

OpenBSD

pf+rdomains create splendid multi-tenancy firewalls

AsiaBSDCon, Tokyo, March 11, 2017

Philipp Bühler <pbuehler@sysfive.com>

Trivia:

Technical lead at sysfive.com GmbH
Hacking computers since 1983
OpenBSD user since 2.7 (2000)
Developer (pf) 2002-2005

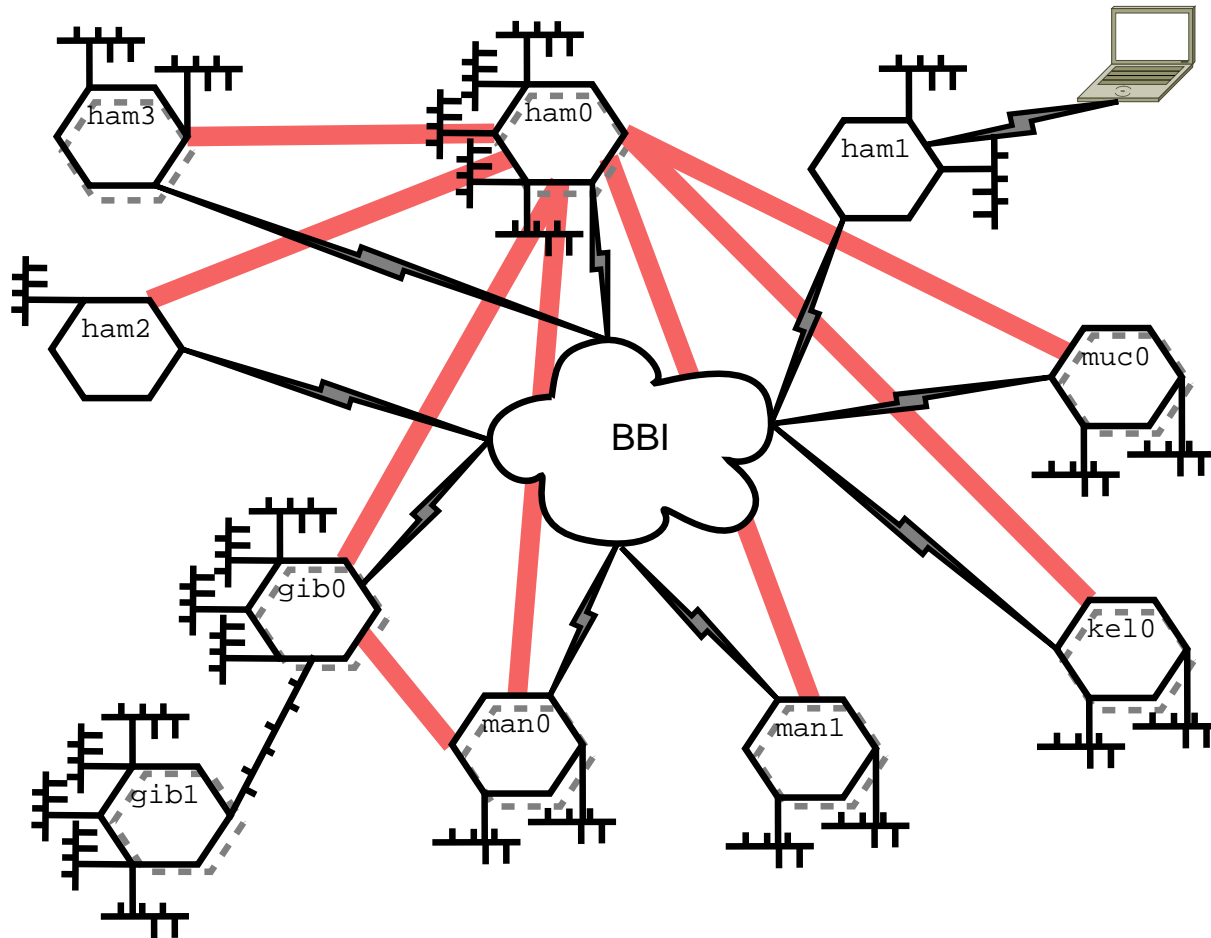
10 people - we're hiring
Apple][e
on i386, amd64, sparc64, macppc
slacked too much



Contents

- Opening
- Ze Problem
 - one and more tenants complexity
 - traditional approaches
- Introduce rdomains
 - interface based
 - tools aware or not
 - pair vs. pf.conf
- Gory details
 - (persistent) configuration
 - pair / pf.conf
 - ntpd
 - pitfalls / debugging
- Test+Automation
 - packer / vagrant, ansible

sysfive.com network



my dia(1) skills are sub par, giving up..

complex network

basic tenant

Network segregation is a must and ends up in numbers if done deeply.

