The new VVorld

Robert Watson and Bjoern Zeeb
(and thanks to Marko Zec)
The FreeBSD Project

UKUUG Spring Conference 2010

UNIVERSITY OF CAMBRIDGE

OS research and development performed by FreeBSD Project, University of Zagreb, FreeBSD Foundation, NLNet, and other contributors over a decade

Still a work-in-progress, but exciting technology coming soon

Introduction

- About virtualization
- FreeBSD Jails
- Virtualizing a kernel
- A virtualized network stack
- A few application ideas

UNIVERSITY OF CAMBRIDGE

2

What is virtualization?

- Illusion of multiple virtual **X** on one real **X**
  - Virtual memory address spaces
  - VLANs, VPNs, and overlay networks
  - Storage volume management
  - Virtual machines, OS instances
- ... you can solve any problem with another level of indirection ...

3

"... a level of indirection..."

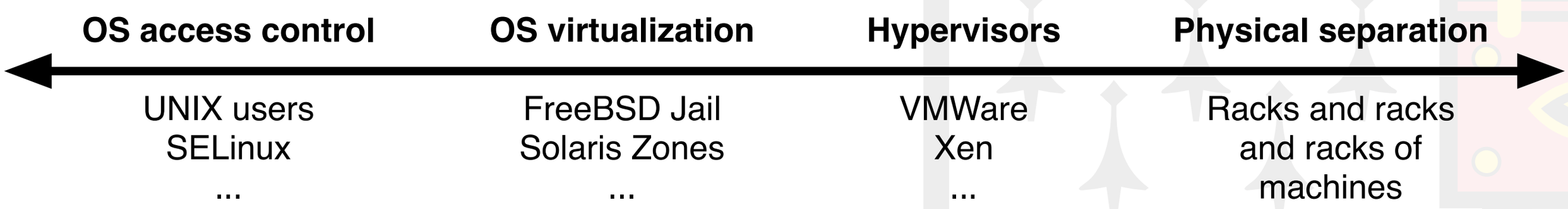
Why virtualize?

- Sharing with the illusion of exclusive use
- Consolidation, managed overcommit
- Flexibility in implementation
- Security and robustness
- Administrative delegation

UNIVERSITY OF CAMBRIDGE

4

Virtualization spectrum



- Tradeoffs: scheduler integration, efficient sharing, overcommit opportunities, functionality, security/isolation, resource management, administrative delegation, ...

Example:
With OS virtualization you get full scheduler integration, but migration very hard
With Hypervisors you get really bad scheduling, but migration is relatively easy

OS virtualization

- Single OS kernel instance, many userspaces
- Safe root delegation, various constraints
- Efficient resource sharing with overcommit
- No hypervisor/virtual device overhead
- ISP virtual hosting, server consolidation, ...

6

History of FreeBSD Jail

1999	Jails merged to FreeBSD 4.x
2002	Virtualized network stack prototype
2006	NLNet/FreeBSD Foundation fund multi-year VIMAGE development project
2007	Jail-friendly ZFS merged from Open Solaris
2008	Multi-IPv4/v6/no-IP patches; VNET integration starts
2009	Hierarchical jail support; FreeBSD 8.0 with highly experimental options VIMAGE shipped

Earliest open source OS virtualization we're aware of

Virtualization work has long timeline

Why change Jail?

- Jail is fast, efficient, secure, useful
- But, Jail "subsets" rather than "virtualizes"
  - For example: employs chroot() internally
    - Some resources subset poorly
    - E.g., System V IPC, loopback interface, ...
- Virtualization is a functional improvement

8

Virtualizing OS services

- New abstraction: the **virtual instance**
- **Replicate** global objects per-instance
- **Multiplex** or **replicate** threads, timers
- **Tag** subjects with virtual instances
- Consider **administrative interfaces**
- Examine **privileges** carefully
- Plan **inter-instance plumbing**
- How to **start** and **stop** instances?

