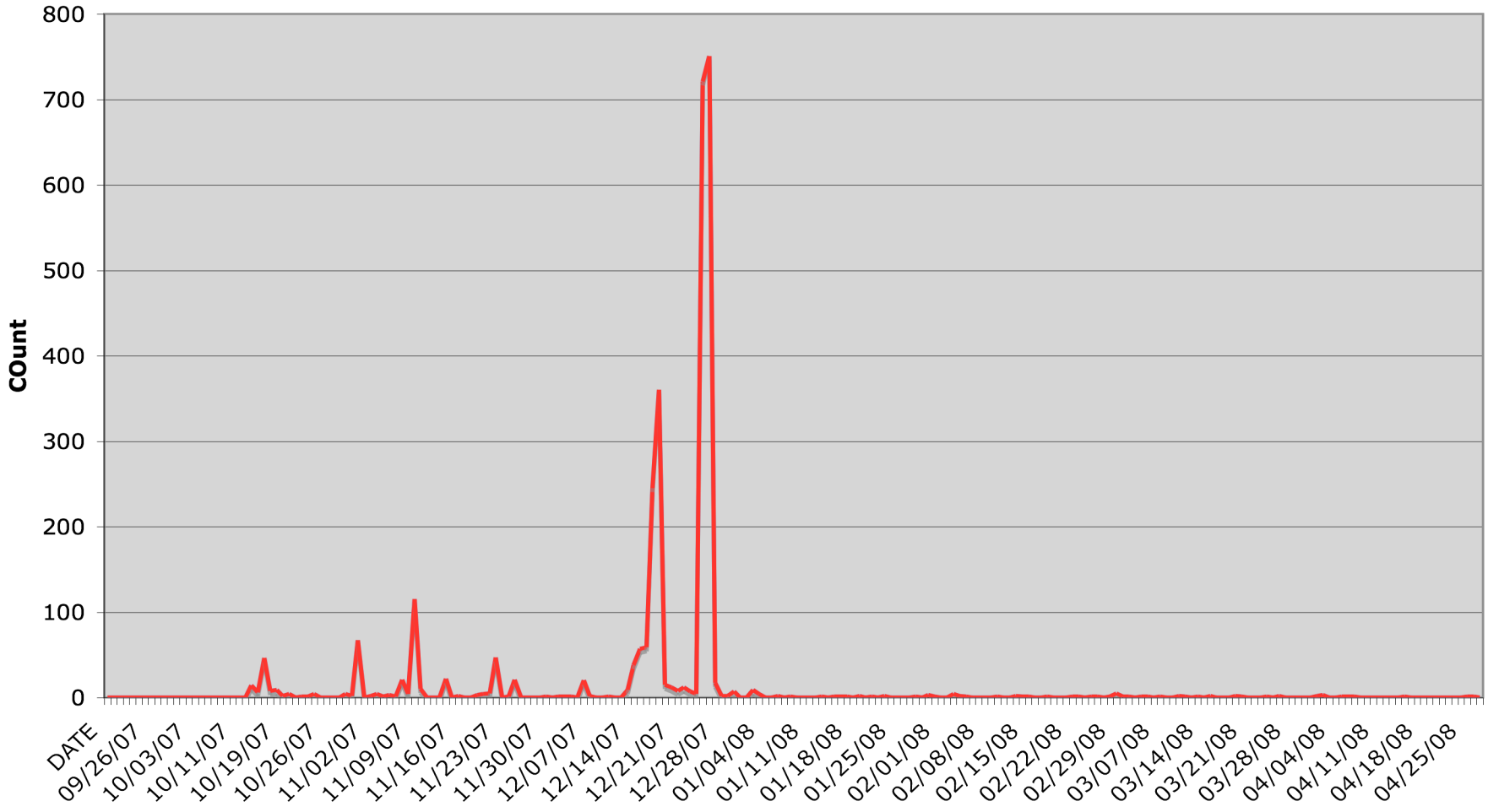
- WALLTIME: User and System error.
- WIDTH: User error.
- NODEFAIL: System error.
- UNEXBUFFER: User error.
- ENOENT: User error.
- LIBSMA: System error.
- SIGTERM: Possible system.
- NOTRACE: Unknown root cause.
- UNKNOWN: Unknown root cause.
- NOAPRUN: User error.
- ATOMIC: System error.
- QUOTA: Currently system error.

