

Unikraft – Crafting Unikernels

Simon Kuenzer, Felipe Huici, Florian Schmidt
<firstname.lastname>@neclab.eu

SYSML Group
NEC Laboratories Europe

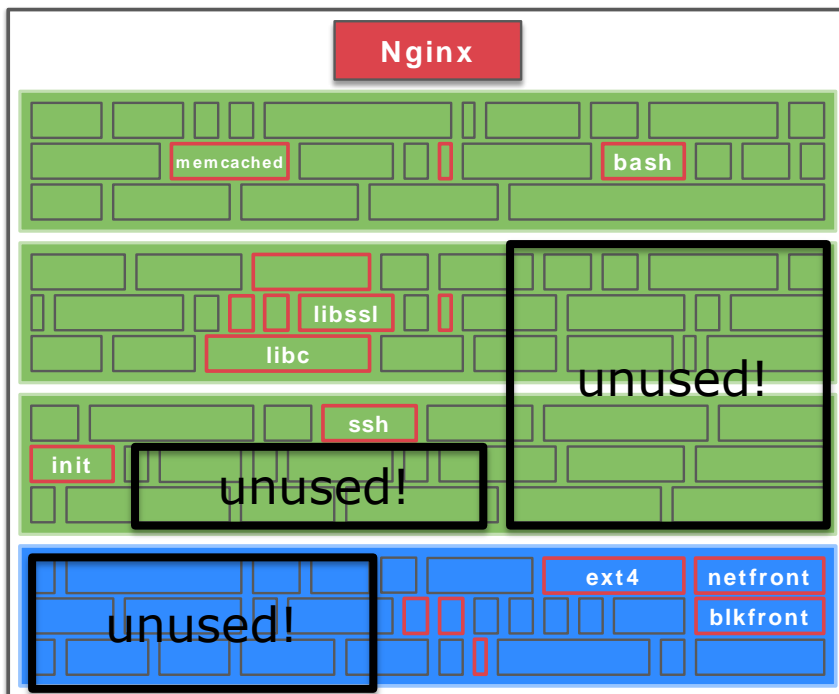
<http://sysml.neclab.eu>

Advantages and the Problem with Specialization

Standard OS/VM/container image:

- **lots** of unnecessary code
- **lots** of overhead!

Specialized image: only what's needed is there **but lots** of development time! (have to change code **by hand**)



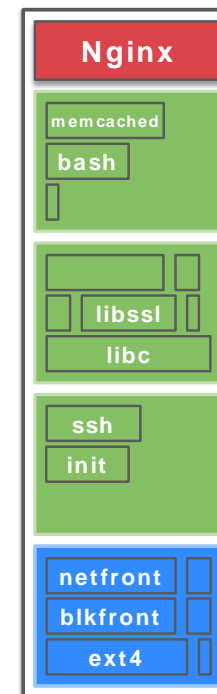
User Application

3rd Party Applications

Libraries

Services

Kernel



Unikraft: The Insight

In a perfect world...

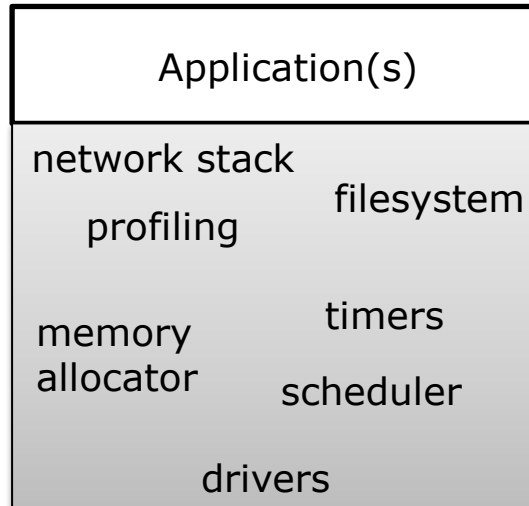
- We would have a menu of libraries for all possible components applications might need
- We would be able to use that menu to select **only** the functionality needed (and possibly automate the selection process)
- A system would **automatically** build a lean, high performance image for the application we're interested in and the platforms we care for



Unikraft is precisely this system!

