RUMP KERNELS

and

{why,how} we got here

New Directions in Operating Systems
November 2014, London

Antti Kantee, Fixup Software Ltd.
pooka@rumpkernel.org
@anttikantee
Motivations

- want to run an application, not an OS
- want a better operating system
- “operating system gets in the way”
FIRST HALF

what is an operating system
Summary of OS's

- drivers
  - for enabling applications to run
  - $n \times 10^6$ LoC

- optional goop defining relation between drivers and applications
  - for protection, resource sharing, ...
  - $10^3 - 10^5$ LoC
kernel

application

driver

driver

driver

driver

application
SECOND HALF

what is a rump kernel
callers (i.e. “clients”)

rump kernel

file systems  TCP/IP  device drivers  syscalls  ...

hypercall interface

hypercall implementation

platform
rump (n):
small or inferior remnant or offshoot; especially: a group (as a parliament) carrying on in the name of the original body after the departure or expulsion of a large number of its members
rump kernel (n):
small or inferior remnant or offshoot; specifically: a monolithic OS kernel carrying on in the name of the original body after the departure or expulsion of a large number of its subsystems
A rump kernel does not provide threads, a scheduler, exec, or virtual memory, nor does it require privileged mode (or emulation of it) or interrupts

> runs anywhere
> integrates into other systems
Wait, that doesn't explain where the drivers come from

< anykernel (NetBSD)
same thread throughout entire stack

platform

hypercall interface

hypercall implementation

platform

application(s)

userspace libraries

libc

rump kernel calls

syscall traps

rump kernel

file systems

TCP/IP

device devs

syscalls

...
THIRD HALF
(with operating systems, expect the unexpected)

how rump kernels happened
Step 1: RUMP (2007)

- **userspace fs framework** (userspace part)
  - rump kernel
  - VFS emustub
  - *unmodified* file system driver
  - ad-hoc shims
- hypercall implementation
- NetBSD userspace

**application**

- syscalls, VFS, etc.

**userspace fs framework** (kernel part)

- file system driver

**userspace**
Step 2: UKFS (2007)

Q: how hard can implementing a few syscalls be?
A: very

- application (e.g. fs-utils)
- UKFS
- rump kernel
- VFS emustub
  - unmodified file system driver
  - ad-hoc shims
- hypercall implementation
- userspace
  - userspace
  - kernel
Step 3: a lot (2008 - 2011)

- support for all driver subsystems
- isolation from the host
- stable hypercall interface
- anykernel completed
- production quality
- rump kernels used for testing NetBSD
- no libc for rump kernels, applications ran partially on the host
Step 3.5: visions (not an actual step)

c. turn of the year 2011/2012:

“An anykernel architecture can be seen as a gateway from current all-purpose operating systems to more specialized operating systems running on ASICs. The anykernel enables the device manufacturer to provide a compact hypervisor and select only the critical drivers from the original OS for their purposes. The unique advantage is that drivers which have been used and proven in general purpose systems, e.g. the TCP/IP stack, may be included without modification as standalone drivers in embedded products.”
Step 4: portability to POSIX
2007-2012, 2012-
buildrump.sh (2012-)
4.4STEP: beyond POSIX (201[234])
Type help if you feel like you need some help.

Cmd:
Arg1:
Data:

Do It!  Clear

The NetBSD Foundation, Inc. All rights reserved.

The Regents of the University of California.

pooka@T61:/home/pooka/src/bsd/src/sys/rump sys

total memory = unlimited (host limit)
timecounter: Timecounters tick every 10.000 msec
cpu0 at thinair0: rump virtual cpu
root file system type: rumpfs
/dev/fss: hostpath /test.ffs (1408 KB)
Questions about VGER’s services: &lt;postmaster@vger.kernel.org&gt; &lt;br&gt; Postmaster contact addresses as an &lt;A HREF="real-vger-postmasters.gif" image&gt;&lt;br&gt; FAQ answers: &lt;A HREF="http://www.tux.org/1kml/"&gt;http://www.tux.org/1kml/&lt;/A&gt;&lt;br&gt; &lt;br&gt; &lt;A HREF=mailto:honey+1366672895@vger.kernel.org&gt;mail to Honey&lt;/A&gt; &lt;br&gt; mailto: hunaja+1366672895@vger.kernel.org --&gt; Bo and others want &lt;A HREF="bo.html"&gt;email&lt;/A&gt; to teach filters about spam. &lt;/P&gt; &lt;/BODY&gt; &lt;/HTML&gt;
Step 5.1: rumprun (2013, 2014)
Step 5.2: rumprun (2013, 2014)

<table>
<thead>
<tr>
<th>application(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>userspace libraries</td>
</tr>
<tr>
<td>libc</td>
</tr>
<tr>
<td>rump kernel calls</td>
</tr>
<tr>
<td>rump kernel</td>
</tr>
<tr>
<td>file systems</td>
</tr>
<tr>
<td>glue code</td>
</tr>
</tbody>
</table>

**Diagram:**

- Hypercall interface
- Hypercall implementation
- Platform
FINAL HALF

conclusions & other tidbits
All le gory technical details:

http://book.rumpkernel.org/

2\textsuperscript{nd} edition is work in progress
Will be available as free pdf, hopefully printed too
Community

- http://rumpkernel.org/
- http://repo.rumpkernel.org/
  - BSD-licensed source code
- http://wiki.rumpkernel.org/
- rumpkernel-users@lists.sourceforge.net
- #rumpkernel on irc.freenode.net
- @rumpkernel
The actual conclusions
You can make an omelette without breaking the kitchen!