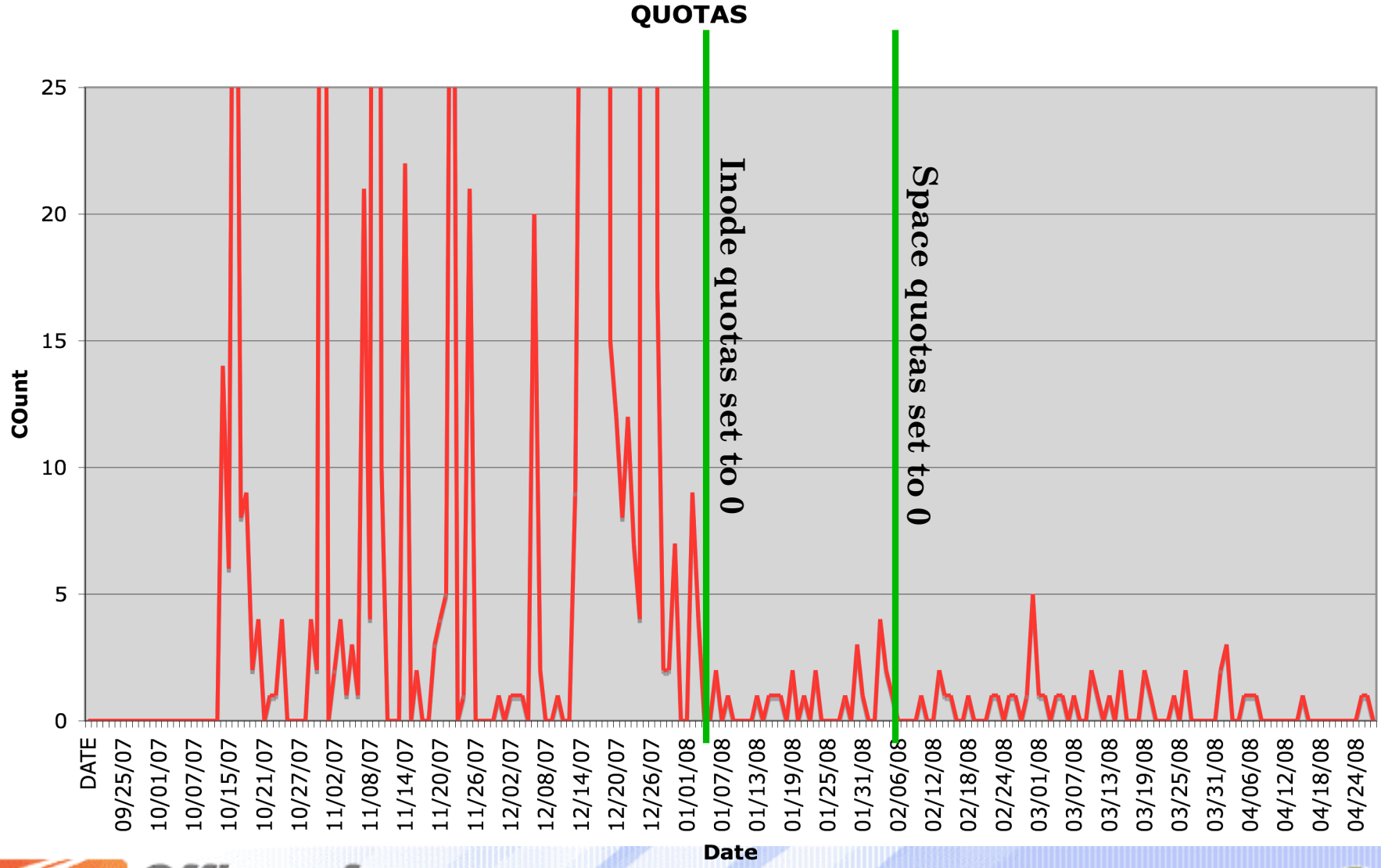
```
#   Epilog Arguments:
#   $1  Job Id
#   $2  User ID
#   $3  Job Name
#   $4  Session ID
#   $5  Resource List
#   $6  Resources Used
#   $7  Queue Name
#   $8  Account String
#
job_id=`echo $1 | /usr/bin/cut -f 1 -d \.`
rc=0
if [ -x /usr/common/nsg/sbin/apinfo ]
then
       /usr/common/nsg/sbin/apinfo -u $2 -s $5 -j $job_id -z
       rc=$?
fi
```