Management — IPMI/KVM, bootstrapping (PXE), monitoring, backup

Services — (rev)proxy, email, ntp, DNS

Application — devel, test/stage, main, DR

Datastorage — RDBMs, NoSQL, LDAP, redis, ..

Others/3rd party — payment services, weather widget, "all the funkyness"

one single real tenant firewall in numbers

```
$ ifconfig | egrep -c '^[a-z]'
```

41

```
$ pfctl -sr | wc -l
```

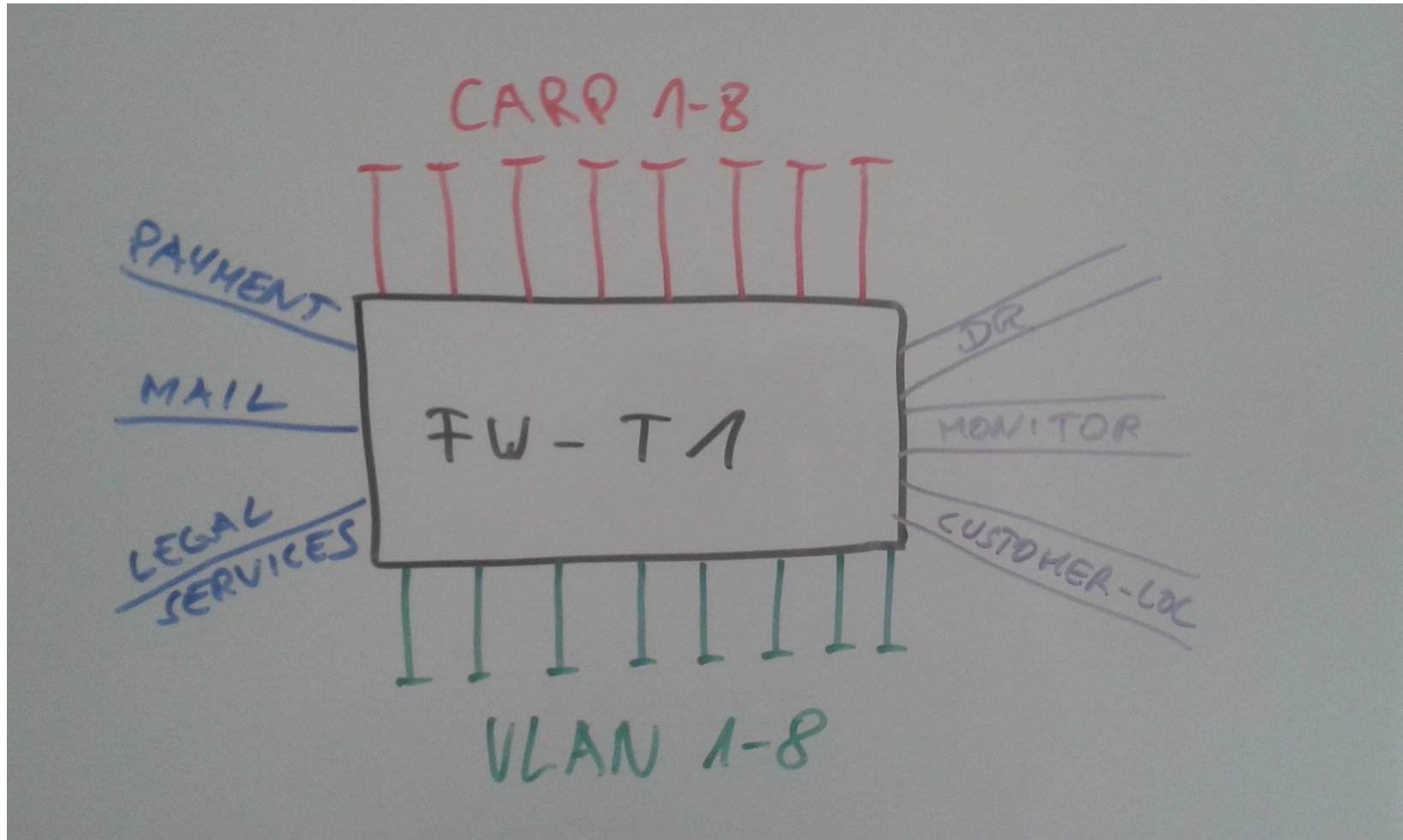
267

```
$ pfctl -a 'relayd/*' -sr | wc -l
```

36

complex network

minimum tenant at ours



traditional approaches (I)

'handcraft'

- overview
- ordering (pf.conf)
- ikwid - I know what I did - you do not
- testing
- panic

templates

- double the above
- exceptions

multiple fw

- what-is-where
- IPv4 scarcity
- rackspace (dah, VMs...)
- time to launch physically

