pkgsrcCon 2016

# The Rumprun Unikernel

Sebastian Wicki

gandro@rumpkernel.org

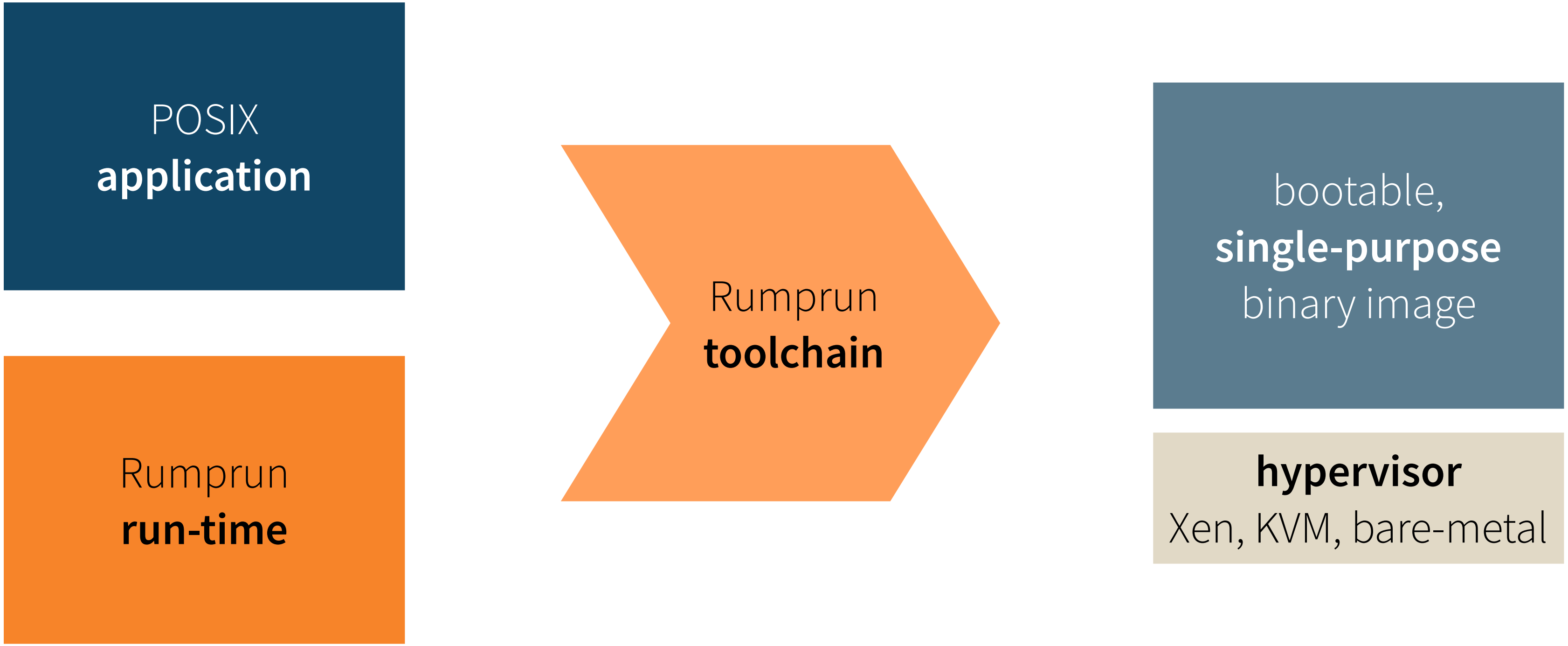
# **unikernels:** how did we get here

- **batch processing:** single app on a single machine
- **time sharing:** multiple apps on a single maschine
  - process isolation, multi-user
  - shared dependencies
  - sandboxing
  - virtualisation, containerization
- **unikernel:** single app on a  virtual machine
  - specialized, no moving parts, isolated through hypervisor

removing

# layers of abstraction

# **uni** · kernel



POSIX
**application**

Rumprun
**run-time**

Rumprun
**toolchain**

bootable,
**single-purpose**
binary image

**hypervisor**
Xen, KVM, bare-metal

# getting started

```
$ git clone http://repo.rumpkernel.org/rumprun
$ cd rumprun
$ git submodule update --init
$ CC=cc ./build-rr.sh hw
[...]
>> Built rumprun for hw : x86_64-rumprun-netbsd
>> cc: x86_64-rumprun-netbsd-gcc
>>
>> ./build-rr.sh ran successfully
```