Analysis Considerations

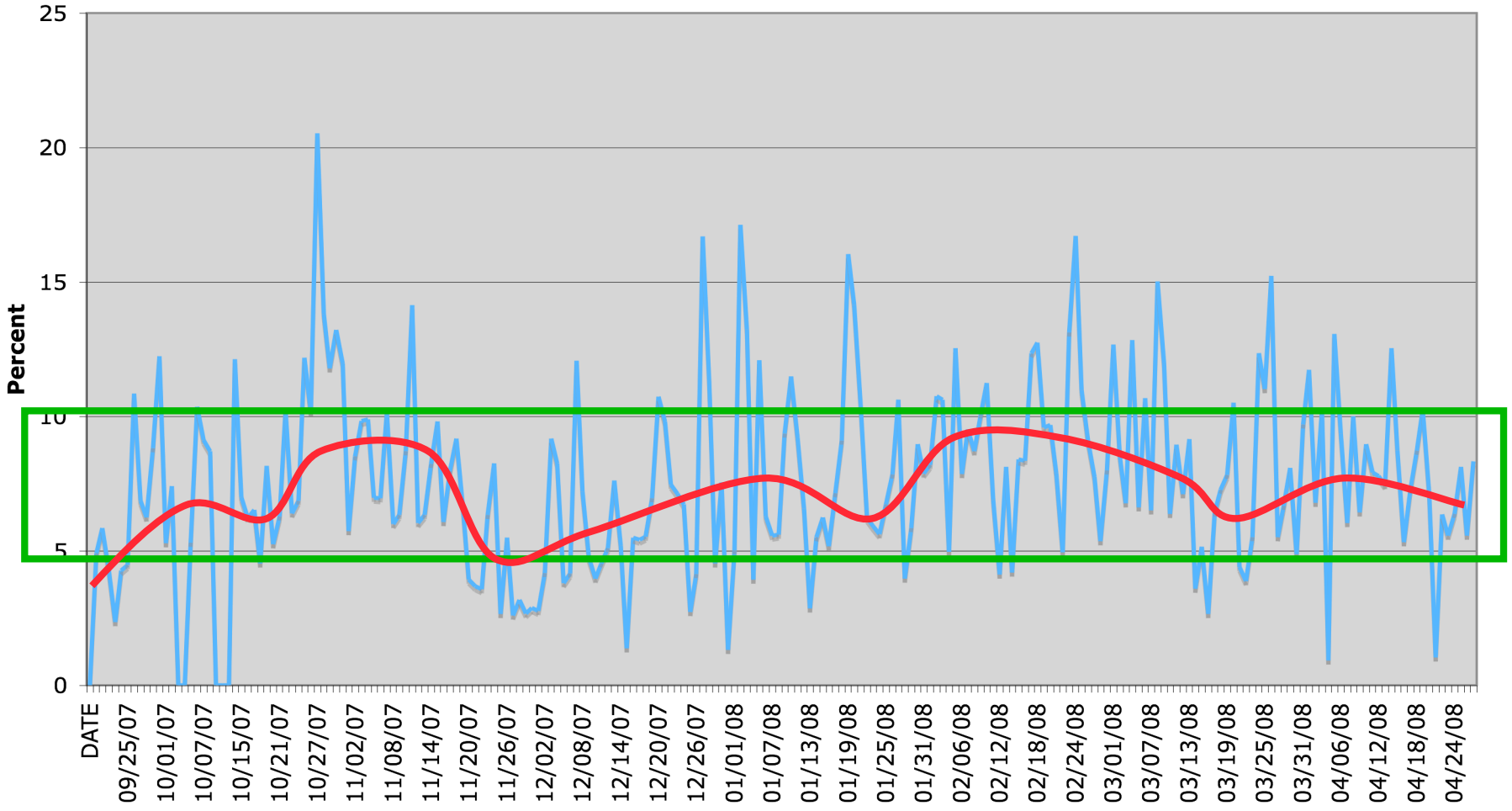
- What's in a number… percentage or count
- What else is going on…
- Did something change…
- Don’t forget the successful jobs!

