using routing domains / routing tables in a production network

Peter Hessler
phessler@openbsd.org

OpenBSD

13 June, 2015

rtable vs rdomain

- rtable
  - alternate routing table, usable with the same interfaces
  - ip addresses cannot overlap
  - multiple rtables can belong to a single rdomain
  - can be used for Policy Based Routing

rtable vs rdomain

- rdomain
  - completely independent routing table instance
  - assign 10.0.0.1/16 a dozen times
  - interfaces can be assigned to only one rdomain at a time
  - how we 'know' which one incoming packets should use
  - rdomains always contain at least one rtable

rdomains

- first added in OpenBSD 4.6, released October 2009
- initially was IPv4 only
- IPv6 support added in OpenBSD 5.5, released May 2014

vrf-lite vs full vrf

- vrf-lite
  - multiple routing domains
  - done by hand
  - very common in smaller enterprises
  - only exists within a single system
- vrf

vrf-lite vs full vrf

- vrf-lite
- vrf
  - also known as 'mpls'
  - requires bgp, ldpd and large networks
  - most frequently used to connect multiple sites in a single network

caveats

- default routes for all the domains!
  - seriously
  - the 'do we have a valid route' check happens *before* pf
  - very common mistake
- debugging can be painful
- which route will be used?
- but, how do we send (some) traffic to a different rdomain?

Simple setup

```
# ifconfig em0 rdomain 1
# ifconfig em0 10.0.0.10/16
# ifconfig lo1 rdomain 1
# ifconfig lo1 127.0.0.1/8
# route -T 1 add default 10.0.0.1
# route -T 1 exec /usr/sbin/sshd
```

Simple setup

```
$ ifconfig em0
em0: flags=88843<UP,BROADCAST,...> rdomain 1 mtu 1500
  lladdr 28:d2:44:ac:5d:59
  priority: 0
  media: Ethernet autoselect
  status: active
  inet 10.0.0.1 netmask 0xffff0000 broadcast 10.0.255.255
$ ifconfig lo1
lo1: flags=28049<UP,LOOPBACK,...> rdomain 1 mtu 32768
  priority: 0
  groups: lo
  inet 127.0.0.1 netmask 0xff000000
```

Simple setup

```
$ netstat -T1 -rnf inet

Routing tables

Internet:
Destination    Gateway            Flags   ~ Prio Iface
default        10.0.0.1           UGS     ~    8   em0
10.0/16        link#1             UC      ~    4   em0
10.0.0.1       28:d2:44:ac:5d:59  UHLl    ~    1   lo0
10.0.255.255   link#1             UHLb    ~    1   em0
127.0.0.1      127.0.0.1          UH      ~    4   lo1
```

Simple setup

pf.conf:

```
pass from any to 10.4.0.4 rtable 2

anchor "cust1.example.com" on rdomain 15 {
    block
    pass proto icmp
    pass proto tcp from any to any port 80
}

pass in on rdomain 2 rdr-to (lo4) rtable 4
pass out from 10.0.0.0/16 to any nat-to (egress) rtable 20
```

production: discovering pitfalls

- route -T 1 exec
- adding rdomain to an interface
- ftp-proxy
- source and destination rdomains matter
- ntpd
- on rdomain

production: discovering pitfalls

- route -T 1 exec
  - originally for testing and hacking, turned out to be very useful
  - recommended method to start a daemon in a second rdomain
  - ...except a few network tools and a limited number of daemons
- adding rdomain to an interface
- ftp-proxy
- source and destination rdomains matter
- ntpd
- on rdomain

production: discovering pitfalls

- route -T 1 exec
- adding rdomain to an interface
  - erases IP address config
  - trunk vs vlan vs parent interface
  - carp
- ftp-proxy
- source and destination rdomains matter
- ntpd
- on rdomain

production: discovering pitfalls

- route -T 1 exec
- adding rdomain to an interface
- ftp-proxy
  - sometimes, you simply want to ftp from *and* to different rdomains
- source and destination rdomains matter
- ntpd
- on rdomain

production: discovering pitfalls

- route -T 1 exec
- adding rdomain to an interface
- ftp-proxy
- source and destination rdomains matter
- ntpd
  - normal solution to needing services in a second rdomain? run the daemon again
  - running a second ntpd to provide time? Holy clock-skew Batman!
- on rdomain