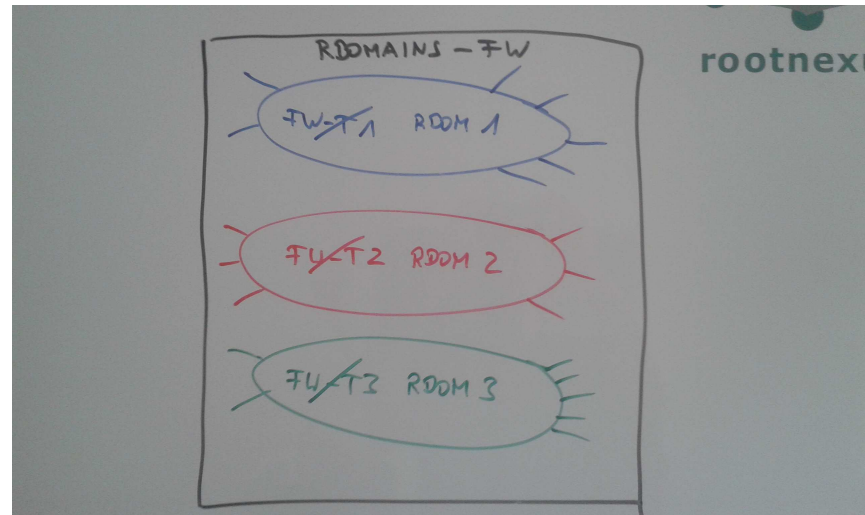
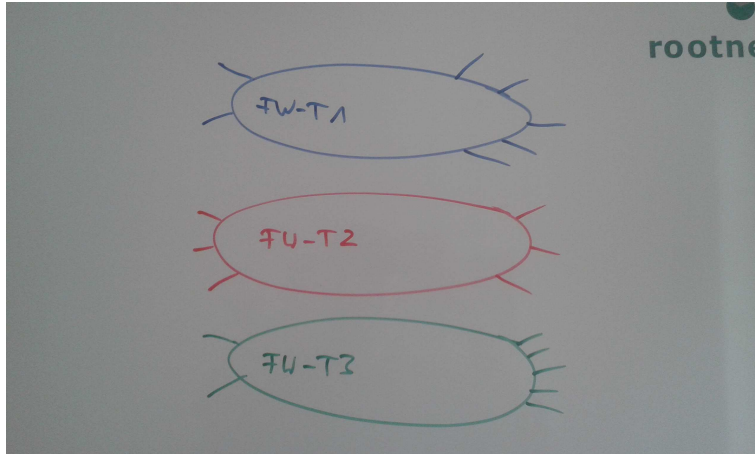
overall problems to be addressed

In general:

- growth
- change
- debug
- quick-fix
- pf.conf ordering matters much
- tenants might buy others
- IP address conflicts (overlap locally and/or VPN)

To consider: rdomains(4) might be very useful when you cannot control (esp. ip-ranges) connected networks

rdomains - many in one



multi routing: each tenant could have its own routing table (and more)

composition: multiple interfaces create one rdomain

tools aware: direct invocation by passing rdomain IDs

daemons: several instances started in different rdomains

pair(4): mesh all the things (connect rdomains)

pf.conf(5): includes, anchors, ..

limits: not again...

aware tools

The following network tools can be invoked with an argument to operate on the given rdomain

netstat(1)

- T <tableid>: show information for given rdomain

route(8)

- T <tableid>: show/operate for given rdomain

- exec: start a process in rdomain (next page)

arp(8) / ndp(8)

- V <tableid>: limit to given rdomain

ping(8)

- V <tableid>: ping from rdomain

traceroute(8)

- V <tableid>: trace a route from rdomain

nc(1)

- V <tableid>: bind socket in given rdomain

ps(1)

- o rtable: adds ID of rtable/rdomain the process runs within

pkill(1)/pgrep(1)

-T <tableid>: limit search/results to given rdomain

tcpbench(1)

-V <tableid>: run benchmark within given rdomain

telnet(1)

-V <tableid>: use given rdomain

ftp-proxy(8)

somewhat via pf(4) tagging — really? ftp?

bgpd(8)/ospfd(8)/ripd(8)/eigrpd(8)/ldpd(8)

rtable/rdomain keywords — out of scope

authpf(8)

can use multiple pf anchors - tricky!

relayd(8)

(details pages)

rcctl(8)/rc.d(8)

(details pages)

ntpd(8)

(details pages)

ifconfig(8)/hostname.if(5)

(details pages)

not aware tools

route exec

Any other tool or daemon can be started (multiple) within a given rdomain via route(8).

```
route -T 23 exec ikev1 -ddvfvf /etc/ikev1.conf.23
route -T 42 exec ikev1 -ddvfvf /etc/ikev1.conf.42
```