QUOTAS

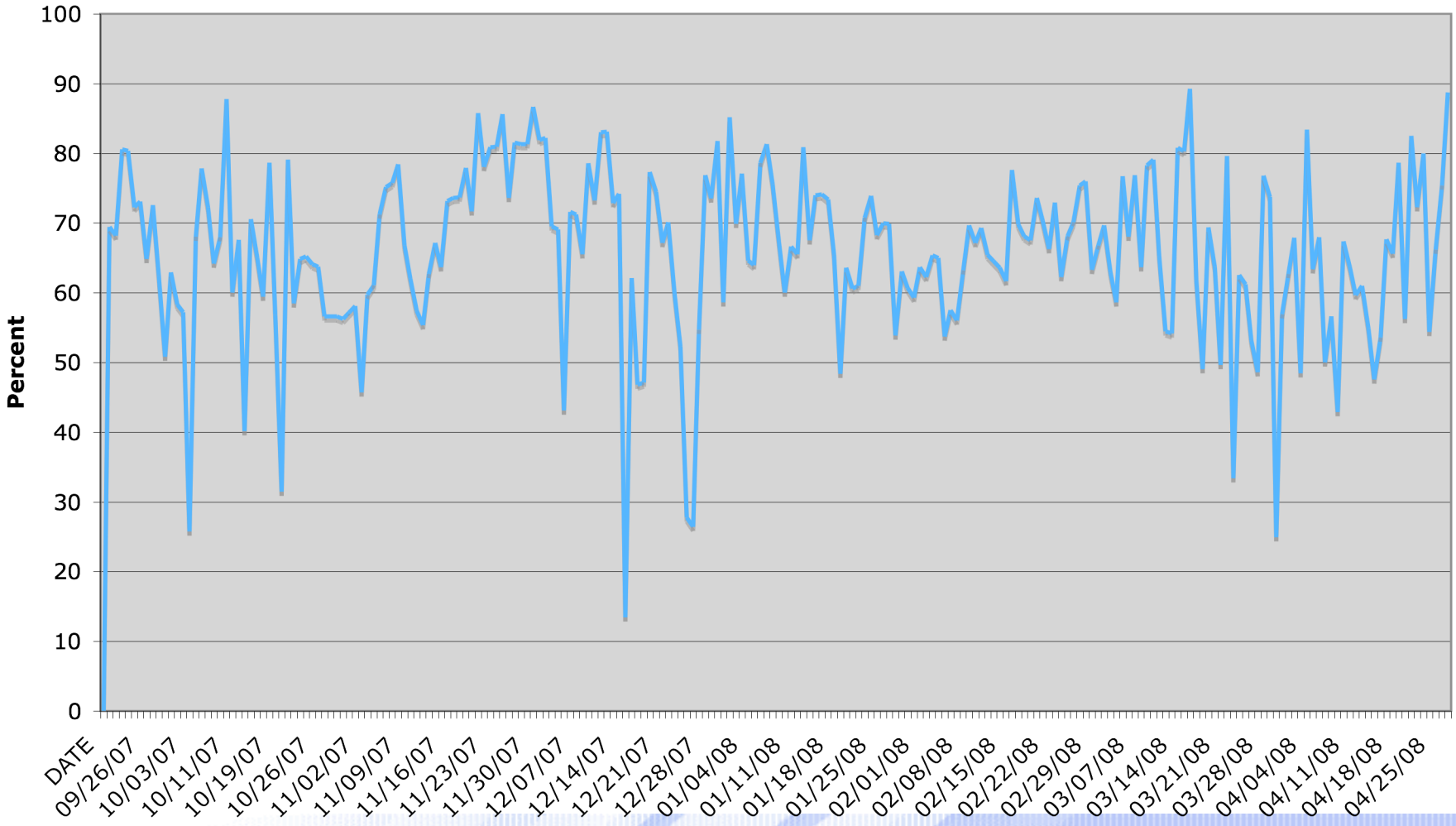

800
700
600
500
400
300
200
100
0
COunt
DATE
09/26/07
10/03/07
10/11/07
10/19/07
10/26/07
11/02/07
11/09/07
11/16/07
11/23/07
11/30/07
12/07/07
12/14/07
12/21/07
12/28/07
01/04/08
01/11/08
01/18/08
01/25/08
02/01/08
02/08/08
02/15/08
02/22/08
02/29/08
03/07/08
03/14/08
03/21/08
03/28/08
04/04/08
04/11/08
04/18/08
04/25/08
Date


