A new Distributed Security Model for Linux Clusters

Makan.Pourzandi@Ericsson.Com

Open Systems Lab
Montréal – Canada

June, 2004
Outline

• Context

• Distributed Security

• Distributed Access Control (DisAC)

• DisAC implementation: Distributed Security Infrastructure
Context

- Target application:
  - Large distributed applications
  - Large software base
  - Up 24 hours/7 days
  - High Availability: from 2 to 5 nines (99.999%) uptime
- Linux Clustered servers
- Providing services to different operators
- Exposed to the Public network/Internet
- Running untrusted third-party software
  - No access to the source code
  - No possibility to audit the code for economical and schedule related reasons
- Software configuration evolves slowly over time: no wild software installations
Specific Security Needs

• Security isolation, compartmentalization
  – Large applications supporting many functionalities
  – Probability of exploitable vulnerabilities in a large software base
  – A vulnerability in some parts of the system can compromise the entire system

• Pre-emptive security: changes in the security context will be reflected immediately

• Dynamic security policy: run time changes in security context and policy

• Don't only rely on application layer security mechanisms:
  “Administrators must contend with vulnerabilities in applications over which they have no direct control”
Distributed Security
Challenges in Distributed security

• Implement coherent distributed security
  – Many layers to **fit together**: Applications, Middleware, OS, Hardware, Network …
  – Heterogeneous environment: variety of Hardware, Software: OS, Middleware, Networking technologis

Security Architectural Elements in ITU-T X.805,
from Security in Telecom and Information Technology, Dec 2003
Challenges in Distributed security (2)

- Integration of different security solutions from potentially different vendors...

- System management
  - If manually managed, it may lead to misconfigurations and inconsistencies
“Distributed Systems Require Distributed Security”

Hartman, Flinn, Beznosov,
Enterprise Security with EJB and CORBA
User based access control approach on cluster computing

• Based on user privileges (login, password)
  – Life time: a session of several hours/days
  – Scope: limited range of operations according to the application’s nature
  – Security policy based upon login and passwords

• Our target application:
  – Few users only
  – Life time: months if not years
  – Scope: wide range of operations, from upgrading software to managing information in database
### Needs vs. Existing mechanisms

<table>
<thead>
<tr>
<th>Current mechanisms</th>
<th>Needs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security policy based upon login and passwords</td>
<td>Security policy depends on the processes and the source of the request rather than the user</td>
</tr>
<tr>
<td>Compartmentalization: necessity of creating/managing many users and groups</td>
<td>Running for a very long time (months) under the same login without rebooting</td>
</tr>
<tr>
<td>User authentication often only at login time</td>
<td>Fine grained security policy based on processes</td>
</tr>
<tr>
<td>Running for short period of time (days) before each reboot</td>
<td>Pre-emptive security</td>
</tr>
<tr>
<td>No pre-emptive security</td>
<td></td>
</tr>
</tbody>
</table>
MAC vs DAC

- Discretionary Access Control (DAC)
  - It is at the discretion of the user to define access control privileges for user resources

- Mandatory Access Control (MAC)
  - Access control no longer solely depends on the user's decision
  - The system administrator defines the access control policy of the system
Existing solutions

• Many existing security solutions exist:
  – As external security mechanisms to the servers such as firewalls and Intrusion Detection Systems
  – As part of servers such as Integrity checks and some mechanisms to enhance security as a part of OS…

• Recent efforts
  – Linux Security Modules
  – SE Linux
  – Process right management in Solaris 10

• However, there are few efforts to make a coherent framework for enhancing security in a distributed system
Distributed Access Control (DisAC)
DisAC

• **Goal**
  – Extending kernel-level Mandatory Access Control features for a single computer into features for a distributed system

• **Usage**
  – Sharing the same cluster among different applications running untrusted third-party software
  – Compartmentalization: Setting up virtual security zones inside the cluster
Characteristics

• Cluster-wide access control

• Process-level granularity access control

• Access control at Operating system kernel level (Linux)

• Extra security functionality in addition to
  – User level authentication
  – Discretionary Access Control
General access model
Process level Security Context

• Process security ID: ScID
  – Privileges associated with each process
    • Defined for the entire cluster: it can be transferred and interpreted through the whole cluster
    • Persistent
  – Security ID: corresponding to the group security context
    • Assigned by local security manager at the creation of the process

• Node security ID: SnID
  – Privileges associated with each node
  – Unique for each node in the cluster

• Security Context for each process
  – Defined by: <ScID, SnID>
ScID for a newly created process

- ScID of binary:
  - Stored in ELF header
  - Integrity protected by digital signatures
- Parent Process ScID
- Node SnID
- New ScID stored at kernel level for each process
Classifying binaries based on security

- Using ScID(s) for categorizing binaries

```
$ /bin/emacs
$ /bin/sh
ScID=2
```

Does `/bin/emacs` have an ScID affected to its binary?