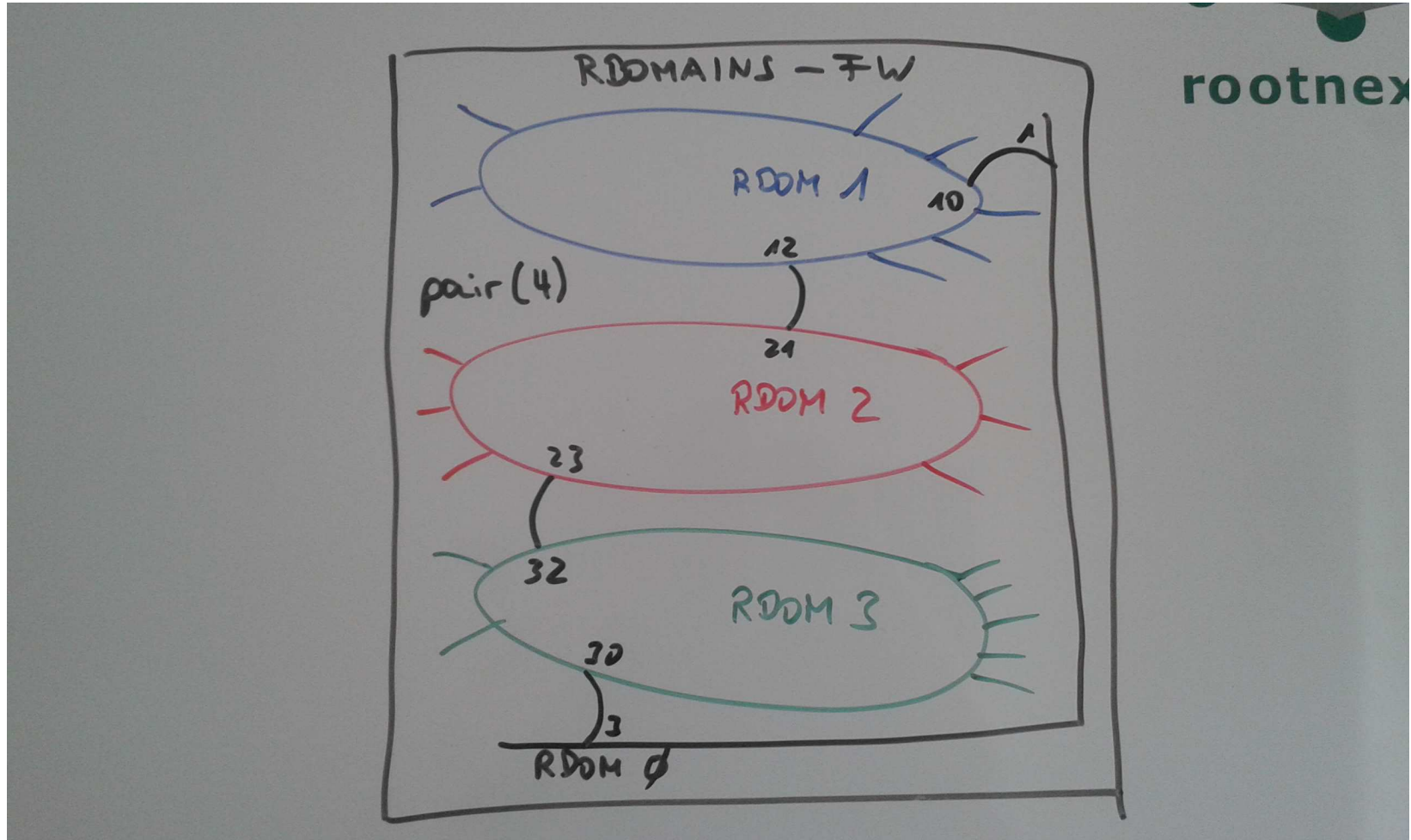
Sidenote: this will need two enc(4) in the corresponding rdomain, too. No choosable socket as of yet, can only use ikev1 on "last one".

Pitfall: daemons operating on shared/global information can still mess up each other, e.g. ntpd(8).

limits

- maximum of 256 routing domains
- any carp(4) devices must be in the same rdomain as its 'carpdev' - see Pitfalls(II)
- some daemons not yet fully aware/usable, but should so

pair(4)



pf.conf syntax

on / rtable

```
pass in on rdomain 21 from $tenant-app to $tenant-email  
#  
pass in from $backup to <tenant1> rtable 21
```

anchors

```
anchor "tenant1.21" on rdomain 21 {  
    block  
    pass out proto tcp from any to any port { 80 443 }  
}  
  
anchor "tenant2.41" on rdomain 41 {  
    block  
    match out to any nat-to $ext-41-ip rtable 0 tag TENANT_41  
    pass out tagged TENANT_41  
}
```

hostname.if (I)

physical(4)/vlan(4)/carp(4)

Creating rdomains is done by assigning 'rdomain N' to an interface, naturally this can be done in hostname.if(5).