QUOTAS
Inode quotas set to 0
Space quotas set to 0
COunt
0
5
10
15
20
25
DATE
09/25/07
10/01/07
10/07/07
10/15/07
10/21/07
10/27/07
11/02/07
11/08/07
11/14/07
11/20/07
11/26/07
12/02/07
12/08/07
12/14/07
12/20/07
12/26/07
01/01/08
01/07/08
01/13/08
01/19/08
01/25/08
01/31/08
02/06/08
02/12/08
02/18/08
02/24/08
03/01/08
03/07/08
03/13/08
03/19/08
03/25/08
03/31/08
04/06/08
04/12/08
04/18/08
04/24/08
Date

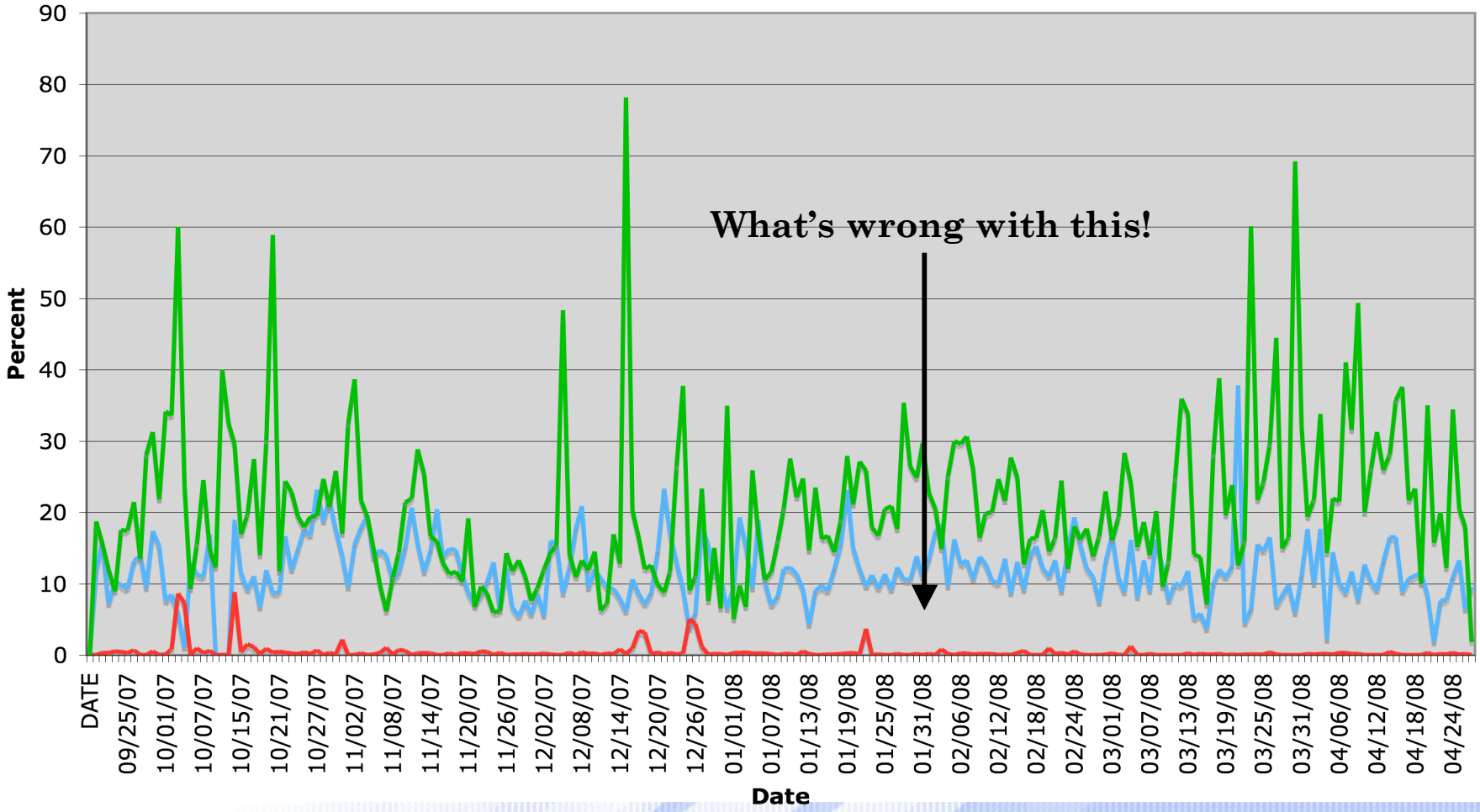
WALLTIME


25
20
15
10
5
0
Percent
DATE
09/25/07
10/01/07
10/07/07
10/15/07
10/21/07
10/27/07
11/02/07
11/08/07
11/14/07
11/20/07
11/26/07
12/02/07
12/08/07
12/14/07
12/20/07
12/26/07
01/01/08
01/07/08
01/13/08
01/19/08
01/25/08
01/31/08
02/06/08
02/12/08
02/18/08
02/24/08
03/01/08
03/07/08
03/13/08
03/19/08
03/25/08
03/31/08
04/06/08
04/12/08
04/18/08
04/24/08
Date

