What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

DPDK Unikernels with Unikraft

Sharan Santhanam

NEC Laboratories Europe GmbH

12th Nov 2019

This work has received funding from the European Unions Horizon 2020 research and innovation program under grant agreements no.825377 (UNICORE). This work reflects only the authors views and the European Commission is not responsible for any use that may be made of the information it contains.
What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK
VNF with DPDK Ecosystem

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Can we do better?

⇝ Guest OS specialization
⇝ Boot Time
⇝ Isolation within the guest

Let's discuss Unikernel...
Unikraft meets DPDK

What we saw

Can we do better?

⇝ Guest OS specialization
⇝ Boot Time
⇝ Isolation within the guest
VNF with DPDK Ecosystem

Can we do better?

- Guest OS specialization
- Boot Time
- Isolation within the guest

Let’s discuss Unikernel...
Unikraft - Do One Thing and Do It Well

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Virtual Machine

Unikernel

Unikernel - Do One Thing and Do It Well
Unikernel - Do One Thing and Do It Well

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Unikernel are purpose built
- Thin kernel layer
- Single image with application, specific kernel primitives
Unikraft - Do One Thing and Do It Well

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Unikernel are purpose built
- Thin kernel layer
- Single image with application, specific kernel primitives

No isolation within a Unikernel
- Flat address space
Unikernel - Do One Thing and Do It Well

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Unikernel are purpose built
- Thin kernel layer
- Single image with application, specific kernel primitives

No isolation within a Unikernel
- Flat address space

Software Stack Specialization
"Really Unikernels!"

- Fast instantiation, destruction and migration times
  - 10 milliseconds or less
    (LightVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015])
”Really Unikernels!”

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Unikraft

DPDK

Sharan Santhanam

⇝ Fast instantiation, destruction and migration times

- 10 milliseconds or less
  (LightVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015])

⇝ Low memory footprint

- Few MBs of RAM or less
  (ClickOS [Martins NSDI 2014])
“Really Unikernels!”

~ Fast instantiation, destruction and migration times
  - 10 milliseconds or less
    (LightVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015])

~ Low memory footprint
  - Few MBs of RAM or less
    (ClickOS [Martins NSDI 2014])

~ High Deployment Density
  - 8k guests on a single x86 server
    (LightVM [Manco SOSP 2017])
"Really Unikernels!"

~~ Fast instantiation, destruction and migration times

- 10 milliseconds or less
  (LightVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015])

~~ Low memory footprint

- Few MBs of RAM or less
  (ClickOS [Martins NSDI 2014])

~~ High Deployment Density

- 8k guests on a single x86 server
  (LightVM [Manco SOSP 2017])

~~ High Performance

- 10-40Gbit/s Ethernet throughput with a single guest CPU
  (ClickOS [Martins NSDI 2014], Elastic CDNs [Kuenzer VEE 2017])
"Really Unikernels!"

- Fast instantiation, destruction and migration times
  - 10 milliseconds or less
    (LightVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015])
- Low memory footprint
  - Few MBs of RAM or less
    (ClickOS [Martins NSDI 2014])
- High Deployment Density
  - 8k guests on a single x86 server
    (LightVM [Manco SOSP 2017])
- High Performance
  - 10-40Gbit/s Ethernet throughput with a single guest CPU
    (ClickOS [Martins NSDI 2014], Elastic CDNs [Kuenzer VEE 2017])
- Reduced attack surface
  - Small trusted compute base
  - Strong isolation by hypervisor
La-la Land

So, Unikernel

- High Performance
- Isolation and reduced attack surface.
- Faster Instantiation Time
- Smaller image size
La-la Land

So, Unikernel

👍 High Performance
👍 Isolation and reduced attack surface.
👍 Faster Instantiation Time
👍 Smaller image size

The problem with Unikernel development:

👎 Building take several months or longer
👎 Potentially repeat the process for each target application
👎 "Specialization” is hard to build
So, Unikernel

👍 High Performance
👍 Isolation and reduced attack surface.
👍 Faster Instantiation Time
👍 Smaller image size

The problem with Unikernel development:
👎 Building take several months or longer
👎 Potentially repeat the process for each target application
👎 "Specialization" is hard to build

😢 Ooops!!
That's not an effective way of doing things!
What is Unikraft?

