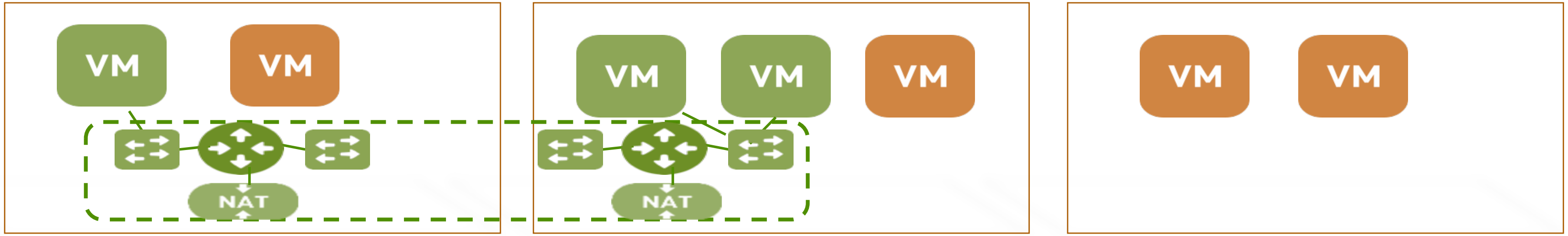
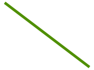
network virtualization BPF and next steps

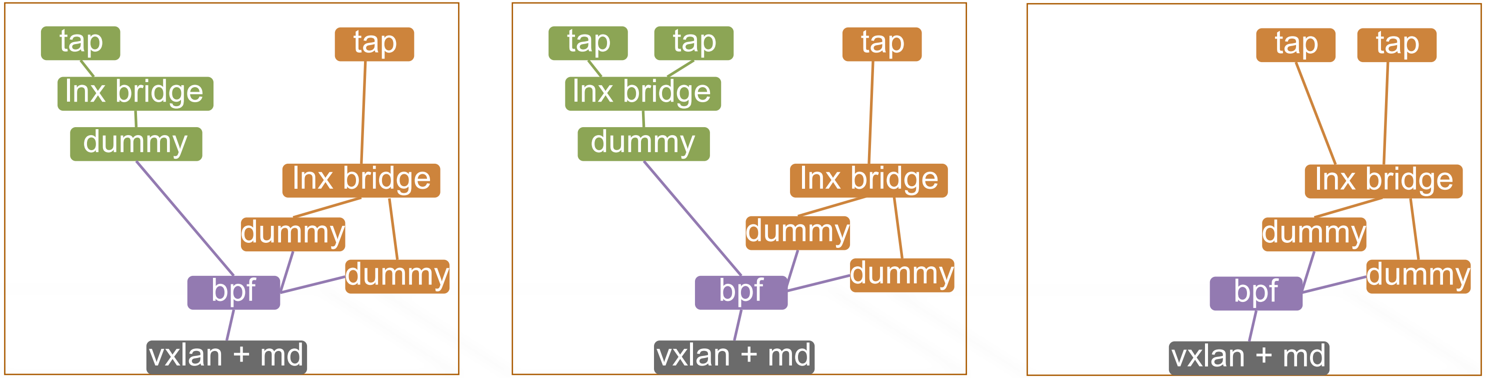
2015 aug 21

Use BPF to build distributed virtual topology out of existing linux bridges and routers



 = BPF

Distributed bridge with BPF



```
/* BPF program RX */
int handle_ingress(struct __sk_buff *skb) {
  struct bpf_tunnel_key tkey = {};
  bpf_skb_get_tunnel_key(skb, &tkey, sizeof(tkey), 0);
  int *ifindex = tunkey2if.lookup(&tkey);
  if (ifindex) {
      skb->tc_index = 1;
      bpf_clone_redirect(skb, *ifindex, 1);
  }
  return 1;
}
```

```
/* BPF program TX */
int handle_egress(struct __sk_buff *skb) {
  int ifindex = skb->ifindex;
  struct bpf_tunnel_key *tkey_p, tkey = {};
  if (skb->tc_index)
      return 1;
  tkey_p = if2tunkey.lookup(&ifindex);
  if (tkey_p) {
      tkey.tunnel_id = tkey_p->tunnel_id;
      tkey.remote_ipv4 = tkey_p->remote_ipv4;
      bpf_skb_set_tunnel_key(skb, &tkey, sizeof(tkey), 0);
      bpf_clone_redirect(skb, tunnel_ifindex, 0);
  }
  return 1;
}
```

- optimize out skb_clone() in bpf_clone_redirect()
  - TC_ACT_REDIRECT (optimize for max performance)
- need persistent maps
  - two fuse implementations exists, but user space daemon that sends/recvs FDs via scm_rights is a ~~showstopper~~
  - potential solutions:
  - add "map_name" to bpf syscall
  - mknod/af_unix like
  - procfs
  - mount –bind /proc/self/fd/5 /my_file
  - bpffs
- stress testing verifier
  - move verifier.c to userspace and apply coverage-guided fuzzing with clang
- cap_sys_admin liberating
  - constant blinding and pointer leak prevention
- redirect to socket
  - avoid netdev per container
  - first step towards arbitrary protocols in bpf

- other bpf news
  - spin_lock removal in act_bpf
  - ksym for JITed programs
  - nft->bpf translator
  - bpf criu
  - bpf in seccomp
  - attaching to tracepoints
  - rhashtable map type
  - idr map type
  - take advantage of new cpu instructions in JIT