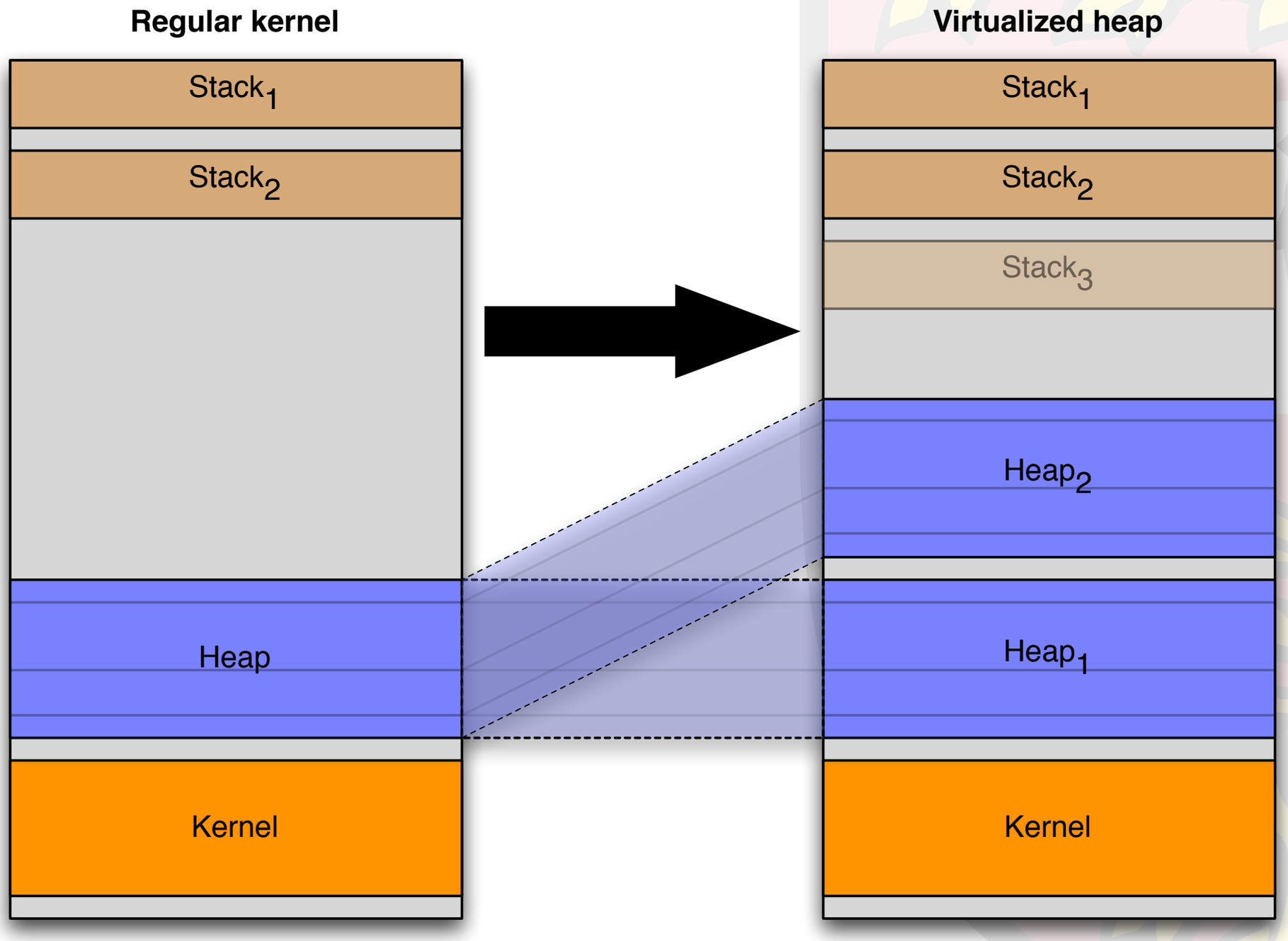
9

Virtual kernel infrastructure

- Sounds complicated, but some new tools
  - Virtualized global variables
  - Virtualized startup/shutdown
  - Virtualized sysctl MIB entries
  - Virtualization-enhanced debugging
  - Multiplex virtualization onto netisrs

10

Virtualized heap



Goal: make it easy for us to take one of something and make many

Notice that same layout is used for virtual instances as original

Virtual global variables

- Tag selected globals as **virtual** in source
- Placed in different ELF section when linked
- Each VNET instance gets a copy of section
- Thread context carries VNET reference
- Globals mapped to VNET when accessed
- Can compile to regular globals if desired

12

Can compile out during development but still in tree. In fact, default today.

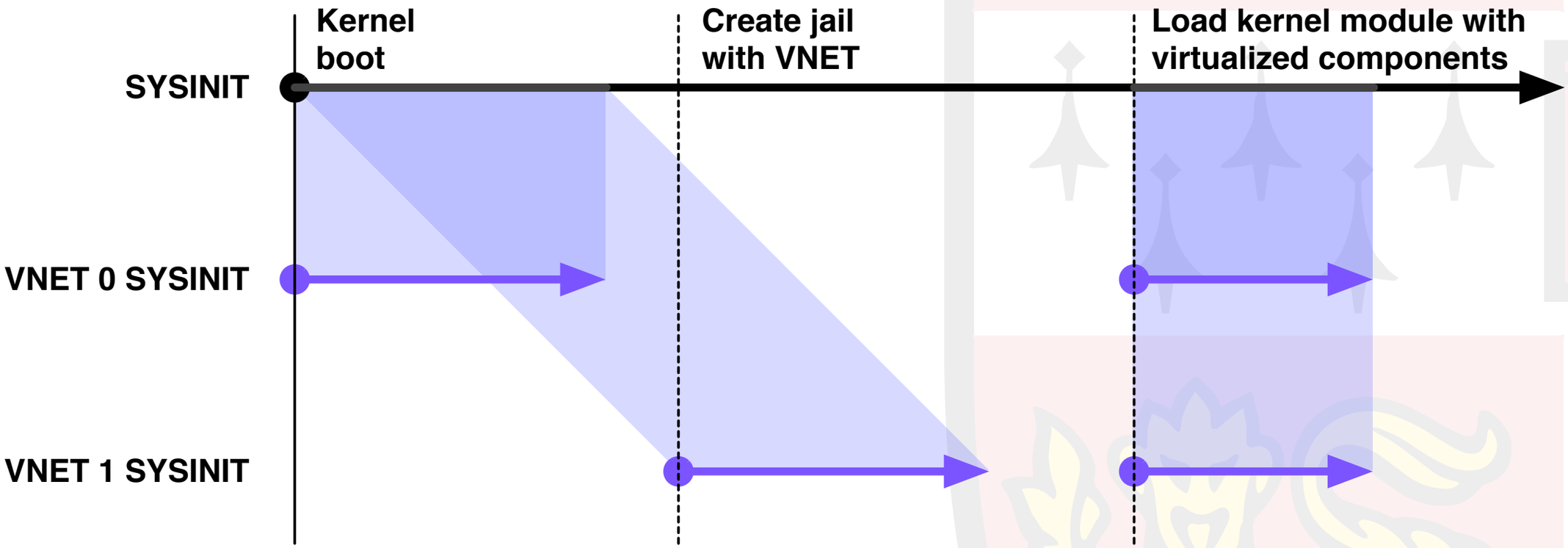
Also valuable for embedded.

```
VNET_DEFINE(struct inpcbhead, ripcb);
VNET_DEFINE(struct inpcbinfo, ripcbinfo);

#define V_ripcb         VNET(ripcb)
#define V_ripcbinfo     VNET(ripcbinfo)

...

void
rip_init(void)
{

        INP_INFO_LOCK_INIT(&V_ripcbinfo, "rip");
        LIST_INIT(&V_ripcb);
```