**Objectives**

- Support wide range of use cases
- Simplify building and optimizing
- Common and shared code base
- Support different hypervisors
- CPU architectures
What is Unikraft?

**Objectives**

- Support wide range of use cases
- Simplify building and optimizing
- Common and shared code base
- Support different hypervisors
- CPU architectures

**Unikraft**

- Everything is a library
- Decomposed OS functionality
- Unikraft's two components:
  - Library Pool
  - Build Tool

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK
What is Unikraft?

Objectives

- Support wide range of use cases
- Simplify building and optimizing
- Common and shared code base
- Support different hypervisors
- CPU architectures

Unikraft

- Everything is a library
- Decomposed OS functionality
- Unikraft's two components:
  - Library Pool
  - Build Tool

Unikraft says Hi!!

Source is BSD-licensed

Kconfig based build system
Take an existing application
- For example, a Python application or a l2fwd
What we saw

Introduction

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Take an existing application
- For example, a Python application or a l2fwd

Pick Unikraft functionality
- Pool of drivers and standard libraries
libukforest - Unikraft System Overview

~ Take an existing application
- For example, a Python application or a l2fwd

~ Pick Unikraft functionality
- Pool of drivers and standard libraries

~ Pick a platform and architecture
- Pool of drivers and standard libraries
## What we saw

### Introduce Unikraft

**Unikraft meets DPDK**

**Unikraft** within DPDK

**Synergy between Unikraft and DPDK**

---

### Take an existing application

- For example, a Python application or a l2fwd

### Pick Unikraft functionality

- Pool of drivers and standard libraries

### Pick a platform and architecture

- Pool of drivers and standard libraries

### Build Unikraft application
libukforest - Unikraft System Overview

Take an existing application
- For example, a Python application or a l2fwd

Pick Unikraft functionality
- Pool of drivers and standard libraries

Pick a platform and architecture
- Pool of drivers and standard libraries

Build Unikraft application
Unikraft - DPDK Target Arch?

### What we saw

- Introduce Unikraft
- Unikraft meets DPDK
- Synergy between Unikraft and DPDK

### Challenges!!

- Build System Integration
- Specialization of Guest OS
- Minimize modification to DPDK library

---

#### Unikraft - DPDK Target Arch?

- **l2fwd**
  - libukdpdk
  - libpthread
  - libvfscore
  - libramfs
  - libkmplat
- **kvm**
  - libx86_64
- **x86**
- **64**
- **uk_dpdk_kvm_x86_64**
Unikraft - DPDK Target Arch?

<table>
<thead>
<tr>
<th>l2fwd</th>
<th>libukdpdk</th>
</tr>
</thead>
<tbody>
<tr>
<td>libpthread</td>
<td>intrinsics</td>
</tr>
<tr>
<td>libvfscore</td>
<td>libnetdev</td>
</tr>
<tr>
<td>libramfs</td>
<td>libvirtio</td>
</tr>
<tr>
<td>libkvmplat</td>
<td>libx86_64</td>
</tr>
</tbody>
</table>

Challenges!!

~ Build System Integration
~ Specialization of Guest OS
~ Minimize modification to DPDK library
Build DPDK as an Unikraft Library

Unikraft Build system

- Config.uk (Kconfig based)
  - Handles dependencies across library
  - Enable/Disable Function

DPDK Build System

- Automatic config generation
  - CPU feature flags
Build DPDK as an Unikraft Library

**Unikraft Build system**

- **Config.uk (Kconfig based)**
  - Handles dependencies across library
  - Enable/Disable Function
- **Makefile.uk (make based)**
  - `[LIBNAME].SRCS`
  - `[LIBNAME].CFLAG`
  - `CFLAG`

**DPDK Build System**

- **Automatic config generation**
  - CPU feature flags
- **Makefile (gmake)**
  - `SRCS`
  - `INCLUDE`
  - `CFLAG`
  - `DIRS`
## Build DPDK as an Unikraft Library

