Exokernel: An Operating System Architecture for Application-Level Resource Management

Authored by: Dawson R. Engler, M. Frans Kaashoek, and James O'Toole Jr.

Presented by: Fery
Overview

- Architecture Comparison
- Issues
- Approaches
- Whole System
- Principles
- Exokernel Tasks
  - Secure Bindings
  - Visible Revocation
  - Abort Protocol
- Measurement
- Performance Comparison
- Conclusions
- History
- Continuation
## Architecture Comparison

<table>
<thead>
<tr>
<th>Traditional OS</th>
<th>Exokernel OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fix interfaces &amp; abstraction</td>
<td>Low level interfaces to application</td>
</tr>
<tr>
<td>Hide hardware resources</td>
<td>Provides application-level management of hardware resources</td>
</tr>
</tbody>
</table>

### Diagram:

```
+---------------------------------+---------------------------------+---------------------------------+---------------------------------+
| Apache                          | SQL Server                      | Apache                          | SQL Server                      |
| Abstractions                    | Interface                        | Abstractions                    | Interface                        |
| Hardware                        |                                 | Exokernel                       |                                 |
```
Issues

- Performance
- Flexibility
- Functionality
Performance

- General-purpose implementations of abstractions
Flexibility

- Abstractions on traditional OS hide the hardware resources
- Difficult to implement custom resource management abstractions
Functionality

- Only one interface available between applications and hardware resources
- Changes of existing abstractions are unlikely
Approaches

- Provides as low-level interface as possible
- Exports resources securely
- Give application as much control as possible the right of resource management
- Special implementation is possible
Whole System

- Exokernel
  - Ex: Aegis
  - Exports hardware resources through a low-level interface

- Library Operating System
  - Ex: ExOS
  - Use the low-level exokernel interface
  - Application can use the library or use its own
Principles

- Securely expose hardware
  - Kernel should provide secure low-level primitives that allow all hardware resources to be accessed as directly as possible.

- Expose allocation
  - Allow to request specific hardware resources

- Expose Names
  - Export physical names.
  - Remove a level of indirection: Translation

- Expose Revocation
  - Utilize a visible resource revocation protocol
Applications use their own Library Operating System (LibOS) which in turn use the exokernel to allocate and deallocate hardware resources.

ExOS library provides a user-level and extensible implementation of an UNIX operating system.

Most UNIX applications like gcc, perl, apache, tcsh, and telnet compile and work without changes using ExOS.
Exokernel Tasks

- Track ownership
- Guard resources usage
- Revoke access

- Secure bindings
  - Visible revocation
  - Abort protocol
Secure Bindings

- Protection mechanism that decouples authorization from the actual use of a resource
- Protection check only at bind time
- Techniques:
  - Hardware mechanisms
  - Software caching
  - Downloading application code
Visible Revocation

- When a bounded resource is requested by another application, kernel lets the application know.
- Application that has the resource has to return the resource within a particular time.
Abort Protocol

- If application doesn’t follow return policy, the resource is taken back by kernel forcefully
Measurement

- Hardware: DECstation 2100 and DECstation 5000
- Software: Ultrix (monolithic kernel) and ExOS (Exokernel)
Performance Comparison

<table>
<thead>
<tr>
<th>Machine</th>
<th>OS</th>
<th>Procedure Call</th>
<th>Syscall (getpid)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC2100</td>
<td>Ultrix</td>
<td>0.57</td>
<td>32.2</td>
</tr>
<tr>
<td>DEC2100</td>
<td>Aegis</td>
<td>0.56</td>
<td>3.2/4.7</td>
</tr>
<tr>
<td>DEC3100</td>
<td>Ultrix</td>
<td>0.42</td>
<td>33.7</td>
</tr>
<tr>
<td>DEC3100</td>
<td>Aegis</td>
<td>0.42</td>
<td>29/35</td>
</tr>
<tr>
<td>DEC5000</td>
<td>Ultrix</td>
<td>0.28</td>
<td>21.3</td>
</tr>
<tr>
<td>DEC5000</td>
<td>Aegis</td>
<td>0.28</td>
<td>1.6/2.3</td>
</tr>
</tbody>
</table>

- Base Cost for null procedure call and a simple System Call
- Times are in microsecond
### Performance Comparison (2)

<table>
<thead>
<tr>
<th>Machine</th>
<th>OS</th>
<th>unalign</th>
<th>overflow</th>
<th>coproc</th>
<th>prot</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC2100</td>
<td>Ultrix</td>
<td>n/a 2.8</td>
<td>208.0 2.8</td>
<td>n/a 2.8</td>
<td>238.0 3.0</td>
</tr>
<tr>
<td>DEC2100</td>
<td>Aegis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEC3100</td>
<td>Ultrix</td>
<td>n/a 2.1</td>
<td>151.0 2.1</td>
<td>n/a 2.1</td>
<td>177.0 2.3</td>
</tr>
<tr>
<td>DEC3100</td>
<td>Aegis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEC5000</td>
<td>Ultrix</td>
<td>n/a 1.5</td>
<td>130.0 1.5</td>
<td>n/a 1.5</td>
<td>154.0 1.5</td>
</tr>
<tr>
<td>DEC5000</td>
<td>Aegis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Time to dispatch an exception in Aegis and Ultrix
- Times are in microsecond
Performance Comparison (3)

- Roundtrip latency of a 60-byte packet
- FRPC: high-performance of RPC for Ultrix
- ASH: untrusted app-level message-handler that are downloaded into the kernel
Conclusions

- Exokernel multiplexes resources among applications securely
- Library operating system implement high-level abstraction that are utilized by applications
- Separation between management and protection
  - Exokernel: protection at low-level
  - Library operating system: management at high-level
History

1994
• Aegis: A secure programmable exokernel
• The Exokernel Approach to Operating System Extensibility

1995
• The design and implementation of a prototype exokernel operating system
• Exterminate all operating system abstractions
• Exokernel: An Operating System Architecture for Application-Level Resource Management

1996
• DPF: Fast, flexible message demultiplexing using dynamic code generation
• Server operating systems
• ASHs: Application-specific handlers for high-performance messaging
• Design and implementation of a modular, flexible, and fast system for dynamic protocol composition

1997
• Application performance and flexibility on exokernel systems
• Decentralizing UNIX Abstractions in the Exokernel Architecture

1998
• The Exokernel operating system architecture

2002
• Exokernel and Exotools
• Fast and flexible application-level networking on exokernel systems
Continuation

- Fast and flexible application-level networking on exokernel systems (published in 2002)
- Cheetah Webserver
Thank you