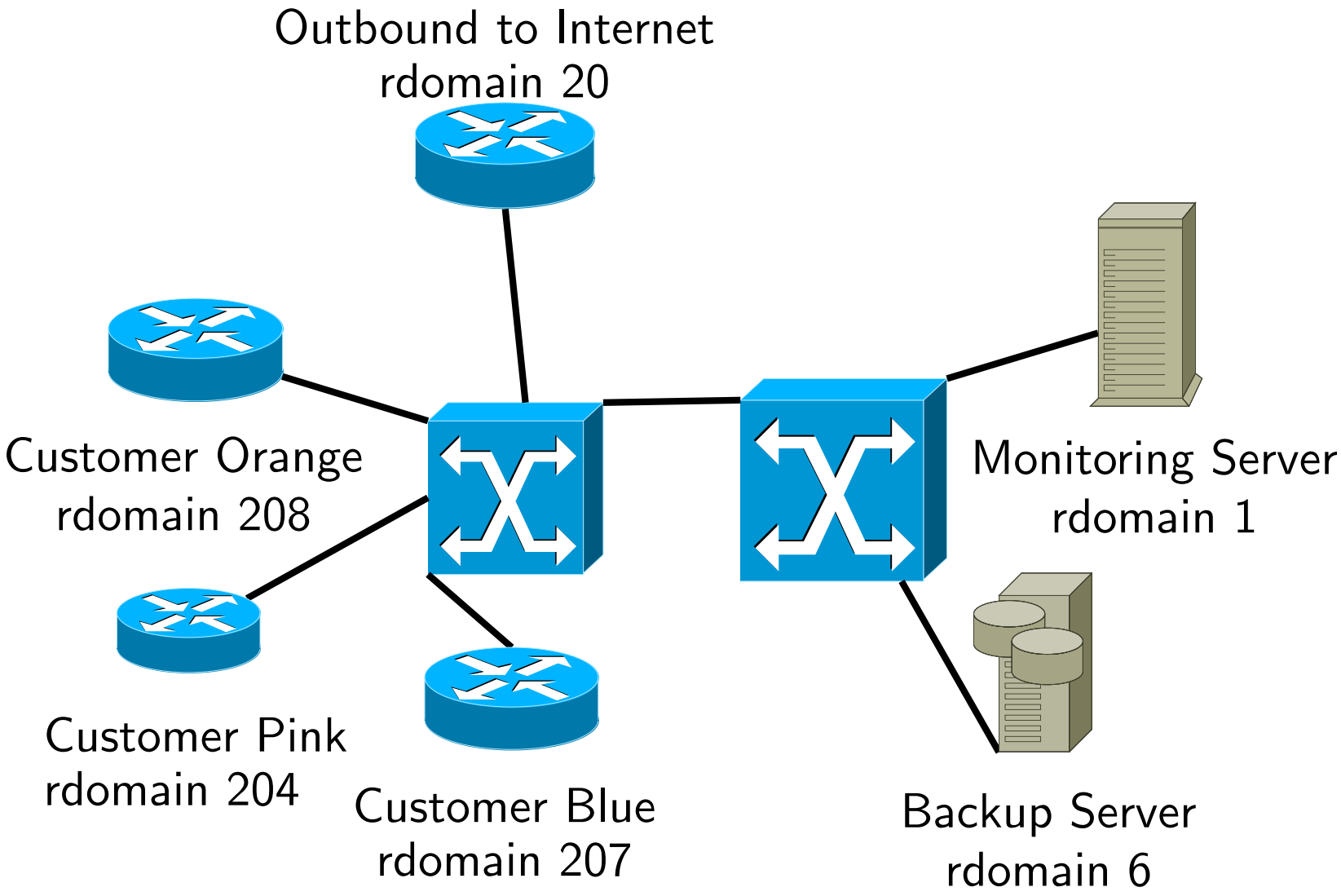
production: discovering pitfalls

- route -T 1 exec
- adding rdomain to an interface
- ftp-proxy
- source and destination rdomains matter
- ntpd
- on rdomain
  - you want to match packets traveling on an rdomain

shared infrastructure (vrf-lite)