### Unikraft Build system

- **Config.uk (Kconfig based)**
  - Handles dependencies across library
  - Enable/Disable Function

- **Makefile.uk (make based)**
  - `[LIBNAME].SRCS`
  - `[LIBNAME].CFLAG`
  - CFLAG

- **exportsyms.uk**

### DPDK Build System

- **Automatic config generation**
  - CPU feature flags

- **Makefile (gmake)**
  - SRCS
  - INCLUDE
  - CFLAG
  - DIRS

- **version map**
# Build DPDK as an Unikraft Library

## Unikraft Build system

- **Config.uk (Kconfig based)**
  - Handles dependencies across library
  - Enable/Disable Function

- **Makefile.uk (make based)**
  - `[LIBNAME].SRCS`
  - `[LIBNAME].CFLAG`
  - `CFLAG`

- **exportsyms.uk**

## DPDK Build System

- **Automatic config generation**
  - CPU feature flags

- **Makefile (gmake)**
  - `SRCS`
  - `INCLUDE`
  - `CFLAG`
  - `DIRS`

- **version map**

---

**libukdpdkbuild**

- Process DPDK Makefile.
## Build DPDK as an Unikraft Library

### Unikraft Build system

<table>
<thead>
<tr>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔷 Config.uk (Kconfig based)</td>
</tr>
<tr>
<td>- Handles dependencies across library</td>
</tr>
<tr>
<td>- Enable/Disable Function</td>
</tr>
<tr>
<td>🔷 Makefile.uk (make based)</td>
</tr>
<tr>
<td>- [LIBNAME].SRCS</td>
</tr>
<tr>
<td>- [LIBNAME].CFLAGS</td>
</tr>
<tr>
<td>- CFLAGS</td>
</tr>
<tr>
<td>🔷 exportsyms.uk</td>
</tr>
</tbody>
</table>

### DPDK Build System

<table>
<thead>
<tr>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔷 Automatic config generation</td>
</tr>
<tr>
<td>- CPU feature flags</td>
</tr>
<tr>
<td>🔷 Makefile (gmake)</td>
</tr>
<tr>
<td>- SRCS</td>
</tr>
<tr>
<td>- INCLUDE</td>
</tr>
<tr>
<td>- CFLAGS</td>
</tr>
<tr>
<td>- DIRS</td>
</tr>
<tr>
<td>🔷 version map</td>
</tr>
</tbody>
</table>

---

**libukdpdkbuild**

- Process DPDK Makefile.
- Add DPDK library
### Unikraft Build System

- **Config.uk** (Kconfig based)
  - Handles dependencies across library
  - Enable/Disable Function
- **Makefile.uk** (make based)
  - `[LIBNAME].SRCS`
  - `[LIBNAME].CFLAG`
  - `CFLAG`
- **exportsyms.uk**

### DPDK Build System

- **Automatic config generation**
  - CPU feature flags
- **Makefile** (gmake)
  - `SRCS`
  - `INCLUDE`
  - `CFLAG`
  - `DIRS`
- **version map**

---

### libukdpdkbuild

- **Process DPDK Makefile.**
  - Add DPDK library
  - Support newer version of DPDK
**What we saw**

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

---

## Unikraft Build system

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config.uk</td>
<td>Config.uk (Kconfig based)</td>
</tr>
<tr>
<td>Makefile.uk</td>
<td>Makefile.uk (make based)</td>
</tr>
<tr>
<td>exportsyms.uk</td>
<td></td>
</tr>
</tbody>
</table>

## DPDK Build System

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic config generation</td>
<td>CPU feature flags</td>
</tr>
<tr>
<td>Makefile (gmake)</td>
<td>SRCS, INCLUDE, CFLAG, DIRS</td>
</tr>
<tr>
<td>version map</td>
<td></td>
</tr>
</tbody>
</table>