# Rumprun **workflow**

**step 1:** cross-compile

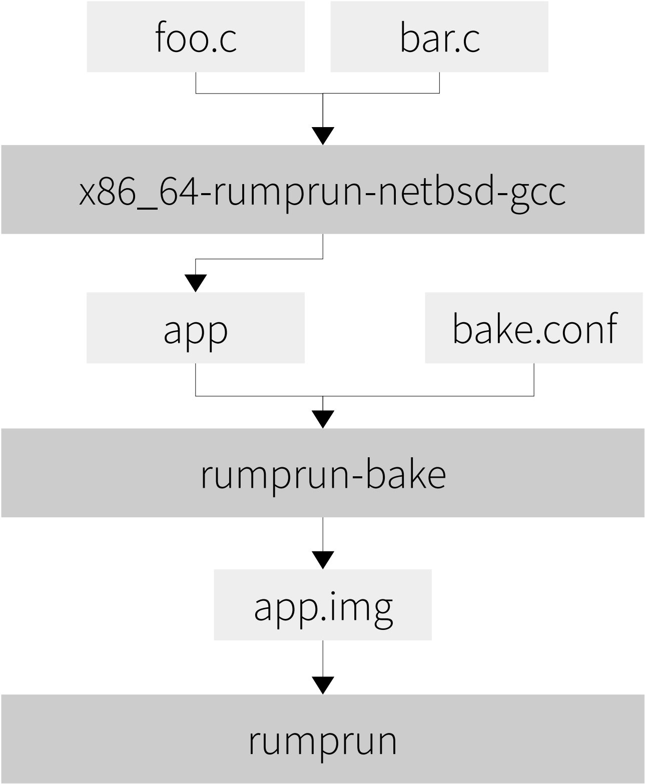
- compile against NetBSD's libc
- support for autotools & cmake

**step 2:** bake

- choose hypervisor, drivers & subsystems

**step 3:** launch

- mount points for block devices
- configure network
- environment variables, main args