13

Goal: make virtualized programming natural

Virtualized boot



Portions of previously serialized kernel and module startup are now per-VNET

UNIVERSITY OF CAMBRIDGE

14

Tag bits of boot process that now need to be per-VNET.

Module case is interesting, and tricky.

Virtual kernel startup

- Kernel, module startup uses SYSINIT()
  - Functions tagged with special ELF section
  - Sorted and executed "in order"
    - Used for 99% of FreeBSD kernel init
- Some events now need to be virtualized
- Add a new event set, run once per VNET

15

```
static void
vnet_igmp_init(const void *unused __unused)
{
        CTR1(KTR_IGMPV3, "%s: initializing", __func__);
        LIST_INIT(&V_igi_head);
}
VNET_SYSINIT(vnet_igmp_init, SI_SUB_PSEUDO,
    SI_ORDER_ANY, vnet_igmp_init, NULL);
```

```
static void
vnet_igmp_uninit(const void *unused __unused)
{
        CTR1(KTR_IGMPV3, "%s: tearing down", __func__);
        KASSERT(LIST_EMPTY(&V_igi_head),
            ("igi list not empty; detached?"));
}
VNET_SYSUNINIT(vnet_igmp_uninit, SI_SUB_PSEUDO,
    SI_ORDER_ANY, vnet_igmp_uninit, NULL);
```


16

Again: goal to make it natural.

Five-character change to each to say “do it virtualized”.

What to virtualize?

- Start with a virtual network stack
  - Immediate demand due to Jail limitations
  - Zec 2002 prototype
    - Validate performance of approach
    - Can parallelize over many net modules
- In the future: VIPC, ...

