A Linux in Unikernel Clothing

Hsuan-Chi Kuo\(^+\), Dan Williams\(^*\), Ricardo Koller\(^*\) and Sibin Mohan\(^+\)

\(^+\)University of Illinois at Urbana-Champaign
\(^*\)IBM Research

IBM Research

Lupine
Unikernels are great

BUT: Unikernels lack full Linux Support

- Hermitux: supports only 97 system calls
- OSv:
  - Fork(), execve() are not supported
  - Special files are not supported such as /proc
  - Signal mechanism is not complete
- Rumprun: only 37 curated applications
- Community is too small to keep it rolling

- Heavy
- Inefficient
- Small kernel size
- Fast boot time
- Improved performance
- Better security

Diagram:

- Hypervisor
- Kernel
- LibOS + App
- App
Can Linux behave like a unikernel?
Lupine Linux

All 16000 Linux configurations

- lupine-base 283 (34%)
- 311 (56%)

Application specific and unnecessary options
- 89 (16%)
- 150 (28%)

microvm 833 (5%)

Specialization via Kconfig
System Call Overhead Elimination via KML

Application (container)
Application manifest

App rootfs

Lupine Linux "Unikernel"
Lupine Linux

- **Kernel mode Linux (KML)**
  - Enables normal user process to run in kernel mode
  - Processes can still use system services such as paging and scheduling
  - App calls kernel routines directly without privilege transition costs

- **Minimal patch to libc**
  - Replace syscall instruction to call
  - The address of the called function is exported by the patched KML kernel using the vsyscall
  - No application changes/recompilation required
Evaluation Metrics

Based on: **Unikernel benefits**

- Boot time
- Image size
- Memory footprint
- Application performance
- Syscall overhead
Configuration diversity

- 20 top apps on Docker hub (83% of all downloads)
- Only 19 configuration options required to run all 20 applications: **lupine-general**
Evaluation - Comparison configurations

Lupine
[Lupine-base + app-specific options]

general
Kernel for 20 apps

Cloud Operating Systems

Linux-based

Unikernels

OSv

Hermitux

Rumprun

MicroVM
Evaluation - Image size

- Configuration is effective
- 4 MB
- 27% of microvm
- Even lupine-general produces smaller images than Rump, OSv
Evaluation - Boot time

- OSv boot heavily depends on FS choice
- Lupine boot time without KML*
- Even lupine-general boots faster than Hermitux, OSv

*KML incompatibility with CONFIG_PARAVIRT
Evaluation - Memory footprint

- Repeatedly tested app with decreasing memory allotment
- Choice of apps limited by unikernels
- No variation in lupine: lazy loading makes binary size irrelevant
Evaluation - System call overheads

- Lmbench
- 56% improvement over microvm from specialization
- Additional 40% from KML
- KML benefit vanishes quickly in more realistic workloads
Evaluation - Application performance

- Throughput normalized to microVM
- Application choice limited by unikernels
- Lupine outperforms microVM by up to 33%
- Lupine-general does not sacrifice performance
- Linux implementation is highly optimized

<table>
<thead>
<tr>
<th>Name</th>
<th>redis-get</th>
<th>redis-set</th>
<th>nginx-conn</th>
<th>nginx-sess</th>
</tr>
</thead>
<tbody>
<tr>
<td>microVM</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>lupine-general</td>
<td>1.19</td>
<td>1.20</td>
<td>1.29</td>
<td>1.15</td>
</tr>
<tr>
<td>lupine</td>
<td>1.21</td>
<td>1.22</td>
<td>1.33</td>
<td>1.14</td>
</tr>
<tr>
<td>lupine-tiny</td>
<td>1.15</td>
<td>1.16</td>
<td>1.23</td>
<td>1.11</td>
</tr>
<tr>
<td>lupine-nokml</td>
<td>1.20</td>
<td>1.21</td>
<td>1.29</td>
<td>1.16</td>
</tr>
<tr>
<td>lupine-nokml-tiny</td>
<td>1.13</td>
<td>1.13</td>
<td>1.21</td>
<td>1.12</td>
</tr>
<tr>
<td>hermitux</td>
<td>.66</td>
<td>.67</td>
<td>.87</td>
<td>.53</td>
</tr>
<tr>
<td>osv</td>
<td>.99</td>
<td>.99</td>
<td>1.25</td>
<td>.53</td>
</tr>
<tr>
<td>rump</td>
<td>.99</td>
<td>.99</td>
<td>1.25</td>
<td>.53</td>
</tr>
</tbody>
</table>

Table 4. Application performance normalized to MicroVM (Note: higher value is better).
Lupine achieves unikernel benefits

- Image size
  - 4MB image size
- Boot time
  - 23ms boot time
- Application performance
  - Up to 33% higher throughput

Best of all, it is still a Linux.
Takeaways

● **Specialization is important:**
  ○ 73% smaller image size, 59% faster boot time, 28% lower memory footprint and 33% higher throughput than the state-of-the-art VM

● **Specialization per application may not be:**
  ○ 19 options (lupine-general) cover at least 83% of downloaded apps with at most 4% reduction in performance

● **System call overhead elimination may not be:**
  ○ only 4% improvement for macro-benchmark, unlike 40% for microbenchmarks

● **Lupine avoids common pitfalls:** has support for unmodified Linux applications, optimized implementation
Thank you!

Hsuan-Chi Kuo  hckuo2@illinois.edu
Dan Williams  djwilla@us.ibm.com
Ricardo Koller  kollerr@us.ibm.com
Sibin Mohan  sibin@illinois.edu

https://synergys.github.io/projects/lupine