Hint: put 'rdomain' **before** any address configuration (inet/inet6).

```
/etc/hostname.em0:
rdomain 0
inet 10.40.40.254/24

/etc/hostname.vlan41:
description "gw-vlan-41"
vlan 41 vlandev em2
rdomain 41
inet 10.40.41.1/24

/etc/hostname.carp1
description "gw-carp-1"
rdomain 0
vhid 1
pass onetwomany
carpdev em0
inet 10.60.5.1/24

/etc/hostname.pair21
description "gw-pair-21"
rdomain 21
inet 10.200.21.2/30
patch pair0
#!/sbin/route -T 21 -n add default 10.200.21.1
```

pair(4) (I)

With pair(4) and route(8) one can interconnect rdomains. Being virtualized ethernet it needs two endpoints that are then patched to each other:

```
$ doas ifconfig pair0 rdomain 0 10.200.21.1/30 up
$ doas ifconfig pair21 rdomain 21 10.200.21.2/30 up
$ doas ifconfig pair0 patch pair21
```

The pair(4) devices can be added to a bridge(4), too. STP dragons around.

To persist the above setup:

```
/etc/hostname.pair0:
description "gw-pair-0"
rdomain 0
inet 10.200.21.1/30

/etc/hostname.pair21:
description "gw-pair-21"
rdomain 21
inet 10.200.21.2/30
patch pair0
!/sbin/route -T 21 -qn add default 10.200.21.1
```


pair(4) (II)

finalized setup would look like this:

```
pair0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    lladdr fe:e1:ba:d0:4a:8a
    description: gw-pair-0
    index 7 priority 0 llprio 3
    patch: pair21
    groups: pair
    media: Ethernet autoselect
    status: active
    inet 10.200.21.1 netmask 0xffffffff broadcast 10.200.21.3
pair21: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> rdomain 21 mtu 1500
    lladdr fe:e1:ba:d1:ca:a3
    description: gw-pair-21
    index 8 priority 0 llprio 3
    patch: pair0
    groups: pair
    media: Ethernet autoselect
    status: active
    inet 10.200.21.2 netmask 0xffffffff broadcast 10.200.21.3
```

Destination	Gateway	Flags	Refs	Use	Mtu	Prio	Iface
default	10.200.21.1	UGS	0	1	-	8	pair21
10.20.21/24	10.20.21.1	UC	0	0	-	4	vlan21
10.20.21.1	08:00:27:5b:02:b2	UHLl	0	0	-	1	vlan21
10.20.21.255	10.20.21.1	UHb	0	0	-	1	vlan21
10.200.21.0/30	10.200.21.2	UC	1	0	-	4	pair21
10.200.21.1	link#8	UHLc	1	1	-	4	pair21
10.200.21.2	fe:e1:ba:d1:ca:a3	UHLl	0	0	-	1	pair21
10.200.21.3	10.200.21.2	UHb	0	0	-	1	pair21

rcctl(8) / rc.d(8)

For automated startup, rc.d(8) has 'daemonname_rtable=<N>' support (default to 0). Consequently this can be configured using rcctl(8):

```
$ doas rcctl set httpd status on
$ doas rcctl set httpd rtable 21

$ doas rcctl get httpd
httpd_class=daemon
httpd_flags=
httpd_rtable=21
httpd_timeout=30
httpd_user=root

$ doas rcctl start httpd
httpd(ok)

$ ps auxo rtable | grep http # note last column
www      46042  0.0  0.7   744  1740 ??   Sp      4:43PM      0:00.00 httpd: server (h      21
[...]
```

The daemon was started like a manual 'route -T 21 exec httpd'.

ntpd(8)

- run only one ntpd(8) or the clock will go very funky - VERY
- 'server(s)' go to rdomain 0 (or whichever it uses to reach them)
- 'listen' can be repeated for every rdomain needed, flagged with 'rtable N'

```
server de.pool.ntp.org
listen 127.0.0.1
listen 127.0.0.1 rtable 69
listen 10.20.21.1 rtable 21
listen 10.40.41.1 rtable 41
```

pf.conf(5)

before / after

all-in-one:

```
match out on $if_ext inet proto { icmp udp tcp } \
    from <net_tenant1> to !<rfc1918> nat-to $nat_tenant1

match out on $if_ext inet proto tcp from any \
    to !<rfc1918> port 25 nat-to $nat_tenant1_mail

match out on $if_ext inet proto { icmp udp tcp } \
    from <net_tenant2> to !<rfc1918> nat-to $nat_tenant2
```