17

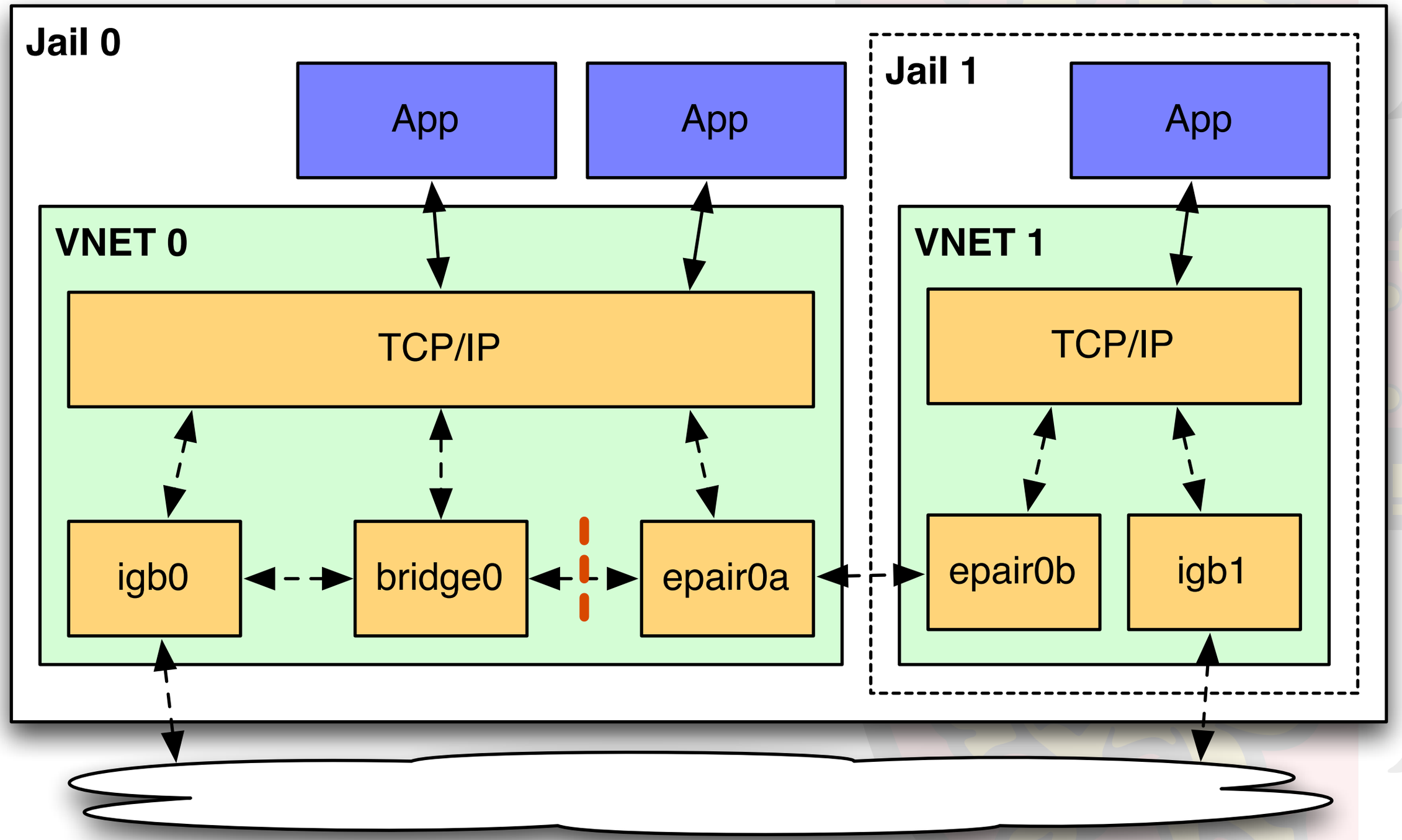
Virtual network stack

- Jails can have their own network stacks
  - TCP/IP socket bindings, routing table, firewall, IPsec, ...
- Real/virtual interfaces belong to one stack, but may be assigned to child stacks
- Packets float between stacks as needed
- Arbitrary virtual network topologies OK

UNIVERSITY OF CAMBRIDGE

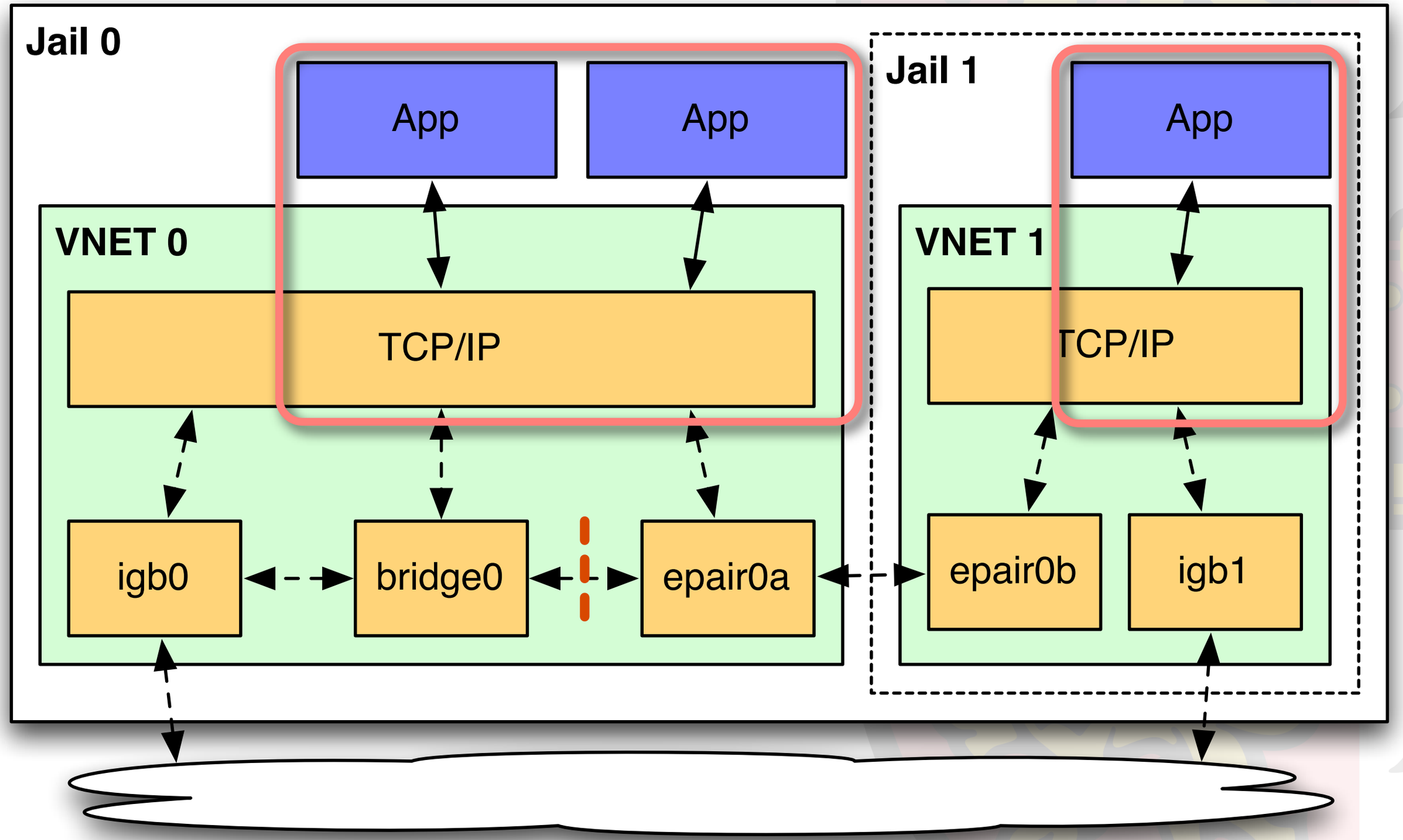
18

Avoid constructs that require additional copying, context switching, etc.