- Yes
  - Is transition from ScID=2 to ScID=43 allowed?
    - Yes
      - `/bin/emacs` with new ScID
    - No
      - Don’t launch emacs

- No
  - Is transition from ScID=2 to Default ScID allowed?
    - Yes
      - `/bin/emacs` with new ScID
    - No
      - Don’t launch emacs
Access control decisions

- Process level based on \(<\text{SnID}, \text{ScID}>\) of source and target
- Locality in the distributed system is taken into account
- An access is only granted if the following access policy exists:
  - Source \(<\text{SnID}, \text{ScID}>\), Target \(<\text{SnID}, \text{ScID}>\), Allowed Action
- Different classes of rules: network, socket, process, transition
Distributed Security Policy (DSP)

• Express a coherent security vision (security policy) throughout all the cluster

• Local security policy:
  – Initially integrated to the secure boot software
  – Maintained and updated by the security server through security communication channel
Node 1 (SnID=1)  
Source Process (ScID=10)  

Node 2 (SnID=2)  
Target Process (ScID=20)  

Distributed Security Policy  

<allow> Connect </allow>
DisAC: Access control verifications

**Source Node**

- **Proc12**
  - **ScID=10**
  - **Main()**
    ```
    ...
    send(sock1,...);
    ...
    ```

**Target Node**

- **Proc14**
  - **ScID=10**
  - **Main()**
    ```
    ...
    receive(sock1,..);
    ...
    ```

**DSP**

- **DSM**

**User Level**

**Kernel Level**

**IP Packet**

- **SScID, SsnID**

**TCP Port x**

- **ScID=10**

**TCP Port 8000**

- **ScID=10**
Node 1 (SnID=1)

Process::Send

1 permission to send msg

Port xxx
Client Socket

Node 2 (SnID=2)

Process:Receive

5 permission to receive msg

Port 8000
Server Socket

Permission to send msg:

1. Process::Send
2. Permission to send IP packets

Permission to receive msg:

5. Process:Receive

Permission for the socket to receive from node 2, ScID = 10:

4. Permission for the socket to receive from node 2, ScID = 10

Explicit assignment ScID=10

Implicit assignment ScID=10

Permission to receive IP packets:

3. Permission to receive IP packets
Cluster-wide Access Control

Node 1 | Node 2 | Node N

Process

Security Zones

Resource

Access
Security zones

- Each ScID corresponds to a security zone (few ScIDs)

- ScIDs are not meant to be unique for each process or application

- Possible simplification by using ALL keyword in DSP for SnID allows to avoid the locality in the cluster

- Very simple; using two ScIDs and SnID set to ALL; it is possible to divide the cluster into two zones: trusted and untrusted zone
DisAC: Creating Virtual Security Zones

Inside the cluster

Outside

Virtual sub-cluster

BackEnd server of PhoneMania

SM

SS

BackEnd server of RingBell

FrontEnd server of PhoneMania

FrontEnd server of RingBell

DSM

SS

SM

10.1.1.2

10.1.1.1

8801

9001

9000

8800

TelecomClient

DisAC: Creating Virtual Security Zones
DisAC implementation: Distributed Security Infrastructure
Distributed Security Infrastructure

- **Primary Security Server Node**
  - SS
- **Node 1**
  - Proc123
  - SM
  - DSM
- **Node 2**
  - Proc978
  - SM
  - DSM
- **Node 3**
  - Proc222
  - SM
  - DSM

**Secure Communication Channel**

- **Inside the Cluster**
- **Outside the Cluster**

- **Data Traffic**

- **Security and O&M/IDS**

- DSM
- SS
- SM

**Distributed Security Module**
**Security Server**
**Security Manager**
**Authenticated**
**Encrypted Communications**
DSP Update

Secure Communication Channels

UpdateDSP

load and check

propagate

Node 1

Node 2

security enforcement

Kernel
Development Environment

- Kernel 2.4.17
- LSM patch
- Red Hat 7.3
- C/C++
- GCC 2.96
**Benchmarking results**

<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>

Results for LMbench on Linux 2.4.17, Pentium 4, 2,4 Ghz.
Time units are microseconds.
Challenges

• Comprehensible and acceptable security policy
  – Seltz & Schroder 1975: security mechanisms must be comprehensible and acceptable to users, or they will be ignored and bypassed.
  – Explicitly defining security zones in DSP
    • generate the low level DSP rules from those high level security zones
Conclusions

- Distributed Access Control
  - DisAC enforces cluster-wide access control at kernel level for distributed systems
  - Unified security view
  - Process level Granularity
  - Setting virtual security zones inside a cluster

- Distributed Security Infrastructure
  - Implemented under GPL (stable version for kernel 2.4.17, development version for kernel 2.6.X)
DSI References

- **Web site**
  http://disec.sourceforge.net/
  http://www.linux.ericsson.ca/dsi

- **DSI Packages**
  http://sourceforge.net/project/showfiles.php?group_id=67502

- **Online CVS**
  http://cvs.sourceforge.net/viewcvs.py/disec/

- **Contact person**
  makan.pourzandi@ericsson.com