---

**libukdpdkbuild**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process DPDK Makefile.</td>
<td>Add DPDK library</td>
</tr>
<tr>
<td>Add dpdk specific configuration file.</td>
<td>Support newer version of DPDK</td>
</tr>
</tbody>
</table>
Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

Unikraft - DPDK Target Arch
Specialize the Guest OS

- Memory management
- Bus/Device Management
- CPU Scheduling and CPU Features
Unikraft as EAL in DPDK

Specialize the Guest OS

〜 Memory management
〜 Bus/Device Management
〜 CPU Scheduling and CPU Features

Memory Management

🔹 Unikraft: flat page table since boot
🔹 Huge pages based 2MB sized pages
🔹 Memory region can be explicitly assigned to the Application
🔹 Custom memory allocator per memory region
Unikraft as EAL in DPDK

Specialize the Guest OS

- Memory management
- Bus/Device Management
- CPU Scheduling and CPU Features

Bus/Device Management

- A simpler bus/device interface
- Directly attached device and usable by DPDK with unikraft
Unikraft as EAL in DPDK

Specialize the Guest OS

- Memory management
- Bus/Device Management
- CPU Scheduling and CPU Features

CPU Scheduling and CPU Features

- Application decides on scheduling on the core.
- Minimal interference / resource usage for other purpose within guest.
Interface between Unikraft and DPDK

- Introduce Unikraft
- Unikraft meets DPDK
- Synergy between Unikraft and DPDK

Interface between Unikraft and DPDK

- Librte_ethdev
- App_tx_burst
- Mbuf
- Nbuf
- Virtio_xmit_pkts_packed

- Libknetdev
- App_tx_burst
- Uk_netdev_bx_one
- Nbuf
- Virtio_netdev_xmit
Interface between Unikraft and DPDK

Unikraft

DPDK

Sharan Santhanam

What we saw

Introduce Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK
Interface between Unikraft and DPDK

Unikraft meets DPDK

Synergy between Unikraft and DPDK
Interface between Unikraft and DPDK

Unikraft meets DPDK
Synergy between Unikraft and DPDK
Future Work

⇝ Evaluate performance
⇝ Add SMP support
⇝ Add NUMA support
⇝ Use DPDK drivers directly
What we saw

### Unikraft

Unikraft meets DPDK

Unikraft within DPDK

Synergy between Unikraft and DPDK

---

Join us!

- **Wiki**
  - https://wiki.xenproject.org

- **Documentation**
  - http://docs.unikraft.org/

- **Sources**
  - http://xenbits.xen.org/gitweb/ (Namespace: Unikraft)
  - https://github.com/unikraft

- **Mailing List**
  - minios-devel mailing list
# What we think

<table>
<thead>
<tr>
<th>Unikraft</th>
<th>DPDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Support multiple platforms</td>
<td>- Performance of Network stack</td>
</tr>
<tr>
<td>- Specialized Guest OS</td>
<td>- Specialized VNF</td>
</tr>
<tr>
<td>- Simpler Management Device</td>
<td>- Wealth of knowledge DPDK driver</td>
</tr>
<tr>
<td>- Increased control for an application</td>
<td>- Increased application base</td>
</tr>
</tbody>
</table>

## What we saw

- Introduce Unikraft
- Unikraft meets DPDK
- Synergy between Unikraft and DPDK

## What we think

- Unikraft supports multiple platforms.
- Specialized Guest OS.
- Simpler Management Device.
- Increased control for an application.

- DPDK provides performance of the Network stack.
- Specialized VNF.
- Wealth of knowledge in the DPDK driver.
- Increased application base.
### What we think

<table>
<thead>
<tr>
<th>Unikraft</th>
<th>DPDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Support multiple platforms</td>
<td>- Performance of Network stack</td>
</tr>
<tr>
<td>- Specialized Guest OS</td>
<td>- Specialized VNF</td>
</tr>
<tr>
<td>- Simpler Management Device</td>
<td>- Wealth of knowledge DPDK driver</td>
</tr>
<tr>
<td>- Increased control for an application</td>
<td>- Increased application base</td>
</tr>
</tbody>
</table>

**:What do you think?:**