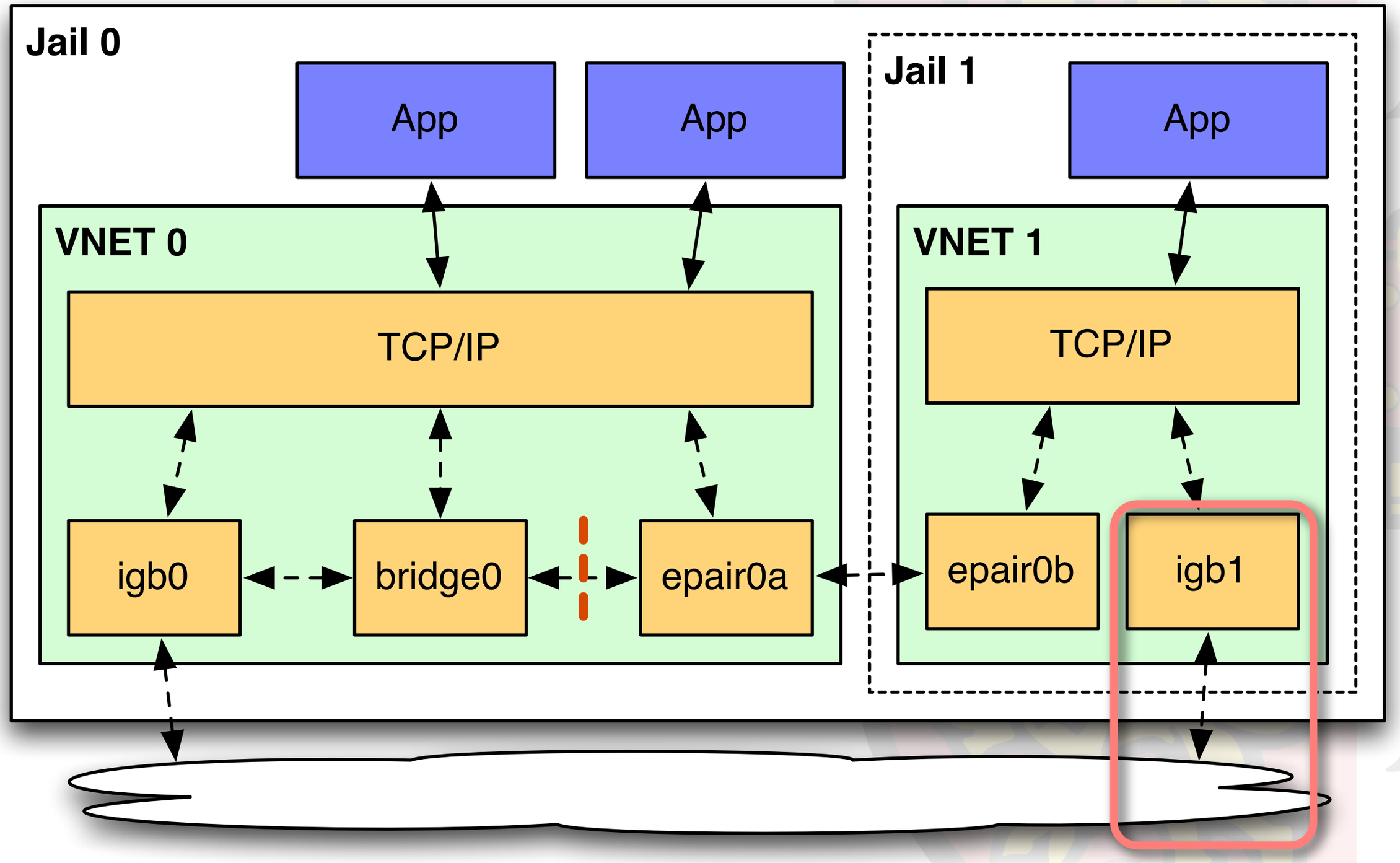
19

Applications pinned to virtual stacks.
Simple case: assign an ifnet to a jail.
Complex case: virtual interfaces, bridging, firewalls, ...



