Feasibility Study and Early Experimental Results Towards Cluster Survivability

Box Leangsuksun
Anand Tikotekar
Makan Pourzandi
Stephan Scott
Ibrahim Haddad

Ericsson
ORNL
OSDL

Louisiana Tech
University

Outline

- Motivation
- Current Scenarios
- An Approach: A proof-of-concept to Survivability - HA OSCAR + DSI
- Feasibility and Suitability of the Approach
- Future Work
- Summary
- References
Service Unavailability Impacts

- Losses of $195K - $58M with 3.5 hrs (Meta Group report, 2000)
- Enterprise/Shared Major computing resources - 7/24/365
- Critical HPC apps such as National Security (Home Land defense)
- Service provider Regulation/Mandate
  - FCC mandate (Class 5 local switch = 5 9’s)
- April 2004 Security Breaches on TeraGrid
- Outages -> No Performance and No Functionality
- Losses time and opportunities
- Life-threatening

Motivation

- Unplanned/Planned downtime in systems translates into huge loss due to security attacks/hardware failures/software failures
- Few approaches towards coherent and robust security in clusters/distributed systems
- Realizing 100% security is a myth and thus need for Effective recovery/Serviceability/Fault tolerance
- Above requirements are tantamount to an approach that leads to overall system Survivability, Dependability and Availability
Current Scenario

- Security in clusters is largely based on securing individual nodes
  - Too weak for hackers and too strong for beginners
- Interoperability issues and maintainability issues.
- Cluster security is not tantamount to securing individual nodes, and differs from security on a single machine
- Existing Cluster management systems show minimal concern towards all important RASS
- Many HA systems such as Linux Failsafe, Kimberlite, HP Serviceguard, HA-OSCAR, lack a coherent security framework essential for system survivability

What we mean by RASS

- Reliability (MTTF): *How fast it fails?*
- Availability: The probability that a system is up and kicking at any given point \(\rightarrow MTTF / (MTTF + MTTR)\)
- Serviceability: The ease with which Preventive and corrective maintenance can be performed on the system
- Security: State of being free from danger or injury
RASS goal or Ineluctability of Failure

Reliability - Availability - Serviceability - Security

- A Chain is only as strong as its weakest link
- Miss one and you end up missing four
- No conflict of interest between parameters
- RASS encapsulates Survivability
- “RAS” parameters are provided through HA OSCAR (discussed next) and “S” is provided through DSI.

HA-OSCAR (RAS) 1/3

- Open source clustering software toward Non-Stop services in HEC environment
- Combination of HA features and HPC capabilities
- First Field Grade HA Beowulf cluster combining High Availability and Critical self-healing services
Component Redundancy: High Availability
Self-Configuration: Cluster Management
Adaptive Self Healing: Serviceability
Service Level Fault Tolerance: Survivability
Disaster Recovery: Serviceability
Distributed Security Infrastructure (DSI)

- Developed for Cluster Environment
  - Fine grained, Flexible, Adaptable
- High level of abstraction for access control
  - Separating administrative, network, computation into different security zones
- Process level + User level
  - Kernel level module (DSM)
  - Real time checks based on the LSM hooks

Distributed Security Infrastructure (DSI) Characteristics

- Coherent framework
  - Single server vs. cluster security are different animals
- Process level approach and more
  - Instead of all or nothing
- Pre-emptive security
- Transparent key management
- Dynamic security policy
DSI – components (1/2)

- Security Server
  - Central point of management, policy holder
- Security Manager
  - Node based enforcement of policy
- Secure Communication Channel
  - Encrypted, authenticated. SS <-> SM
- Distributed Security Policy (DSP)
  - XML, rules spanning entire cluster

DSI – components (2/2)

- Robust Security
  - Enforce rules at kernel level
- Distributed Security Module (DSM)
  - Set of kernel hooks implementing DSP via LSM
- Linux Security Module (LSM)
  - Enforces as part of DSI access control service
  - Integrated with SCC
DSI Security Policy (DSP)

