Puffalanche - OpenBSD by the busloads

OpenBSD and Vagrant: make (auto)install by the busloads easy

Philipp Bühler <pb@sysfive.com>

sysfive.com portfolio

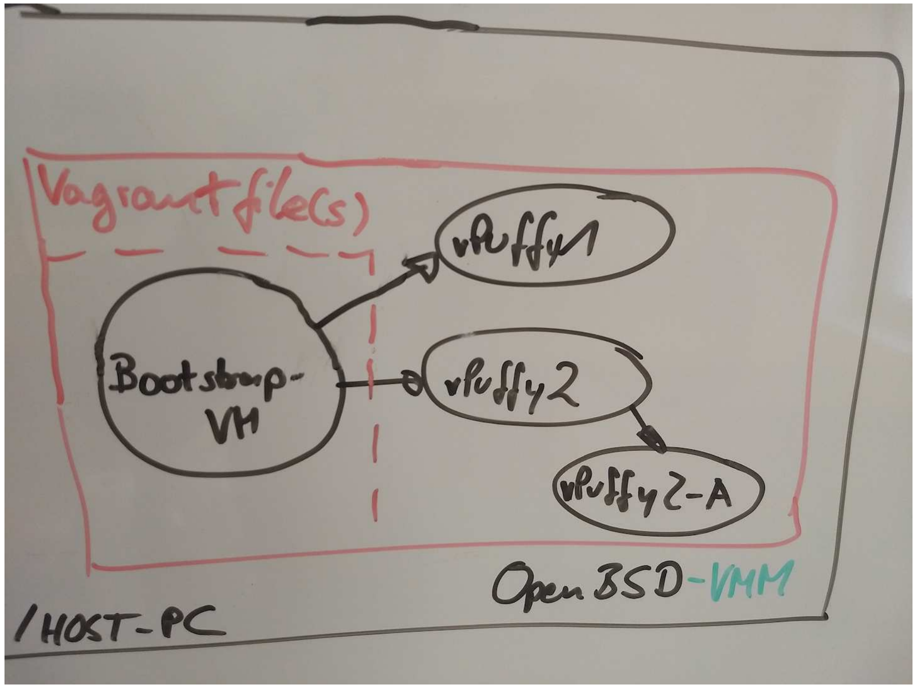
- Continous system and application operation
- Incident, problem, disaster response
- Collaborations with Providers, Developers, Services and QA
- Hybrid cloud provisioning
- cost efficient scaling on commodity HW
- scale out to AWS/RS/GCE

- Service availability independent of solution scenario
- migrate from or to private/public cloud or own HW
- robust, scalable technology portfolio
- continuous improvements in security and server architecture
- coherent provisioning across platforms (dev/stage/live)
- vendor/provider independence, OSS focus
- ... and we’re hiring.

Solving what?

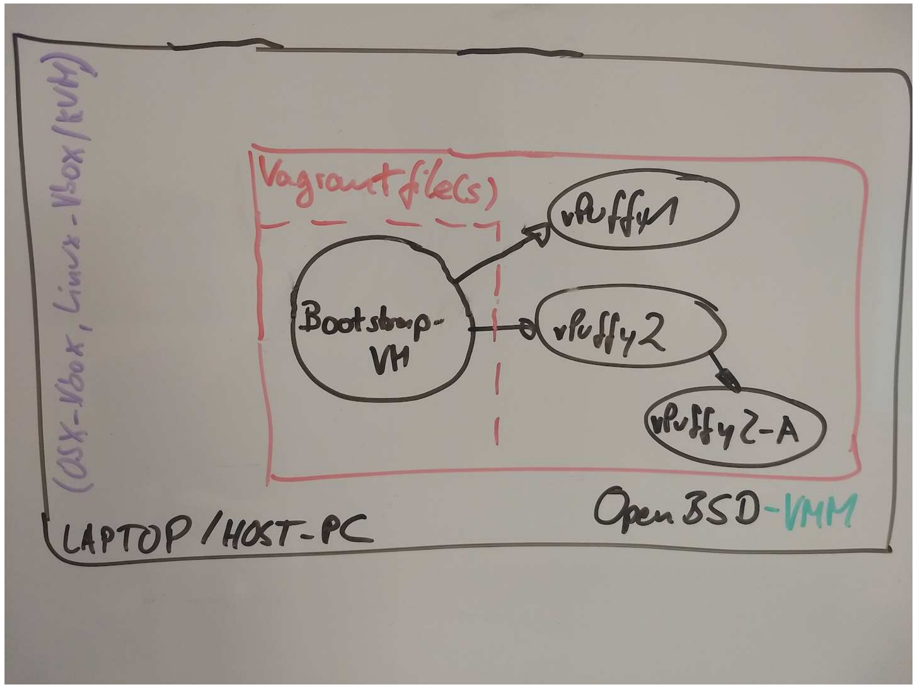
- Run multiple OpenBSD VMs on OpenBSD - w/o dealing with vm.confs at all
- make inter/intra-networking "just work" - no bridge(8) "hassle"
- use the same(!) Vagrantfile on OpenBSD/OSX/Linux (Windows?) to reach the same resulting VM package/network/setup
- develop and TEST autoinstall at 30,000ft (or -50)
- create reproducable installs - even "me so unique" ones
- and also retrospective (live->test)

