Real Time Health Monitoring of the Cray XT3/XT4 Series Using the Simple Event Correlator (SEC)

Jeff Becklehimer, Cathy Willis,Don Maxwell, Josh Lothian, David Vasil

Jaguar

- 124 cabinets; 56 XT3 and 68 XT4
- 11,708 nodes; 11,508 compute
- 2.6 GHz dual-core AMD opteron
- 4 GB memory per node
- Full 3D torus – 31 x 16 x 24
- It’s kinda big

- System has many logfiles
  - Console – console text from all nodes
  - Consumer – Cray RAS and Management System (CRMS) events i.e. node health
  - Netwatch – errors associated with high speed network (HSN)
  - Combined syslog

- Logfiles are ASCII
- Console, consumer and netwatch don’t rotate
- Files grow rapidly due to size of system (~850 MB/day)
  - Console – ~250 MB/day
  - Netwatch - ~500 MB/day
  - Consumer - ~500 KB/day
  - Syslog – ~100 MB/day

- The XT troubleshooting paradigm
  - Wait for users to complain
  - Dump system
  - Analyze dump
- This grew old very quickly
- Prone to human error

- We desired a way to watch the system as events happen and to notify us immediately upon problems.

- Currently two ways to approach this
  - Write custom scripts
  - Leverage open source

- Only need a temporary solution until Mazama arrives. Under development proposed release Unicos/LC 2.1

Desired Features

- Filter in vs. filter out
- Temporal relationships
- Event threading
- Single vs Multiple Line events
- Maintained by any of the admins
- Interface into existing site infrastructure (i.e. nagios)

SEC – Simple Event Correlator

- Open source and platform independent
- Accepts input from files, pipes, stdin
- Configuration stored in text files as rules
- Rules specify
  - Event matching condition
  - Action list
  - Optionally a boolean expression whose values decides whether to apply rule
- Event matching conditions can be defined with regular expressions, perl scripts, sub-strings, boolean expression
- Produce output events by executing scripts (snmptrap, mail) or by writing to pipes/files.

SEC Rules

- 9 rule types
- These can be broken into two groups
  - Basic – perform actions and do not start an active correlation operation that persists in time
  - Complex – start a multi-part operation that exists for some time after the initial event.

SEC – basic rule types

- Suppress – suppress matching input event. This stops the event from being matched by later rules

- Single – match input event and immediately execute an action that is specified by the rule

- Calendar – execute an action at a specific times using a cron like syntax

SEC – complex rule types

- SingleWithScript – match input event and depending on the exit value of an external script, execute an action
- SingleWithSuppress – match input event and execute an action immediately, but ignore following matching events for the next T seconds
- Pair – match input event, execute the first action immediately, and ignore following matching events until some other input event arrives (within an optional time window). On arrival of the second event execute the second action

SEC – complex rule types con’t

- PairWithWindow – match input event and wait for T seconds for another input event to arrive. If that event is not observed within a given time window, execute the first action. If the event arrives in time, execute the second action.
- SingleWithThreshold – count matching input events during T seconds and if given threshold is exceeded, execute an action and ignore all matching events during the rest of the time window.
- SingleWith2Thresholds – count matching input events during T1 seconds and if a given threshold is exceeded, execute and action. Now start to count matching events again and if their number per T2 seconds drops below the second threshold, execute another action

Sample SingleWithSuppress

```
#
# Detect nodes that panic
#
type= SingleWithSuppress
ptype= RegExp
pattern= \[([0-9A-z._-]+) (\d\d:\d\d:\d\d)\]\[([0-9A-z._-]+)\]0- PANIC_SP
desc= $1 $2 Node $3 Paniced
action=  add PANICED_$3  $1 $2 Node $3 Paniced; \
         report PANICED_$3 /bin/mail -s "Hood Node $3 Paniced"
xxxx@yyyy.gov
window= 120P
```

Sample output

```
Date: Wed, 08 Nov 2006 22:46:13 -0500
To: xxxx@yyyy.gov
Subject: Jaguar Node c12-0c0s1n3 Paniced
User-Agent: nail 10.6 11/15/03
From: crayadm@yyyy.gov

2006-11-08 22:46:13 Node c12-0c0s1n3 Paniced
```

SEC Actions

- Creating, deleting and performing other operations on contexts
- External programs
- Perl mini-programs
- Setting/using variables
- Create new events

XTSec Event List

- Link Inactive
- RX message header CRC error
- RX message CRC error
- Send Buffer Overrun
- RECV Sequence Error

- MCA Error – Machine Check Architecture

- QK Panic

XTSec Event List con't

- Node Corefail
- Node Thermtrip
- VDDIO Fail
- Verty Fail
- Voltage Faults
- Seastar Heartbeat Faults
- Node Heartbeat Faults
- SSNAL Looping too Long
- SCSI Errors – requires running sec on syslog node
- System shutdown

XTSec Event Scripts

- Decode MCA – works with MCA rule to decode MCA errors and only report uncorrectable errors
- Get Node HDT – read node voltages after a node HB fault
- Get Node SS – gather fwstat, fwlog and ptltrace after SS HB fault

SingleWithScript

```
#
# BEGIN Rules
#
#[2006-10-02 09:53:12][c6-0c0s1n1]0- MCA error in bank 4 status 0xd417c001b7080813
#[2007-02-14 15:07:39][c6-2c2s7n0]1- MCA error in bank 2 status 0xf60020000000017a
#
# Detect and decode MCA errors
#
type= SingleWithScript
ptype= RegExp
pattern= \[([0-9A-z._-]+)\s+(\d\d:\d\d:\d\d)\]\[([0-9A-z._-]+)\][0-9]+-\s+MCA error in bank ([0-9]+) status
([0-9A-z._-]+)
script=/home/crayadm/sec-rules/decode_mca.sh $4 $5
desc= $1 $2 Node $3 uncorrectable MCA error Bank $4 Status $5
action=  add MCA_ERROR  $1 $2 Uncorrectable MCA Error $3 Bank $4 Status $5; \
        report MCA_ERROR /bin/mail -s “Jaguar UC MCA Error $3 Bank $4" xxxx@yyyy; \
        delete MCA_ERROR
window= 120
```

SingleWithScript con’t

```
!/bin/bash
lengthstatus=${#2}
if [ "$lengthstatus" -eq 18 ]; then
  if [ "$1" -eq 4 ]; then
    /opt/cray/sys/diag/bin/xtopteronmca -t M2 $1 $2 | grep -i uncorrectable
    exit $?
  else
    /opt/cray/sys/diag/bin/xtopteronmca $1 $2 | grep "Error NOT Corrected by HW"
    exit $?
  fi
else
    exit 1
fi
```

SingeWithScript Sample Output

```
Date: Wed, 08 Nov 2006 22:46:13 -0500
To: xxx@yyyy
Subject: Jaguar UC Memory Error c12-0c0s1n3 Bank 4
User-Agent: nail 10.6 11/15/03
From: crayadm@yyyy

2006-11-08 22:46:13 Uncorrectable MCA Error c12-0c0s1n3 Bank 4 Status
    0xf60ae000000000136
```

Utility Scripts

- Startup – can be started from xtbootsys automation file
- Shutdown – can be invoked by xtshutdown
- Restart – run manually to add new rules while active

Conclusion

- SEC has proven itself to be an ideal framework for implementing a real time monitoring system
- Rules are simple to configure
- In most cases the system administrators are aware of errors before the users