- Coherent security vision for entire cluster
- Maintained, Updated by Security Server
- Based on Domain Enforcement
- Expressed In XML
- Separation of Policy decision logic & Policy enforcement logic (Flask architecture)

DSP Architecture

[Diagram showing the DSP Architecture]
Sample DSP Rule

- `<class_PROCESS_Rule>`
  - `<ScID>1</ScID>` // a process with ScID 1
  - `<SnID>2</SnID>` // on node SnID 2
  - `<allow>CREATE</allow>` // can create a
  - `</class_PROCESS_Rule>` // process/call fork()
Cluster Survivability Architecture

HA-OSCAR: RASS framework

Key Approach to our RASS framework

- Enforceable Kernel Security through DSM/DSP (DSI)
- Distributed Security Layer through DSI
- Serviceability layer to check important services including security services of DSI through HA-OSCAR
- Failover or Availability layer – HA-OSCAR monitoring and self-healing core
- Reliability/Dependability/Survivability proposed through the layered architecture
Reality Checks

- But How much improvement?
  - The total uptime?
  - Performance?

- Analytical model and prediction
  - How many 9’s? (downtime per/year)
  - Stochastic Reward Net using SPNP package from Duke U.

Availability Improvement

HA-OSCAR (A) = 99.993 (36 min)
vs.
Beowulf (A) = 99.65 (30 hr)
Feasibility study - Overhead analysis

- DSI 0.3 and HA-OSCAR 1.0 beta on Linux cluster, Redhat 9.0, P4 2.4Ghz
- Overhead in Package According to polling Interval
- CPU usage between 0.9% to 1% by HA OSCAR Mon daemons

More Benchmarking

<table>
<thead>
<tr>
<th>Test type</th>
<th>Without DSI</th>
<th>With DSI</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stat</td>
<td>1.92</td>
<td>1.94</td>
<td>1.0%</td>
</tr>
<tr>
<td>Open/Close</td>
<td>2.68</td>
<td>2.68</td>
<td>0%</td>
</tr>
<tr>
<td>Fork</td>
<td>92.81</td>
<td>93.58</td>
<td>0.82%</td>
</tr>
<tr>
<td>Exec</td>
<td>322.56</td>
<td>328.33</td>
<td>1.78%</td>
</tr>
<tr>
<td>Sh proc</td>
<td>2140.75</td>
<td>2150</td>
<td>0.43%</td>
</tr>
<tr>
<td>UDP</td>
<td>9.68</td>
<td>10.61</td>
<td>9.6%</td>
</tr>
<tr>
<td>RPC/UDP</td>
<td>17.66</td>
<td>18.7</td>
<td>5.9%</td>
</tr>
<tr>
<td>TCP</td>
<td>11.08</td>
<td>12.68</td>
<td>14.4%</td>
</tr>
<tr>
<td>RPC/TCP</td>
<td>23.42</td>
<td>24.3</td>
<td>3.75%</td>
</tr>
</tbody>
</table>
### Issues
- Few issues from OmniORB (htpds as future alternative)
- Distros conflicts (various gcc issues from OSCAR vs DSI 0.3) were overcome.
- DSM using LSM has an equivalent functionality of SElinux, NOT Good.

### Future Work
- DSP Enhancement/Ease of Expression
- Porting to SELinux Functionality instead of DSM
- Localization of Damages
- Intelligent Recovery Policy using event analysis
- More Application opportunities for our RASS framework
Summary

- Existing security solutions adhere to “Individually Repealing attacks”
- Demonstrated RASS based Survivability
- Minimal performance overhead, thus keeping alive HPC proposition
- Cluster RASS can be achieved with acceptable overheads
- Satisfying goals of Mission Critical Systems

Related Links

How do we plan on doing it?

- SELinux
  - Accepted into mainstream kernel
  - Default on the 2.6.xx series
  - However, SELinux is for end systems

- **GOAL**: Combine the granularity of SELinux and distributive characteristic DSI into a cluster security solution
SPN Model

- P Server up
- P Server down
- Failover
- P server repair
- Failback

- S is up and ready
- S takes control
- S Server down
- S repair