19

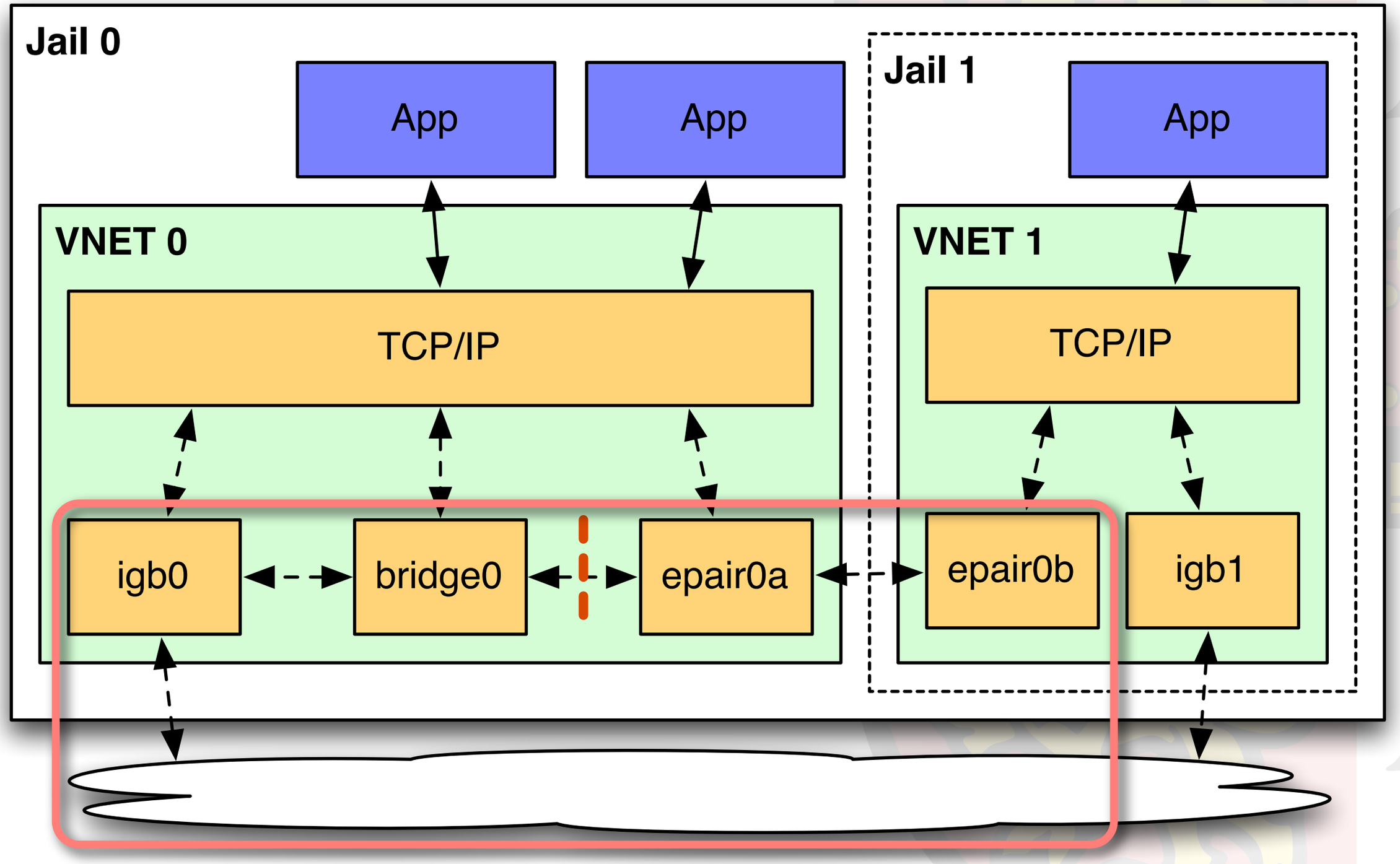
Applications pinned to virtual stacks.
Simple case: assign an ifnet to a jail.
Complex case: virtual interfaces, bridging, firewalls, ...



UNIVERSITY OF CAMBRIDGE

19

Applications pinned to virtual stacks.
Simple case: assign an ifnet to a jail.
Complex case: virtual interfaces, bridging, firewalls, ...



UNIVERSITY OF CAMBRIDGE

19

Applications pinned to virtual stacks.
Simple case: assign an ifnet to a jail.
Complex case: virtual interfaces, bridging, firewalls, ...

Really hard problem: shutting down cleanly

- We've been doing that for years, right?
- Actually, no -- we've been booting for years.
- But we've never shut down the network.
- We just power off and it goes away. :-)
- Now we need **destructors**.

20

Status

- FreeBSD 8.0 VIMAGE "highly experimental"
  - Known memory leaks on stack shutdown
  - Several known crash conditions
  - Many subsystems not fully virtualized
- Foundation will shortly announce new funding for productionization work
- Goal production-quality VIMAGE in 9.1/9.2

21

How to give it a spin

- Update to 8-STABLE or 9-CURRENT
- Compile kernel with "options VIMAGE"
- Simple example:

```
jail -ci vnet path=/jail command=/bin/csh
ifconfig vlan100 vnet <id>
```