running

# hello world

# rumprun-**packages**

## applications

- apache2, nginx, haproxy
- redis, mysql, sqlite, leveldb
- tor, mpg123, …

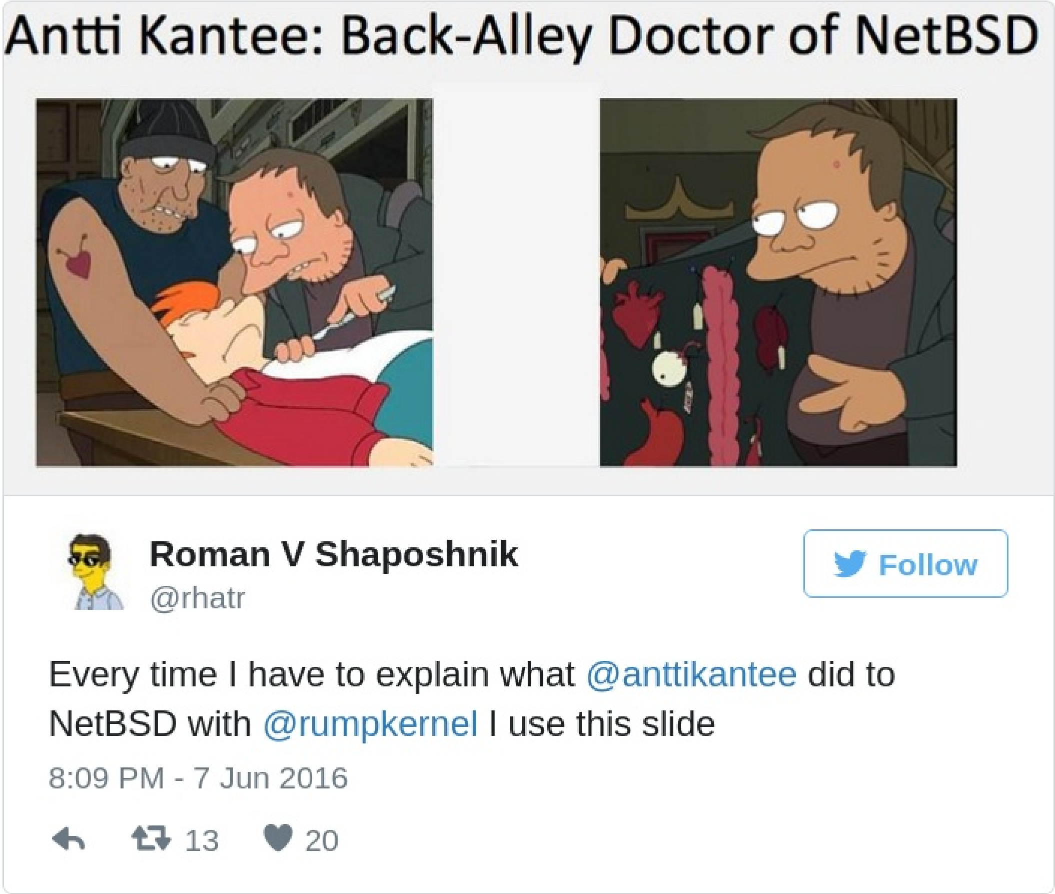
## programming languages

- C/C++ (from toolchain)
- Lua, PHP, Python, Ruby, node.js
- Rust, Erlang, Go

## contiguous integration

- ensuring all packages build
- running twice a day (3+hrs)

# second **demonstration**



Antti Kantee: Back-Alley Doctor of NetBSD

Roman V Shaposhnik
@rhatr

Every time I have to explain what @anttikantee did to NetBSD with @rumpkernel I use this slide

8:09 PM - 7 Jun 2016

*“Pssst, want a portable,
kernel-quality TCP/IP stack?”*

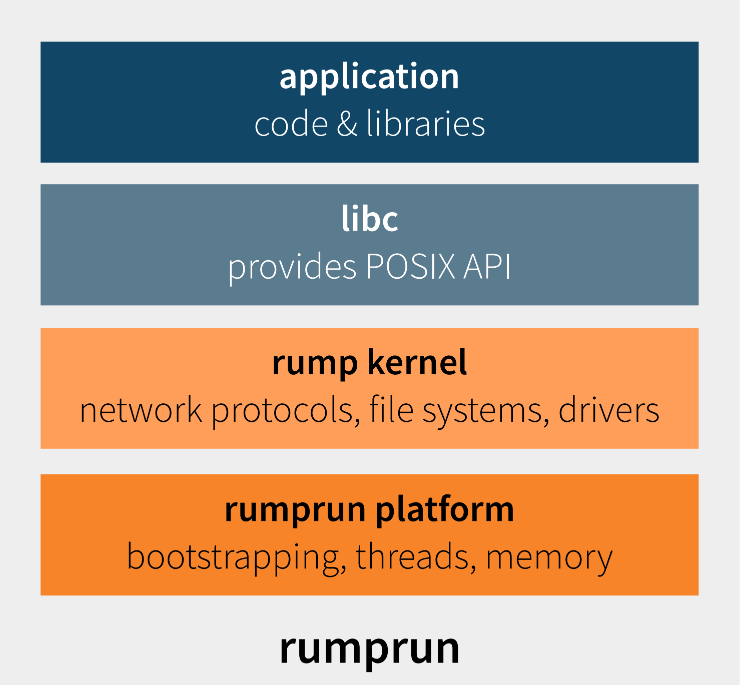
**rump kernels**

- free, reusable, componentized, kernel-quality drivers
  - hardware drivers
  - file systems, network protocols
  - POSIX system calls

https://twitter.com/rhatr/status/740244315411251201
https://blog.xenproject.org/2015/08/06/on-rump-kernels-and-the-rumprun-unikernel/

# **Rumprun:** unikernel based on rump kernels

- from rump/NetBSD
  - rump kernel & drivers
  - (mostly) unmodified libc
- our own
  - platform-specific bootstrapping
  - “bare-metal” hypercall implementation
    - thread scheduler
    - memory allocator
    - console output



# **debugging** unikernels

## gdb

- using qemu's debugging interface
  - same for Xen
- unikernel is a single ELF file
  - can step through the full stack

## rump sysproxy

### rumpctrl

- “remote shell”
- ifconfig, mount, sysctl

### syscalls over TCP/IP

- not enabled by default
- even works for bare-metal

# limitations

single address-space

- no processes
- no virtual memory
- no signals

toolchain

- still experimental

threading

- cooperative
- single-core
  - need to spawn multiple unikernels to use multiple cores

# more **rump kernel**

**frankenlibc**

- alternative rump unikernel
- interesting software architecture
- runs on Linux/FreeBSD/NetBSD
  - seccomp & Capsium support

**nolibc** Rumprun

- directly use the rump kernel
- some assembly required
- experimental Linux/LibOS support

## getting started:

http://rumpkernel.org

@rumpkernel

**#rumpkernel** irc.freenode.net

## contact me:

gandro@rumpkernel.org

@gandro23

**gandro** on irc.freenode.net

## documentation:

- wiki, tutorials, how-to
- video tutorials
- rump man pages

## code:

repo.rumpkernel.org/rumprun

repo.rumpkernel.org/rumprun-packages