rdomains:

```
anchor "tenant1" on rdomain 21 {

    match out on $if_ext inet proto { icmp udp tcp } \
        from <net_tenant1> to !<rfc1918> nat-to $nat_tenant1

    match out on $if_ext inet proto tcp from any \
        to !<rfc1918> port 25 nat-to $nat_tenant1_mail
}

anchor "tenant2" on rdomain 41 {

    match out on $if_ext inet proto { icmp udp tcp } \
        from <net_tenant2> to !<rfc1918> nat-to $nat_tenant2
}
```

pf.conf(5)

before / after

all-in-one /etc/pf.conf:

```
set skip on lo0 enc0 enc1
set optimization aggressive

block in from $tenant1 to $tenant2 # watch out XXX!

pass from $tenant1 to any nat-to $tenant1_public

match out from $tenant2 to any nat-to $tenant2_public #on request call 3am

match out from any to any nat-to (egress)
```

rdomains with includes /etc/pf.conf:

```
include "/etc/pf/globals.conf"
include "/etc/pf/management.conf"
anchor "tenant1" on rdomain 21 {
    include "/etc/pf/tenant1.conf"
}
anchor "tenant2" on rdomain 41 {
    include "/etc/pf/tenant2.conf"
}
# EOF
```

Pitfalls (I)

- route lookup: if there's none, nothing will happen (even with pf.conf), some sane default if not using pair(4):

```
/etc/hostname.vlan21:  
up  
rdomain 21  
!/sbin/route -T21 -qn add -net 127 127.0.0.1 -reject  
!/sbin/route -T21 -qn add default 127.0.0.1 -blackhole
```

- rdomains(4): rdomains rtable is not removable so look out for 'remains' when playing (adding/removing/..) around
- ifconfig(8): when adding (changing) an rdomain, the inet or inet6 configuration will be removed
- bridge(4): when adding pair(4) it's "easy" to create a loop, use 'stp' on the pair members

```
$ doas ifconfig bridge0 add pair0 add pair21 stp pair0 stp pair21 up
```

- ping(8), traceroute(8): careful with 'bind-address' when overlapping IP-networks around

Pitfalls (II)

- `iked(8)`: no choosable socket (but `ikectl(8)` can..)
- `authpf(8)`: tricky to use different config files
- `isakmpd(8)`, `iked(8)`: needs multiple `enc(4)`, one per `rdomain(4)` or no IPsec packets will flow
- Use 'group' in `ifconfig(8)/hostname.if(5)` to lookup quickly, e.g. in `pf.conf(5)`
- `carp(4)`: by its nature, it must be in the same `rdomain` as its parent 'carpdev'. Possible virtual interfaces that can be inserted for abstraction: `vlan(4)` or `vether(4)+bridge(4)` as below:

```
$ doas ifconfig vether51 rdomain 51 10.51.1.2 up
$ doas ifconfig vether52 rdomain 52 10.52.1.2 up
$ doas ifconfig bridge0 add em1 add vether51 add vether52 \
    stp em1 stp vether51 stp vether52 up
$ doas ifconfig carp51 rdomain 51 carpdev vether51 10.51.1.1 up
$ doas ifconfig carp52 rdomain 52 carpdev vether52 10.52.1.1 up
```

Output: <https://gist.github.com/double-p/d3a20fded7e8ced30735705e1dfea5c4>

relayd(8)