Puffy boxed in OpenBSD (Dev#1)



- Bootstrap-VM: might be based on manual install -> better in packer
- vPuffy1+2: auto-install from B-VM directly
- vPuffy2a: auto-install (PXE via dhcrelay on Puffy2)

Puffy boxed in Linux/OSX/.. (Dev#2-n)

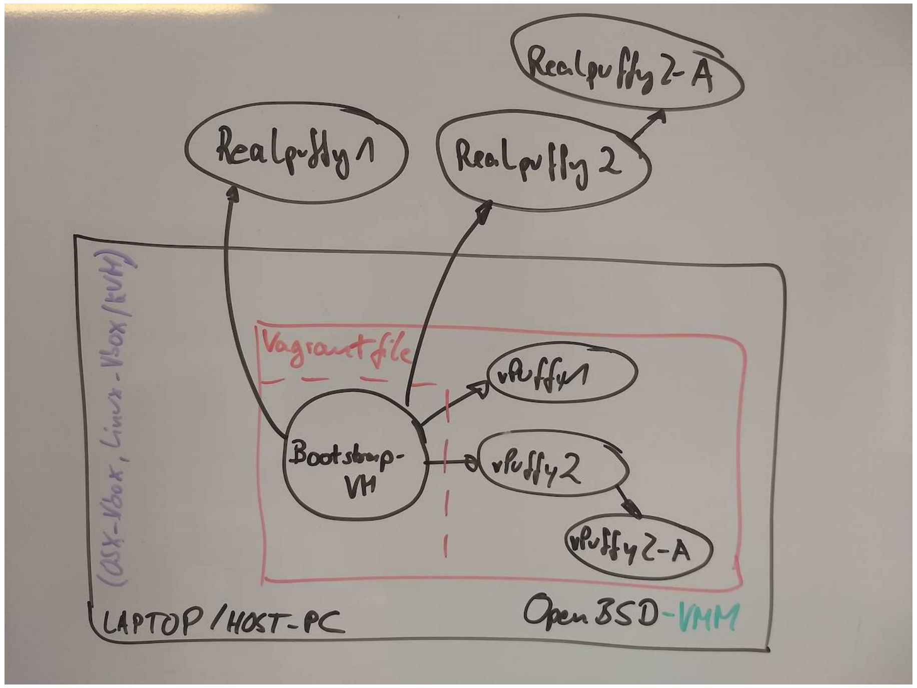


Just run the SAME "infrastructure" on

- OSX (VMware/Virtualbox)
- Linux (Virtualbox/libvirt)
- Cloud (AWS/GCE/..)
- basically everything that Vagrant supports (given enough capabilities)

Infrastructure going on a trainride or being airborne.

Puffy BREAKOUT to physical.



- Not impressed so far? Let’s go physical..
- Run the very SAME "infrastructure" on REAL puffymachines
- Test virtually, use results for:
- Confidence in rollouts
- debug problems on Laptop, roll-out solution to Realpuffy

What did I need to work on? (ongoing)

- OpenBSD: VMM PXE enabled BIOS (hi Mike)
- Vagrant "Core" (plugin-loader) (+port)
- OpenBSD’s VMM as a Vagrant provider-plugin (+gem port)
- [Ruby development tools - only for plugin development (BUILD.md in repo)]
- integrated vether(8)/bridge(8)/dhcpd(8) setup (VM to VM)
- deeper knowhow on autoinstall(8) features
- [installer enhancements (pre-install.sh)]

Groundwork is done, but open points:

- Better automation in network lookup (no magic numbers/assumptions).
- Gem of vagrant-openbsd
- ports(7) of Vagrant and vagrant-openbsd. First one "complicated" for me, second should be a breeze after having a Gem.

