# Modular virtual network stack

Bjoern A. Zeeb
bz@FreeBSD.org

The FreeBSD Project

BSDCan 2010

# Overview

1. Virtualized network stack
2. VNET allocator
3. Modularized network stack
4. MVnS

# Virtualization - Where are we?

- 2002 Marko Zec *BSD network stack virtualization*, BSDCon Europe
- 2008 integration of virtual network stack starts.
- 2009 ”vnet allocator” just before 8.0-R.
- 2010 work continues.
- last week: ”undo lots of whitespace noise”.

# What needs to be done?

- http://wiki.freebsd.org/Image/TODO
- finish and integrate pf, do ipfilter
- generally handle cloned interfaces
- IPX, Appletalk, . . .
- plug resource leaks, do free
- virtualize “missed” variables
- be more careful with resource allocation

# Network stack teardown

- Do you know what happens during shutdown in the network stack?
- How do we do teardown?

# Teardown problem

- ordering?
- currently interfaces go first
- resource leaks and “no free” uma zones
- callout draining / locking
- freeing

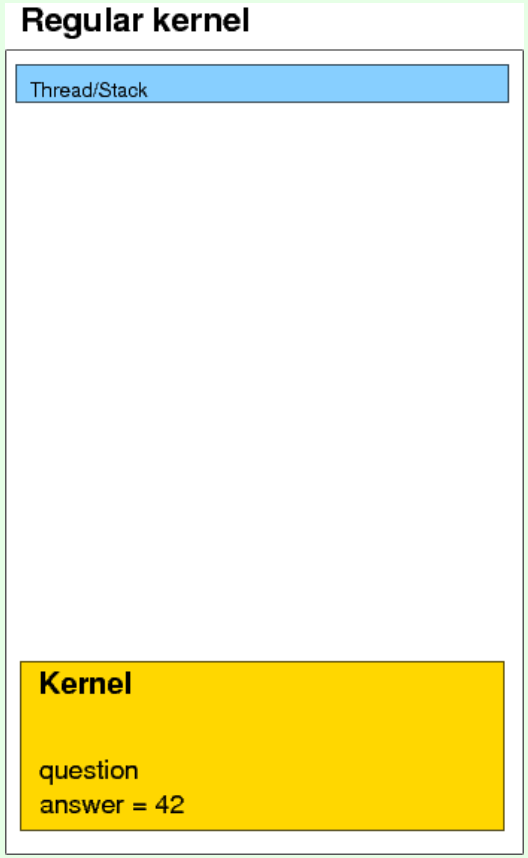
# Teardown problem (cont.ed)

- do we understand teardown yet?
- per network stack subsystem (TCP, UDP, legacy IP, IPv6, link-layer, . . . ) cleanup and shutdown?
- checkpoint broadcast events: ”stop processing new connections”, ”cleanup”, ”really go away”?

# Step by step walkthrough - step 0

Just a global:

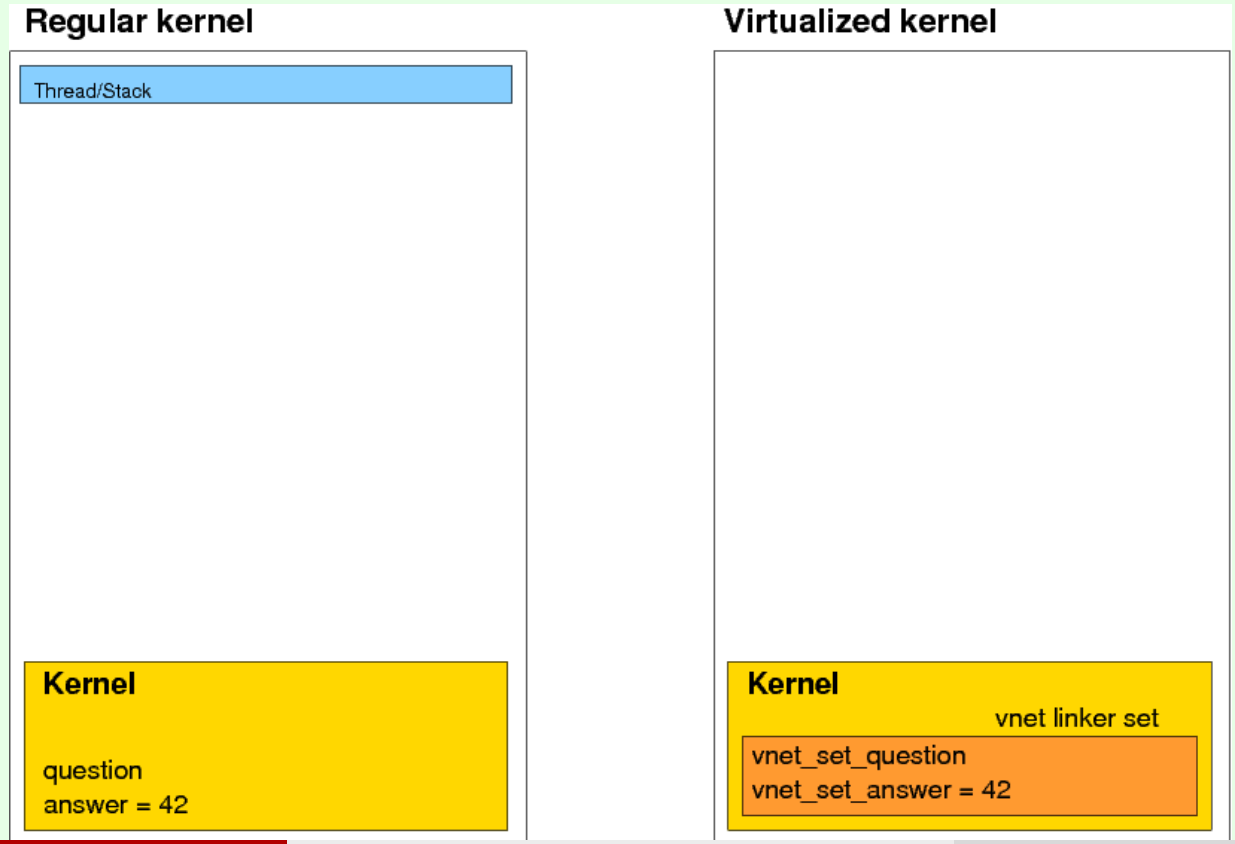
```
static int question;
int answer = 42;
```