- very common
- just a management network
- two rdomains, one pipe
- backup servers
- monitoring
- stuck in the middle with you
- etc

example network: isp



example network: isp customer pink

```
$ /etc/hostname.vlan204
vlan 204 vlandev trunk4
rdomain 204
group pink
inet 203.0.113.1/24
$ /etc/hostname.lo204
rdomain 204
inet 127.0.0.1/8
!/sbin/route -T204 -qn add -net 127 127.0.0.1 -reject
!/sbin/route -T204 -n add default 127.0.0.1 -blackhole
```

example network: isp customer pink

pf.conf:

```
anchor "customer_pink" on rdomain 204 {
    block
    pass in on pink
    pass proto icmp
    pass from $monitor to (pink:network)
    pass proto tcp from (p:net) to $bak port 873 rtable 6
    match out to !(p:net) nat-to $pink_ext_ip rtable 20
}
pass in proto icmp from $monitor to (p:net) rtable 204
```

example network: isp customer pink

```
$ netstat -T204 -rnf inet
Routing tables

Internet:
Destination   Gateway      Flags   Mtu  Prio Iface
default       127.0.0.1    UGBS  32768     8 lo204
127/8         127.0.0.1    UGRS  32768     8 lo204
127.0.0.1     127.0.0.1    UHl   32768     1 lo204
203.0.113/24  203.0.113.1 UC          -     8 vlan204
203.0.113.1   link#14      UHLl        -     1 lo0
203.0.113.255 203.0.113.1 UHb         -     1 vlan204
```

example network: isp customer orange

```
$ /etc/hostname.vlan208
vlan 208 vlandev trunk4
rdomain 208
group orange
inet 203.0.113.1/24
$ /etc/hostname.lo208
rdomain 208
inet 127.0.0.1/8
!/sbin/route -T208 -qn add -net 127 127.0.0.1 -reject
!/sbin/route -T208 -qn add default 127.0.0.1 -blackhole
```

example network: isp customer orange

pf.conf:

```
anchor "customer_orange" on rdomain 208 {
    block
    pass in on orange
    pass proto icmp
    pass from $monitor to (orange:network)
    pass proto tcp from (o:net) to $bak port 873 rtable 6
    match out to !(o:net) nat-to $orange_ext_ip rtable 20
}
pass in proto icmp from $monitor to (o:net) rtable 208
```