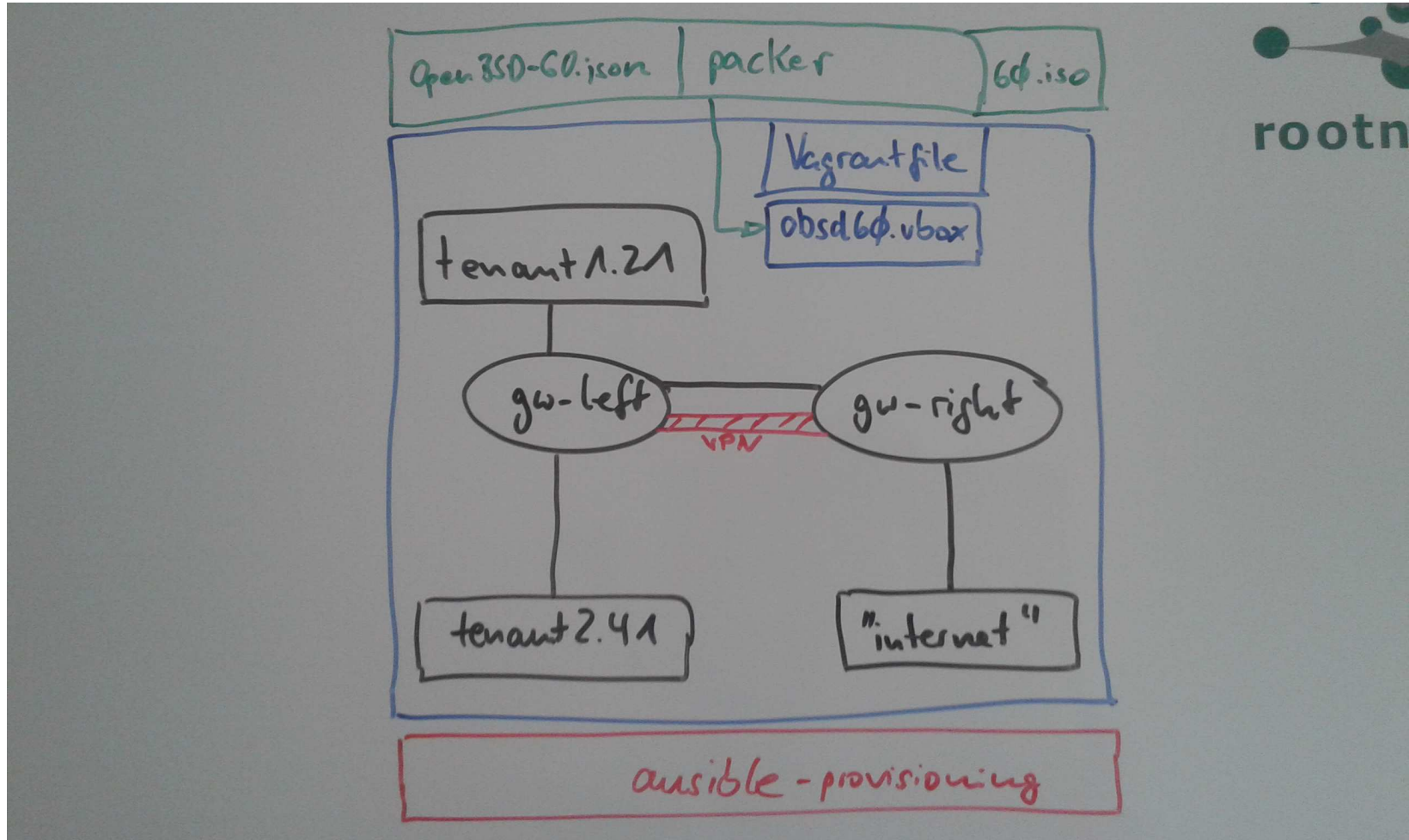
- Currently: Limited capability of doing rdomains(4):
 - no choosable anchorname - be very careful in redirects
 - no choosable socket - relayctl(8) will work on the last started instance
 - 'routers' - can insert routes in given rtable
 - 'redirects' - only uses 'on rdomain' in which relayd(8) was started.
- Future?: diff on github, enabling:
 - -s : choose a socketname
 - -a : choose an anchorname instead of 'relayd/*'
 - rtable in 'redirect' : lookup up destination in choosable rtable

```
$ cat /etc/relayd.conf
table <local> { 127.0.0.1 }
redirect "foo" { listen on 127.0.0.1 port http
forward to <local> port 8000 check icmp rtable 21 }

$ doas route -T 21 exec relayd -a rdr21 -s /var/run/relayd21.sock

$ doas pfctl -a 'rdr21/*' -sr
anchor "foo" all {
  pass in quick on rdomain 21 inet proto tcp from any to 127.0.0.1 port = 80 flags
  S/SA keep state (tcp.established 600) rtable 21 rdr-to <foo> port 8000 round-robin
}
```


Testbed layout



Tooling

- packer: create 'Vagrant box', baseline image
- Vagrant: provide, start above image in distinct VMs based on VirtualBox, VMware,... and configure networks
- ansible: provision /etc/hostname.if and more