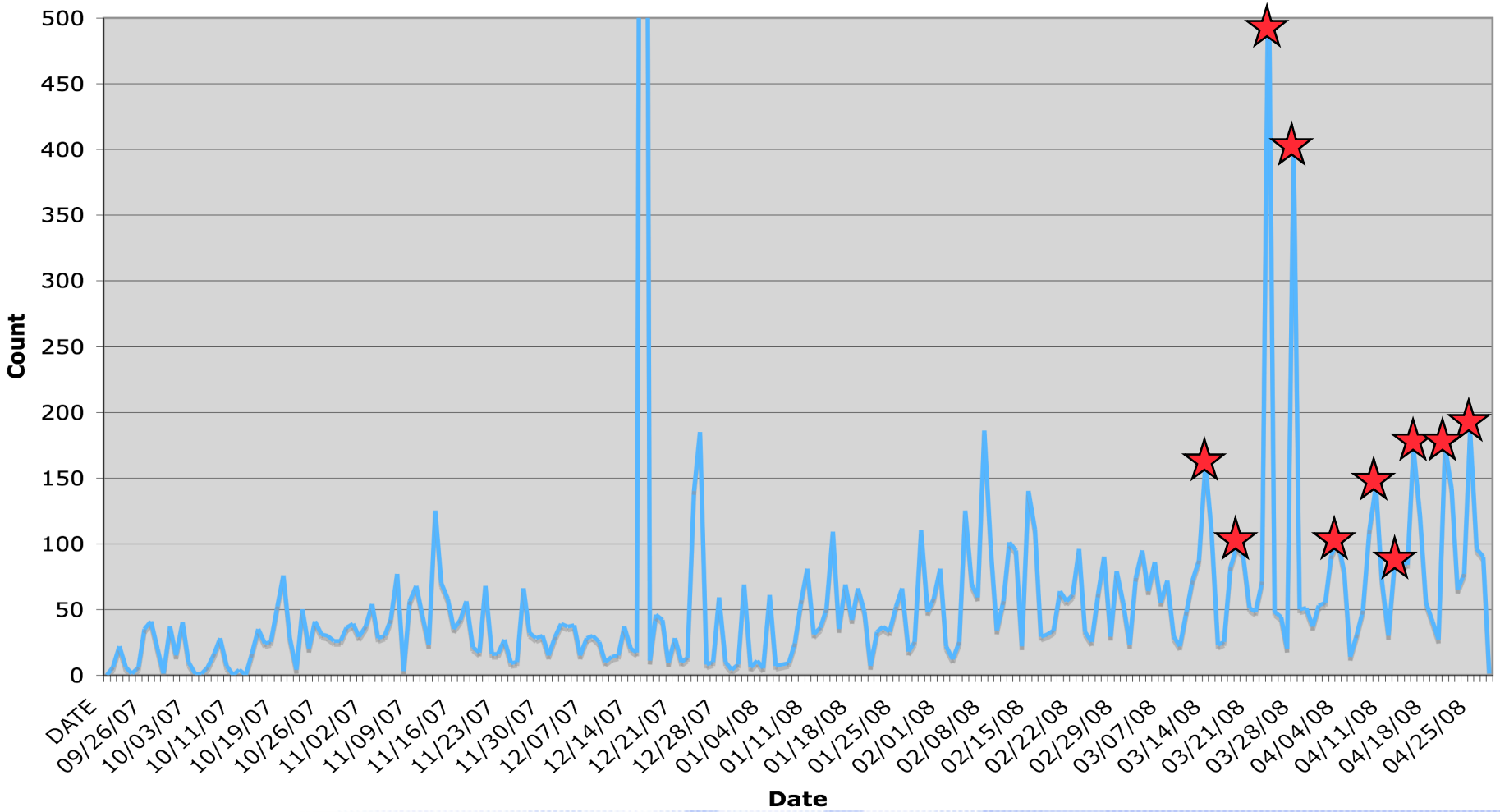
SUCCESS


Percent
100
90
80
70
60
50
40
30
20
10
0
DATE
09/26/07
10/03/07
10/11/07
10/19/07
10/26/07
11/02/07
11/09/07
11/16/07
11/23/07
11/30/07
12/07/07
12/14/07
12/21/07
12/28/07
01/04/08
01/11/08
01/18/08
01/25/08
02/01/08
02/08/08
02/15/08
02/22/08
02/29/08
03/07/08
03/14/08
03/21/08
03/28/08
04/04/08
04/11/08
04/18/08
04/25/08
Date

Root Cause


What's wrong with this!
Percent
90
80
70
60
50
40
30
20
10
0
DATE
09/25/07
10/01/07
10/07/07
10/15/07
10/21/07
10/27/07
11/02/07
11/08/07
11/14/07
11/20/07
11/26/07
12/02/07
12/08/07
12/14/07
12/20/07
12/26/07
01/01/08
01/07/08
01/13/08
01/19/08
01/25/08
01/31/08
02/06/08
02/12/08
02/18/08
02/24/08
03/01/08
03/07/08
03/13/08
03/19/08
03/25/08
03/31/08
04/06/08
04/12/08
04/18/08
04/24/08
Date
User
System
Unknown

NOTRACE


Count
500
450
400
350
300
250
200
150
100
50
0
DATE
09/26/07
10/03/07
10/11/07
10/19/07
10/26/07
11/02/07
11/09/07
11/16/07
11/23/07
11/30/07
12/07/07
12/14/07
12/21/07
12/28/07
01/04/08
01/11/08
01/18/08
01/25/08
02/01/08
02/08/08
02/15/08
02/22/08
02/29/08
03/07/08
03/14/08
03/21/08
03/28/08
04/04/08
04/11/08
04/18/08
04/25/08
Date