example network: isp customer orange

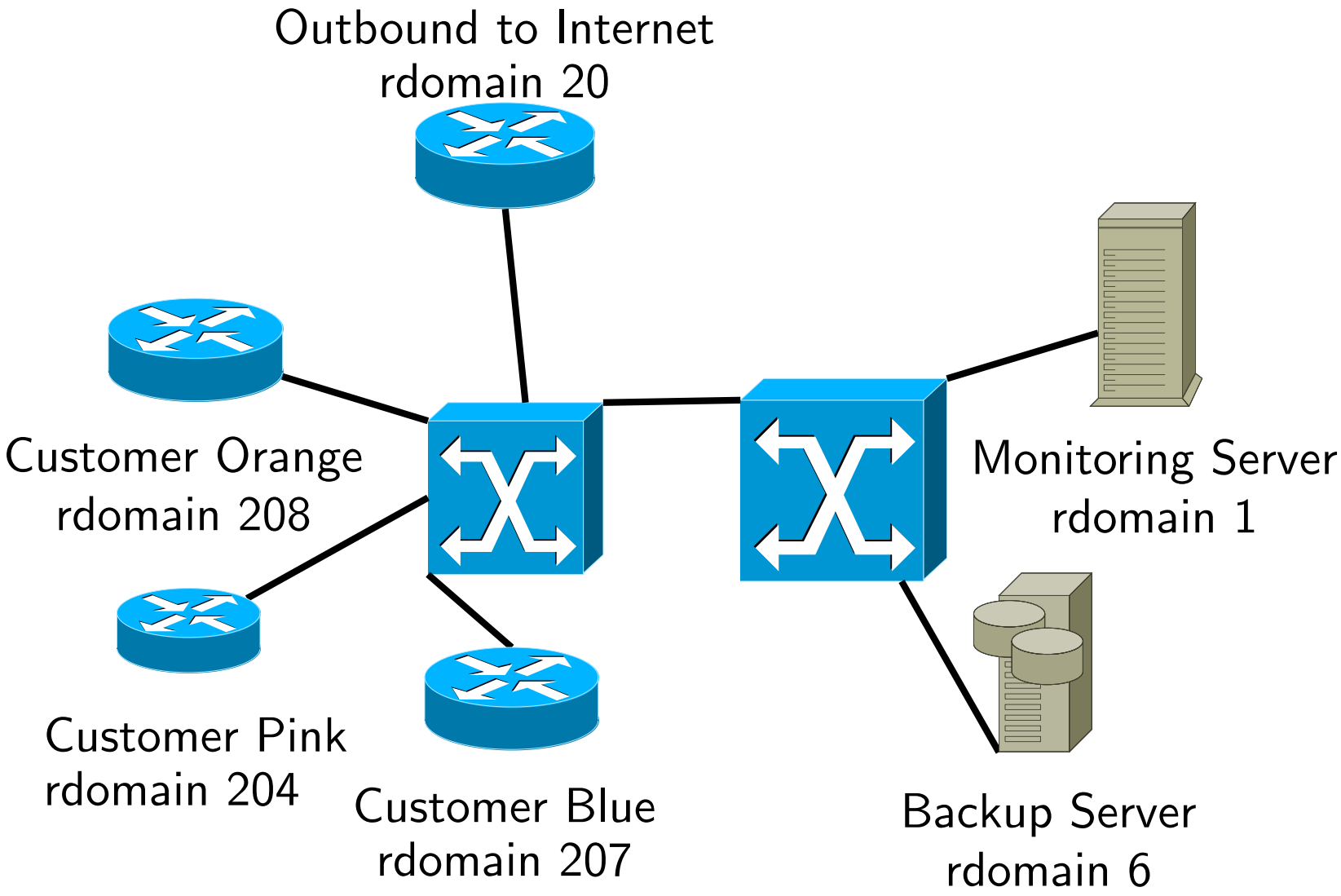
```
$ netstat -T208 -rnf inet
Routing tables

Internet:
Destination   Gateway      Flags   Mtu  Prio Iface
default       127.0.0.1    UGBS  32768     8 lo208
127/8         127.0.0.1    UGRS  32768     8 lo208
127.0.0.1     127.0.0.1    UHl   32768     1 lo208
203.0.113/24  203.0.113.1 UC          -     8 vlan208
203.0.113.1   link#14      UHLl        -     1 lo0
203.0.113.255 203.0.113.1 UHb         -     1 vlan208
```

example network: isp

- use anchors to segment rdomains from each other
- ... *much* easier to write rulesets
- must think about crossing rdomains differently

example network: isp



example network: isp

- pink and orange have conflicting ip addresses
- ... how does monitoring connect to the correct one?
- two options
- #1 put monitoring itself in the appropriate rdomains
- #2 give them unique ips in the monitoring rdomain

example network: isp

pf.conf:

```
anchor "monitoring" on rdomain 1 {
    pass in from any to 198.19.204.0/24 \
        rdr-to 203.0.113.0/24 bitmask rtable 204
    pass in from any to 198.19.208.0/24 \
        rdr-to 203.0.113.0/24 bitmask rtable 208

    pass from any to $bak rtable 1
}
```

full vrf

- ldpd
  - label distribution protocol daemon
  - distributes mpls label mappings
- bgpd
  - distibute our networks over the mpls "tunnel"

full vrf

- read claudio's paper from eurobsdcon 2011

best practices

- default routes for all the things
  - as i said, real common mistake
- pf.conf tricks
- spend extra time in the planning stages

very special thanks

- henning@ for adding the multiple routing table support
- claudio@ writing the code and for putting up with all of my asinine questions when we first tested
- reyk@ for lots of work in bringing this into the tree and funding this via his (former) company

Questions?