# Step by step walkthrough - step 1

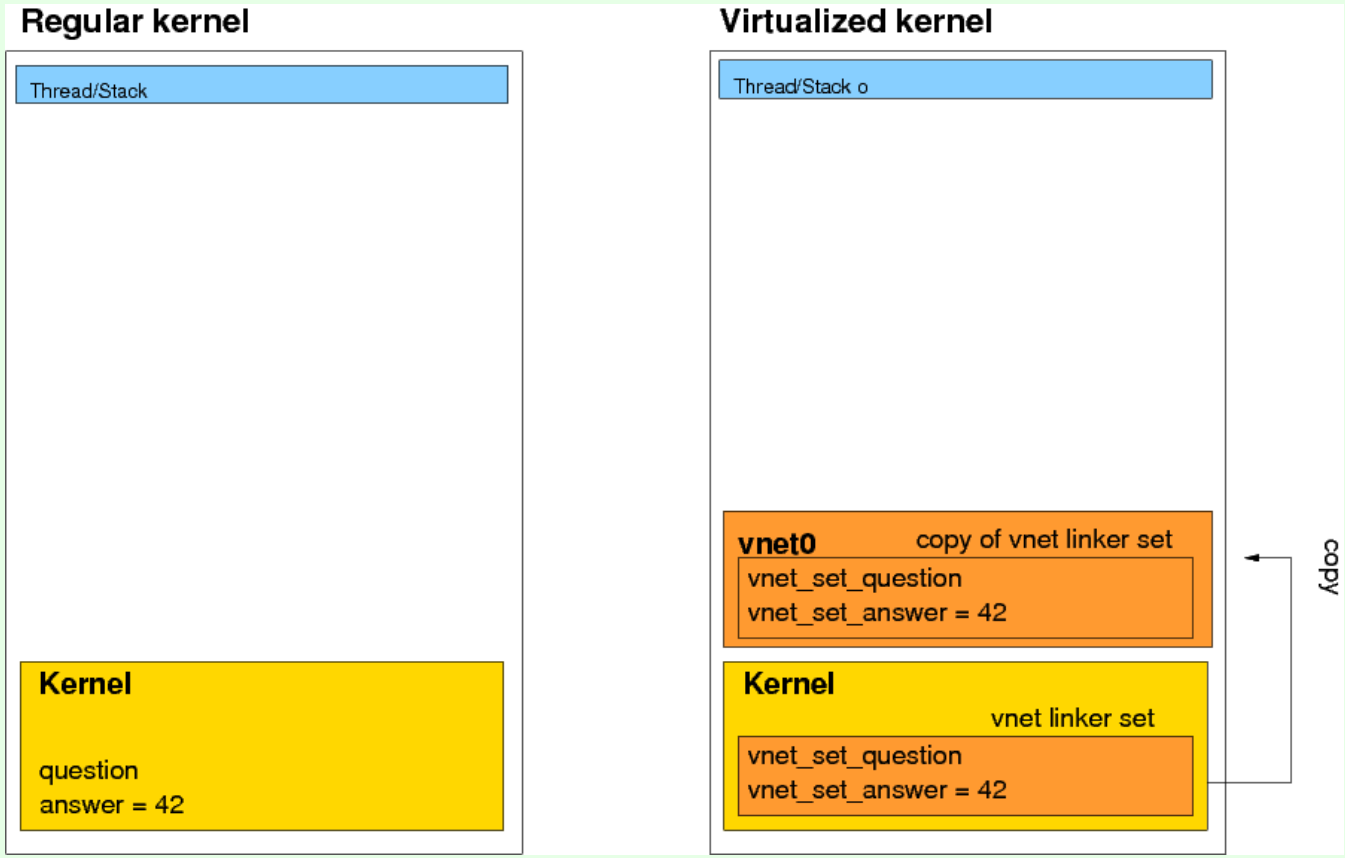
Linker set (own ELF section) - our master copy,

```
static VNET_DEFINE(int, question);
VNET_DEFINE(int, answer) = 42;
#define V_question  VNET(question);
#define V_answer    VNET(answer);
```