From: 04/26/08 00:07:21
to: 04/26/08 23:50:19

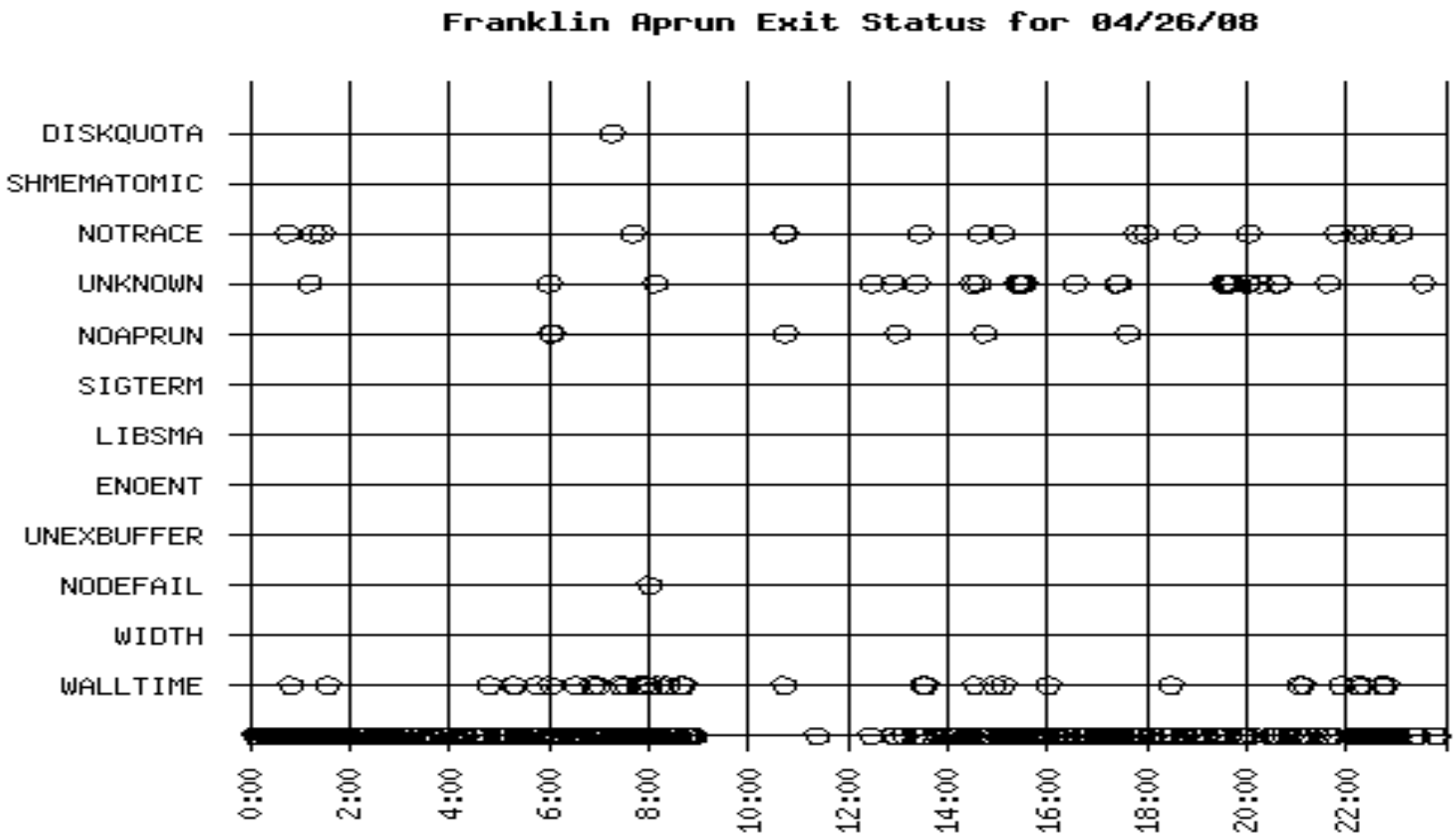
Exit Status	Count
APINFO_SUCCESS	555
APINFO_TORQUEWALLTIME	41
APINFO_APRUNWIDTH	0
APINFO_NODEFAIL	1
APINFO_MPICHUNEXBUFFERSIZE	0
APINFO_ENOENT	0
APINFO_LIBSMA	0
APINFO_SIGTERM	0
APINFO_NOAPRUN	6
APINFO_UNKNOWN	42
APINFO_NOTRACE	90
APINFO_SHMEMATOMIC	0
APINFO_DISKQUOTA	1

Top Ten Failed Users Report

Count	Username
21	user1
18	user2
12	user3
12	user4
7	user5
7	user6
6	user8
5	user9
5	user10
5	user11

Top user in each failed category

Exit Code	CNT	Username
APINFO_TORQUEWALLTIME	5	usera
APINFO_NODEFAIL	1	userb
APINFO_NOAPRUN	2	userc
APINFO_UNKNOWN	8	userd
APINFO_NOTRACE	13	usere
APINFO_DISKQUOTA	1	userf


Franklin Aprun Exit Status for 04/26/08
DISKQUOTA
SHMEMATOMIC
NOTRACE
UNKNOWN
NOAPRUN
SIGTERM
LIBSMA
ENOENT
UNEXBUFFER
NODEFAIL
WIDTH
WALLTIME
0:00
2:00
4:00
6:00
8:00
10:00
12:00
14:00
16:00
18:00
20:00
22:00

Summary

- Failed apruns can be detected
- 100% certainty is not there
- Must use trends
- Must use all other knowledge
- Must collect LOTS of data
- Hard to define expected behavior
- Some errors not detectable

Questions?

Cray Can HELP!

- Improve aprun
- Carefully detect and pass back errors
- Need meaningful error messages