- http://wiki.freebsd.org/Image
- WARNING: EXPERIMENTAL

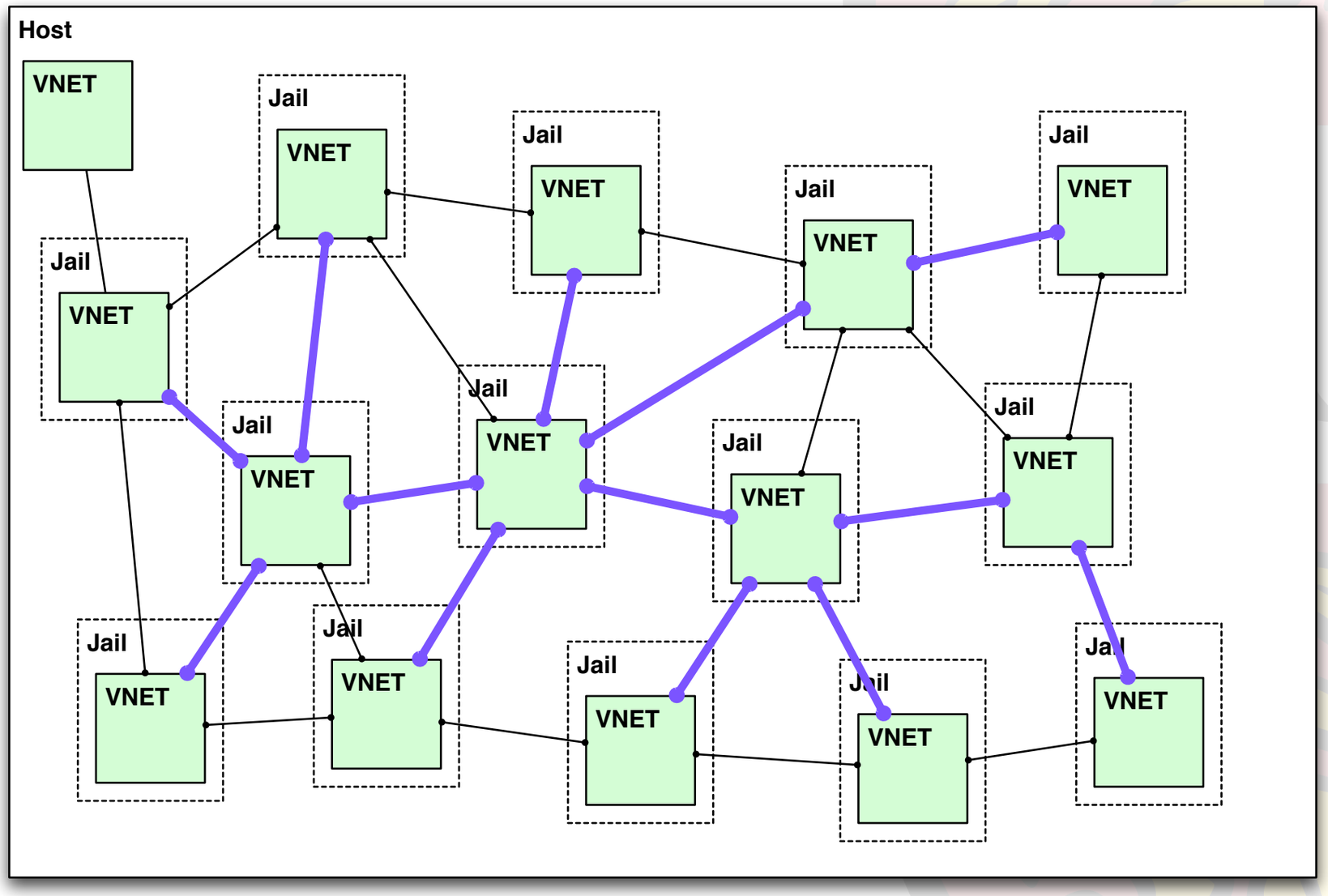
22

A few applications

- Network routing research
- Parallel overlay networks
- Large-scale hosting

23

Routing simulation

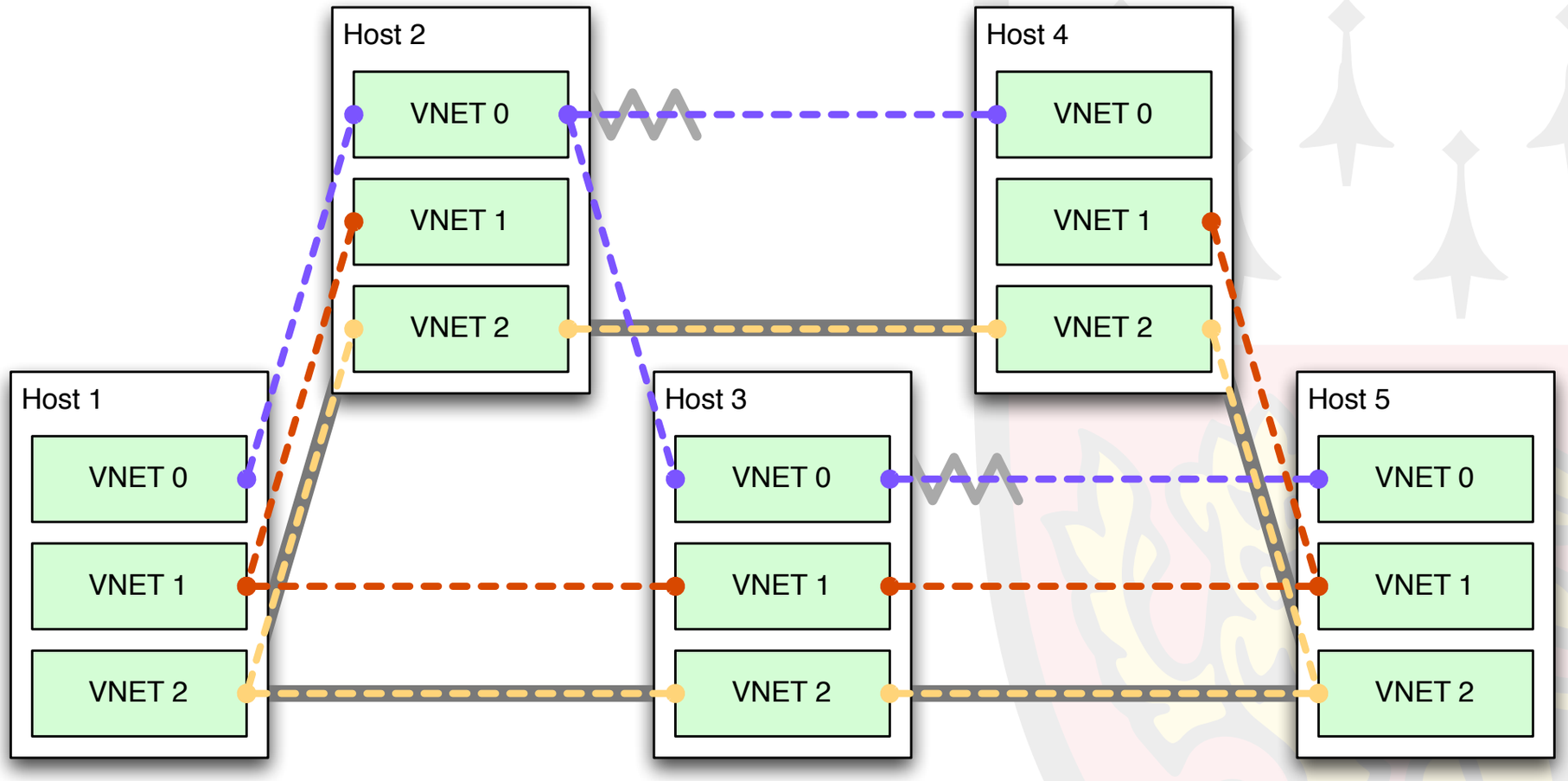


Trivially simulate thousands of nodes with arbitrary topologies and fully functional, independent network stacks