What’s already around?

- non-published PXE BIOS
- bundle(1) Vagrant 2.1 (but likely works with 1.5+)
- Vagrant provider-plugin: 0.3.0
    - box support (disk or PXE-BIOS)
    - host OS detection by vagrant
    - VM lifecycle "import/up/halt/destroy"
    - Host-to-Guest networking + SSH
- autoinstall concepts:
    - PXE response steering (tftp per IP, dhcrelay)
    - install.conf steering
    - disklabel templates
    - multiple set sources
    - siteNN.tgz
    - siteNN-hostname.tgz
    - install.site

Architecture

Naming - what’s in the bento?

- Core: plugin loader "framework" + utils
- host + capabilities (Linux, OSX, Free/OpenBSD, ..)
- "box" Disk/BIOS image + metadata (tar.gz)
- guest + capabilities (Linux, Free/OpenBSD, ..)
- provider + capabilities (vbox/VMM/bhyve/...)
- communicators (ssh/winssh/winrm)
- provisioner (shell/ansible/chef/puppet/...)

Plugins

Provider

- lazy loader - overloading classes
- Action (abstraction classes, workflow)
- Driver (host integration, here mainly vmctl)
- Templates (ERB) (vm.conf)

Networking capabilities

- port-forward: open arbitrary ports (on 127.0.0.1) on the host and ssh-forward it into the VM
- bridged network: reach out from VM to The Internet
- "private" network: VM to VM communication on isolated network (brige(8))

Provisioner - post-postinstall

Almost any automation stack can be included into a Vagrant based VM

- (inline) shell scripts
- ansible
- Chef
- Puppet
- Salt
- you-name-it, likely there’s a plugin

Anatomy of an UP session

```
$ uname -a ; bundle exec vagrant status ; bundle exec vagrant up ; \
 bundle exec vagrant ssh -c "uname -a"
OpenBSD ssfnhv011.ham3.rootnexus.net 6.2 GENERIC.MP#134 amd64
Current machine states:
vagrobsd                 not_created (openbsd)

The instance is not created. Run ‘vagrant up‘ to create it.
Bringing machine ’vagrobsd’ up with ’openbsd’ provider...
==> vagrobsd: Verifying VMM present and CPU capable...
==> vagrobsd: Importing an OpenBSD instance
    vagrobsd: Cloning virtual hard drive...
    vagrobsd: Successfully imported a VM image
    vagrobsd: Creating vmctl configuration
==> vagrobsd: Starting the machine...
==> vagrobsd: Waiting for the machine to report its IP address...
    vagrobsd: IP: 100.64.2.3
==> vagrobsd: Waiting for machine to boot. This may take a few minutes...
    vagrobsd: SSH address: 100.64.2.3:22
    vagrobsd: SSH username: root
    vagrobsd: SSH auth method: password
    vagrobsd: Inserting generated public key within guest...
    vagrobsd: Removing insecure key from the guest if it’s present...
    vagrobsd: Key inserted! Disconnecting and reconnecting using new SSH key...
==> vagrobsd: Machine booted and ready!
OpenBSD openbsd62.example.com 6.2 GENERIC#132 amd64
Connection to 100.64.2.3 closed.
$ cat Vagrantfile
Vagrant.configure("2") do |config|
  config.vm.box = "vagrobsd"
  config.ssh.shell = "ksh -l"
  config.ssh.sudo_command = "doas -n %c"
  config.vm.define "vagrobsd" do |v|
    v.vm.hostname = "openbsd-vagrant"
  end
end
```

Overview / Concept

- shell scripts, common and MD (˜3500 lines)
- simple answerfile
- answers can occur multiple for special cases
- install or upgrade
- https + signify

Anatomy

- bsd.rd, init and to /etc/rc
- dot.profile basic setup and launch installer
- choosing autoinstall if netboot (after 5s)
- sets mode and installsets
- configure network
- fetch official mirror list
- fetch answerfile
- disk config
- fetch+install sets
- system configuration, user setup
- relink kernel
- install bootblocks
- custom post-install
- /etc/rc.firsttime after reboot (sys{patch,merge}, fw_update)

Disks (amd64)

- fetch a disklabel(8) template
- OR calculate a root disk layout
- no softraid support yet (quirks available)