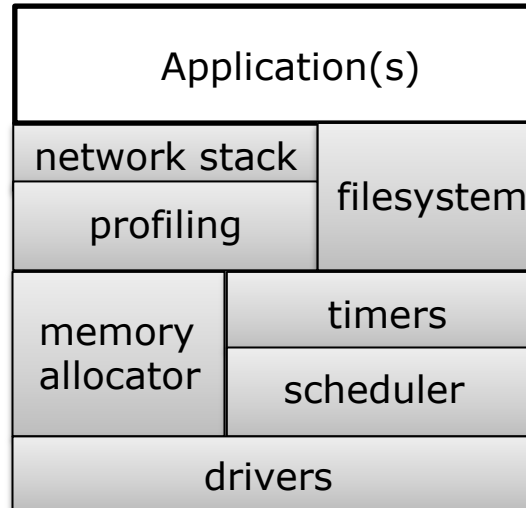
Unikraft – Operating System Decomposition

- Standard operating systems are **monolithic**: they are not modular so it's **not** possible to separate their parts



Unikraft – Operating System Decomposition

■ Could we decompose, i.e., break apart an operating system?

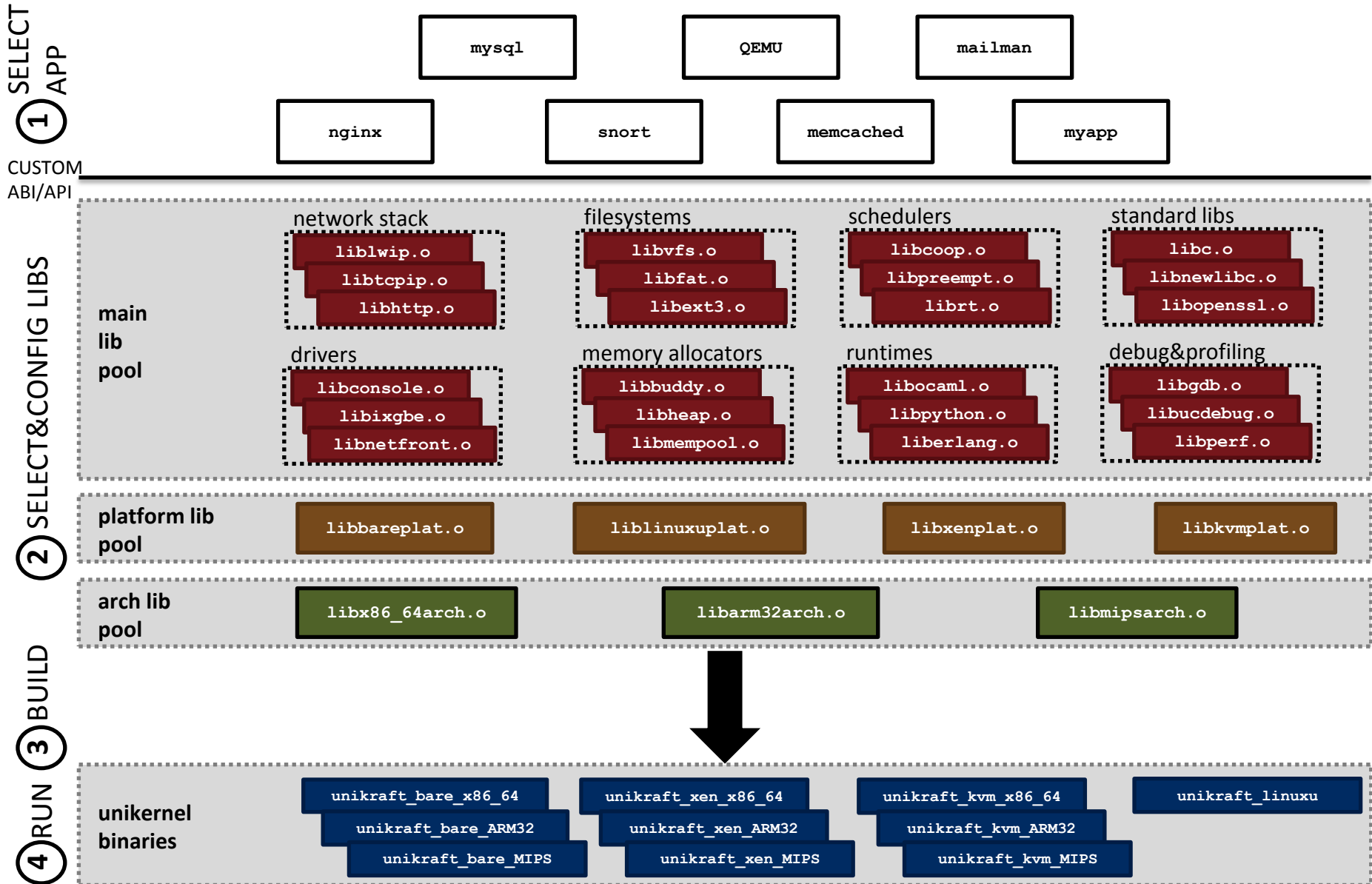


Unikraft – Operating System Decomposition

■ Could we decompose, i.e., break apart an operating system?