UNIVERSITY OF CAMBRIDGE

24

Virtualized overlay infrastructure

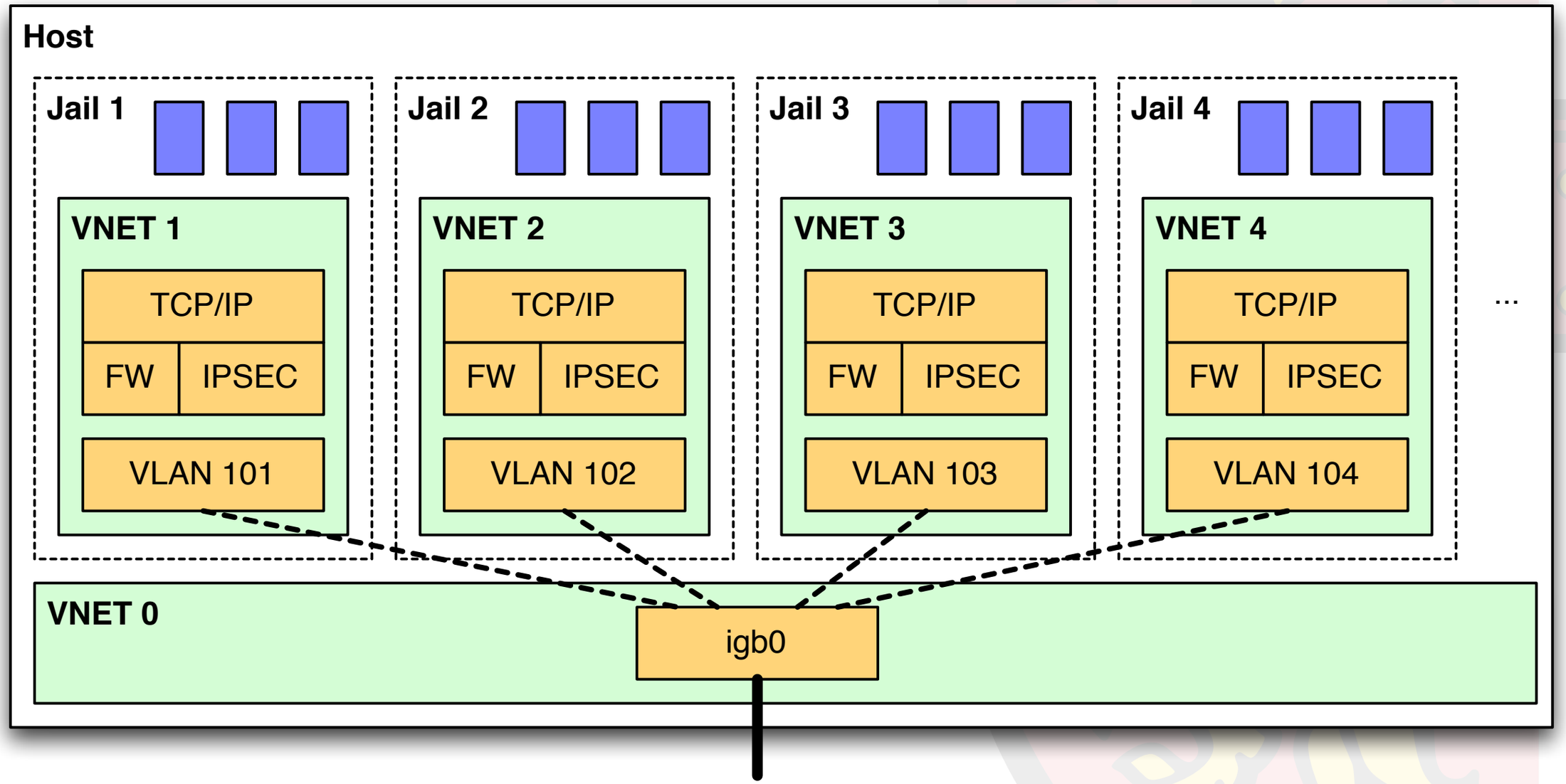


Multiple network stacks allow router/bridge/VAP nodes to implement complex policies using minimal hardware

UNIVERSITY OF CAMBRIDGE

25

Large-scale hosting



Jails each have their own fully delegated connection tables, routing tables, firewalls, IPsec, ...

UNIVERSITY OF CAMBRIDGE

26

Nested jails, with and without VNETs

Some other ideas

- Efficient server consolidation
  - 500K memory overhead vs. 256M+ VM
- Virtualized appliances
  - Multi-instance appliances, such as file stores, firewalls, filters, ...
- Neat: Debian/kFreeBSD on VNETs

UNIVERSITY OF CAMBRIDGE

27

2MB with ZFS or nullfs providing efficient storage, before applications

Conclusion

- Virtual kernel features, such as a virtual network stack, finally becoming a reality
- Prototype operates with increasing stability and little performance overhead
- Adds to virtualization menu; can be combined with other techniques like Xen
- Coming soon(ish)...

UNIVERSITY OF CAMBRIDGE

28

Cleverly, we are able to take advantage of many virtualization-centric hardware optimizations.

For example: MAC address filtering with RSS.