Network

- DHCP (inet4) or SLAAC (inet6)
- can use http[s]_proxy
- fetch answerfile
- ftplist.cgi

Debugging

- bails to shell if errors occur
- /tmp/ai/ai.log
- /tmp/ai/ai.conf # answerfile as provided
- /tmp/i/$MODE.resp # logged answers
- /tmp/i/httplist,httpsec,wlanlist
- /tmp/i/cgiinfo
- from shell: install -af $answerfile

base system settings

Generally order doesn’t matter - unless one uses same question multiple times, like installing from more than one source. A full list of questions with defaults and options is in the backup slides.

```
System hostname = myhost
Choose your keyboard layout = us
Start sshd(8) by default = yes
Do you expect to run the X Window System = no
Do you want the X Window System to be started by xenodm = no
Change the default console to = com0
Terminal type = vt220
speed should com0 use = 115200
What timezone are you in = Europe/Berlin
```

Sets location and Disk

It’s possible to repeat the question/answer tuples with differing values. So it’s possible to install the base OpenBSD from official mirrors, and subsequently pull siteNN.tgz from a different/local server.

```
Location of sets = h # http(s)
Set name(s)? = -x* +site*
```

Can be used multiple times, but (A)utolayout only for the rootdisk

```
disk do you wish to initialize = sd0
Which disk is the root disk = sd0
Use (A)uto layout, (E)dit auto layout, or create (C)ustom layout = A
URL to autopartitioning template for disklabel = https://10.1.1.100/disklabeltemplate
```

User

```
Password for root account = seebelow
Allow root ssh login = prohibit-password
Setup a user = toor
Password for user toor = ************* # 13 asteriks
Full name for user toor = Mr Toor
Public ssh key for user toor = ssh-rsa 9092392342394907213 49...=
Public ssh key for root account = ssh-rsa 236745734239489023 84...==
```

installtime networking

```
Time appears wrong. Set to = yes # off > 120s from HTTP
network interface should be used for the initial DHCP request = ix0 # defaults to netboot dev
HTTP proxy URL? = none
HTTP Server? = [http[s]://]10.1.1.100 # also goes to installurl(5)
Unable to connect using https. Use http instead = no # if no protocol given above, https is t
Server directory? = pub/OpenBSD/6.2/amd64
```

runtime networking

```
DNS domain name = example.com
DNS nameservers = 1.1.1.1
network interface do you wish to configure = (phy0|vlan0) # configure hostname.if(5) per inte
Symbolic (host) name for $_if = virtahost # /etc/hosts entry for additional ifs - wont be ask
IPv4 address for (em0|ix0|..) = (dhcp|10.1.1.1|10.2.2.2/24) # static configuration, multiple
Netmask for for (em0|ix0|..) = 255.255.255.0 # static configuration if CIDR missed in above,
Default IPv4 route? = 10.1.1.254 # static configuration if no dhcp, mygate(5)
IPv6 address for (em0|ix0|..) = (autoconf|fd8e:c35e:4631:0::1/64) # static configuration, mul
IPv6 prefix length for (em0|ix0|..) = 64 # static configuration if prefix length missed in ab
# vlan
Which interface:tag should $_if be on = em0 # any physical if, $_if like vlan0
```

Wireless

```
Access point? = any # 80211 setup, ESSID
Security protocol? = (O|W|P) # 80211 setup, answer means: Open/WEP/WPA-PSK
WEP key? = 13_characters # 80211 setup, see ifconfig(8) /nwkey
WPA passphrase? = longpassphrase # 80211 setup, see ifconfig(8) /wpakey
```

Error handling

```
These will happen for customized/additional sets like siteNN.tgz
Checksum test for $_f failed. Continue anyway = no # $_f will be packNN.tgz, currently no use
Unverified sets ... Continue without verification = no #  as above
```

Site packages / scripts

Installer will offer those for selection if present (index.txt!) and matches the hostname. Contents will be just be unpacked like ‘tar zpxf siteNN.tgz -C /mnt’. install.site can be any arbitrary shell script that will be run chrooted in /mnt.

- siteNN.tgz
- siteNN-hostname.tgz
- install.site / upgrade.site

Ohai + Links + Thanks

- Code/Slides - https://github.com/double-p/vagrant-openbsd/
- Kickoff - Glarus, Switzerland / https://ungleich.ch
- Work Time+Travel / https://sysfive.com

Questions?

BEER after the closing session and auction!