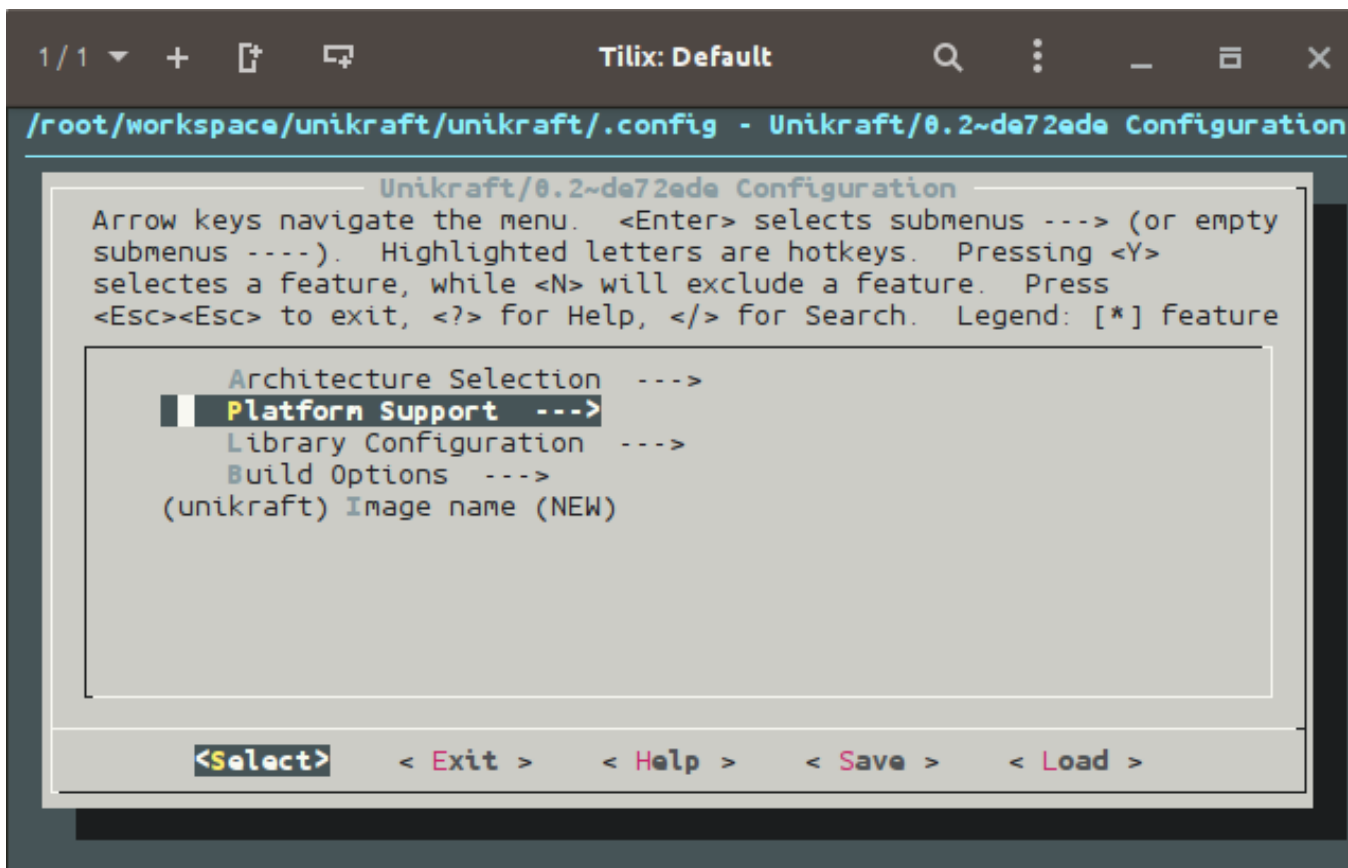


The Unikraft Library and Build System



Building a Specialized Image in One Minute

- Type "make menuconfig"
- Choose options in the menu that you want for your application
- Choose your target platform(s) (e.g., Xen, KVM, Linux, baremetal)
- Save config and type "make"



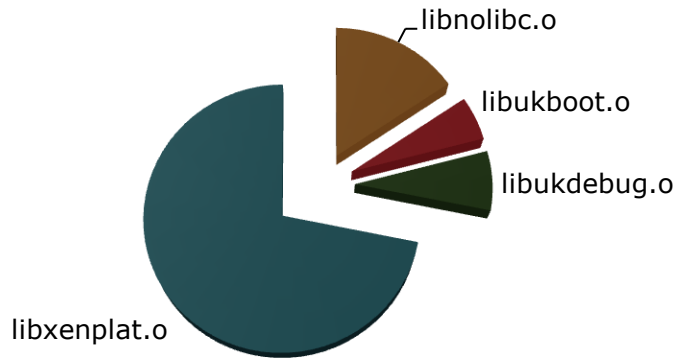
The screenshot shows a terminal window titled "Tilix: Default" with the current directory set to "/root/workspace/unikraft/unikraft/.config - Unikraft/0.2~de72ede Configuration". The main content is a configuration menu for "Unikraft/0.2~de72ede Configuration". The menu instructions state: "Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> selects a feature, while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] feature". The menu items are: "Architecture Selection --->", "Platform Support --->" (highlighted with a white bar), "Library Configuration --->", "Build Options --->", and "(unikraft) Image name (NEW)". At the bottom, there are navigation options: "<Select>", "< Exit >", "< Help >", "< Save >", and "< Load >".

An Unikraft Image Example

Xen PV x86_64 binary

Compiles to a 32.7kB image

`unikraft_xen-x86_64.o` (50,2kB)



`unikraft_xen-x86_64`
(32,7kB)