packer

```
{
  "variables": {
    "hostname": "",
    "architecture": "",
    "keyboard_layout": "",
    "root_password": "",
    "timezone": "",
    "ntp_pool": ""
  },
  "builders": [
    {
      "type": "virtualbox-iso",
      "name": "vbox-obsd60-amd64",
      "boot_command": [
        "S<enter>",
        "cat <<EOF >>install.conf<enter>",
        "Choose your keyboard layout = {{user 'keyboard_layout'}}<enter>",
        "System hostname = {{user 'hostname'}}<enter>",
        "Password for root = {{user 'root_password'}}<enter>",
        "Start sshd(8) by default = yes<enter>",
        "Do you expect to run the X Window System = no<enter>",
        "Setup a user = no<enter>",
        "Allow root ssh login = yes<enter>",
        "What timezone are you in = {{user 'timezone'}}<enter>",
        "Location of sets = cd0<enter>",
        "Set name(s) = -comp*.tgz -game*.tgz -x* +bsd.mp<enter>",
        "Continue without verification = yes<enter>",
        "SHA256.sig. = yes<enter>",
        "Time appears wrong = yes<enter>",
        "EOF<enter>",
        "install -af install.conf && reboot<enter>"
      ],
    }
  ]
}
```

```
"boot_wait": "15s",
"disk_size": 5120,
"guest_os_type": "OpenBSD_64",
"iso_url": "./ISO/install60.iso",
"iso_checksum": "a9b6b812cb456f11ba4362e232efdd5f0c47868f4281e383ecee77541b54498c",
"iso_checksum_type": "sha256",
"ssh_username": "root",
"ssh_password": "vagrant",
"ssh_wait_timeout": "10m",
"shutdown_command": "echo '/sbin/halt -p' > shutdown.sh; sh 'shutdown.sh'",
"guest_additions_path": "disable",
"virtualbox_version_file": ".vbox_version",
"vboxmanage": [
  [ "modifyvm", "{.Name}", "--memory", "64" ],
  [ "modifyvm", "{.Name}", "--cpus", "1" ]
]
},
"provisioners": [
{
  "type": "shell",
  "environment_vars": [
    "PKG_PATH=http://artfiles.org/openbsd/snapshots/packages/%a/"
  ],
  "inline": "pkg_add -Iz sudo-1.8 python-2.7"
}
],
"post-processors": ["vagrant"]
}
```

Bottom line: boot install60.iso, write out install.conf, autoinstall(8) that, add sudo and python and after shutdown, create a box-file that can be used with Vagrant.

Vagrant

```
inlprv=<<<SCRIPT
route delete default
[ -z "$1" ] || route add default $1
echo 'export PS1="\h:\W # "' >> .profile
SCRIPT
```

```
Vagrant.configure("2") do |config|
  config.vm.guest = :openbsd
  config.ssh.shell = "ksh -l"
  config.ssh.username = "root"
  config.ssh.password = "vagrant"
  config.vm.box = "obsd60"
  config.vm.synced_folder ".", "/vagrant", disabled: true

  config.vm.define "tenant1.21" do |v|
    v.vm.network :private_network, ip: "10.20.20.21"
    v.vm.hostname = "tenant1"
    v.vm.provider :virtualbox do |vb|
      vb.customize ["modifyvm", :id, "--memory", 128]
    end
    v.vm.provision "shell", inline: $inlprv, args: "10.20.20.254"
  end

  config.vm.define "rdomain-gw-left", primary: true do |v|
    v.vm.network :private_network, ip: "10.20.20.254"
    v.vm.network :private_network, ip: "10.40.40.254"
    v.vm.network :private_network, ip: "10.0.123.254"
    v.vm.hostname = "rdomain-gw-left"
    v.vm.provider :virtualbox do |vb|
      vb.customize ["modifyvm", :id, "--memory", 256]
      vb.gui = true;
    end
    v.vm.provision "shell", inline: $inlprv
  end
end
```

Sidenote: the 'private_network' will be connected if they share the same prefix, e.g. 10.20.20.21 in tenant1.21 can directly reach 10.20.20.254 on rdomain-gw-left.

ansible

- inventory: VM or physical hosts
- tasks: actual work
- template: to create hostname.if(5)

Closing wrapup

- Complex setups are difficult to operate without fallouts
- Three times the above if it's 3 in the morning
- rdomains keep tenants separated without need for a ruleset
- rulesets can be separated by includes and/or anchors
- separated setups are easier to operate, even from "junior" staff
- Scared? Test in virtualized environment to remove breakage
- Testing creates confidence in your setup
- Automation + provisioning saves the day (and weekend)
- Check back for further development of 'smtf' on github

Input + Thanks

Peter Hessler

for the talks, experiences and help in rdomains

Ingo Schwarze

for helping out with roff/gpresent to create this doc

OpenBSD developers

for adding this and OpenBSD itself

sysfive.com GmbH

for giving enough working hours to get this done

Some more to learn from - and foundations for this talk

manpages (kid you not) • `pf.conf(5)`, `route(1)`, `rdomain(4)`, `pair(4)`, ..

<https://www.youtube.com/watch?v=BizrC8Zr-YY> • Peter Hessler on rdomains (BSDCan2015)

<http://www.openbsd.org/papers> • All OpenBSD presentations, including the above talks

<http://github.com/double-p/smtf> • This talk and paper, relayd patches and automation environment

https://www.youtube.com/watch?v=uFeEP_hzFN0 Reyk Floeter on vxlan/cloud networks

<https://www.youtube.com/watch?v=mN5E2EYJnrw> David Gwynne on pf/pfsync