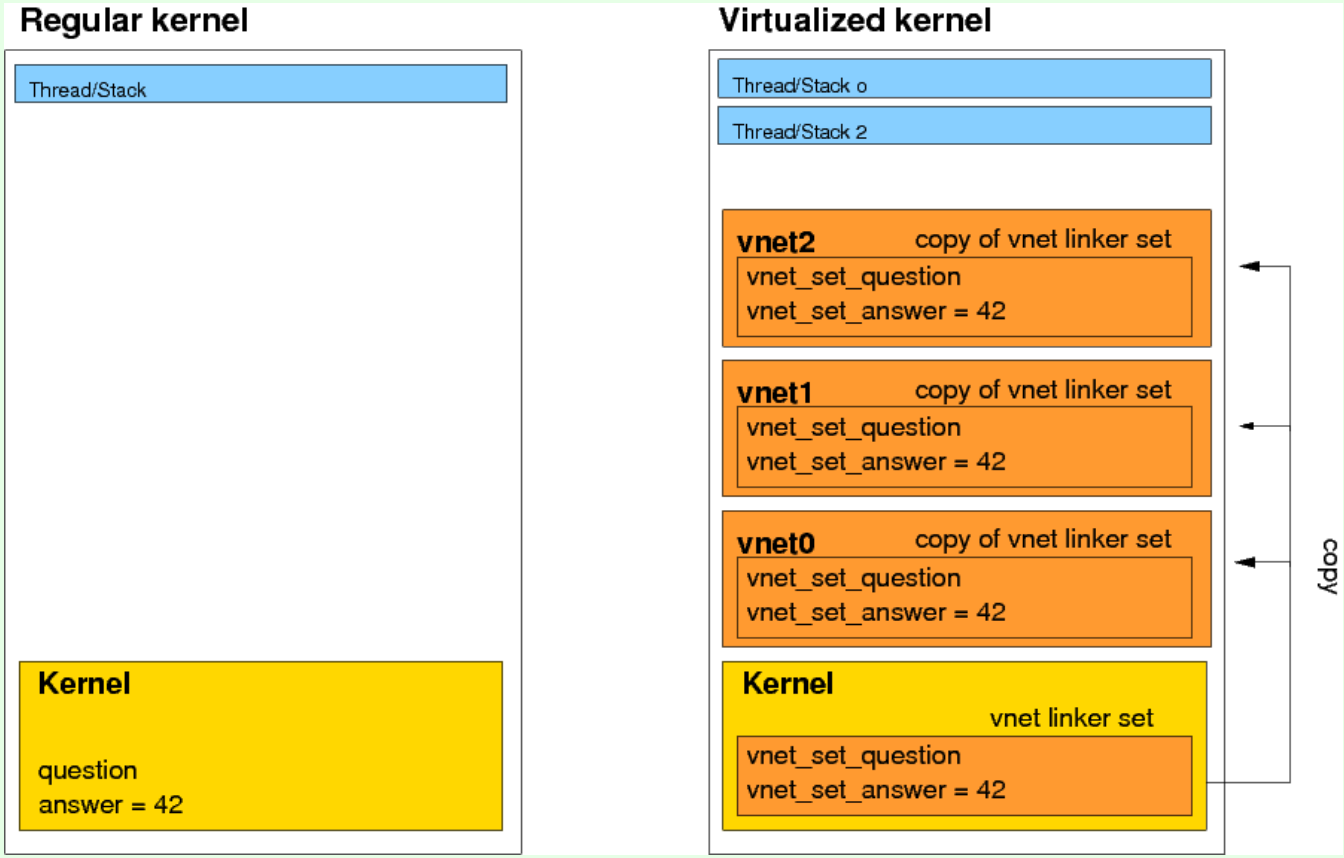
# Step by step walkthrough - step 2

Start base system instance: allocate, copy set, ...



# Step by step walkthrough - step 3

Now start another instance, . . .



# Summary

- Prototype operates with increasing stability and little performance overhead.
- Try 8-STABLE or CURRENT.
  EXPERIMENTAL. You have been warned.
- Goal production-quality VIMAGE in 9.x.

# Modular Network Stack - Motivations

- no INET6
- no INET, no INET6 last year
- no INET but INET6 earlier this year (p4)
- “load” IPsec?

# Why?

- dual stack works just fine.
- this is about the future not the past
- “pfil lock problem”
- make developers aware of how to code

# Code sample

tcp_subr.c

```
 718 #ifdef INET6
 719             if (isipv6)
 720                     (void) ip6_output(...);
 721 #endif /* INET6 */
 722 #if defined(INET) && defined(INET6)
 723             else
 724 #endif
 725 #ifdef INET
 726             (void) ip_output(...);
 727 #endif
```

# Why?

- identify v4/v6 only code
- identify “side code”
  FT, pfil and firewalls, IPsec, . . .
- get rid of same but different code (protosw6)

# Bring them together

Per network layer/subsystem teardown for virtualization
+ clearly separated network layer/subsystem code
+ solve locking and
= be able to kldload layer 3/4 or side code.

(exaggerated simplification)

Do not expect this to happen but keep it in mind when touching code!

# Code, comments and suggestions

```
//depot/user/bz/noinet/...
//depot/user/bz/protosw/...
```

- Comments or suggestions?
- bz@FreeBSD.org or just sit together this evening.

Thanks!