Boots and prints message to debug console (with min. 208kB RAM)

```
1/1 + [ ] [ ] Tilix: Default [ ] [ ] [ ] [ ] [ ]
(d9) Info: [libxenplat] setup.c @ 174 : Entering from Xen (x86, PV)...
(d9) Info: [libxenplat] setup.c @ 189 : start_info: 0x156000
(d9) Info: [libxenplat] setup.c @ 190 : shared_info: 0x2000
(d9) Info: [libxenplat] setup.c @ 191 : hypercall_page: 0x3000
(d9) Info: [libxenplat] setup.c @ 154 : start_pfn: 15e
(d9) Info: [libxenplat] setup.c @ 155 : max_pfn: 20000
(d9) Info: [libxenplat] mn.c @ 160 : Mapping memory range 0x15e000 - 0x20000000
(d9) Kern: Welcome to
(d9) Kern:
(d9) Kern:
(d9) Kern:
(d9) Kern: Titan 0: 2-de72ede
(d9) Info: [libukboot] boot.c @ 72 : Calling main(, ['unikraft', 'console=hvc0'])
(d9) Kern: weak main() called. Symbol was not replaced!
(d9) ERR: [libukboot] boot.c @ 195 : weak main() called. Symbol was not replaced!
(d9) Info: [libukboot] boot.c @ 82 : main returned -22, halting system
[ ] develi root ~ > workspace > unikraft > unikraft [ ]
```

Potential Unikraft-built Systems

Specialized Python images for Xen, KVM and ARM, x86_64

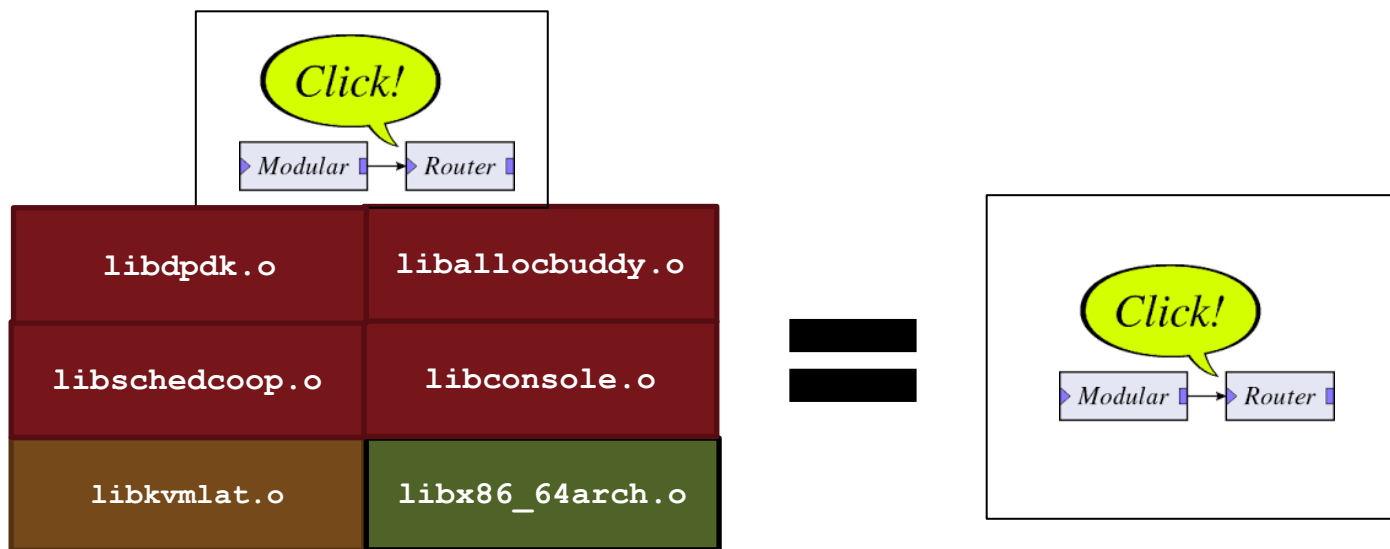


Verticals:

- IoT gateways
- Smart city gateways
- Cloud computing platforms (e.g., AWS Lambda)

Potential Unikraft-built Systems

Specialized NFV image for KVM on x86_64



Verticals:

- vCPE
- vRouters
- vBRAS
- Your network application here!