Puppet and the HashiStack

Bram Vogelaar
@attachmentgenie

$ whoami

- Used to be a Molecular Biologist
- Then became a Dev
- Now an Ops
- Currently Cloud Engineer @ The Factory

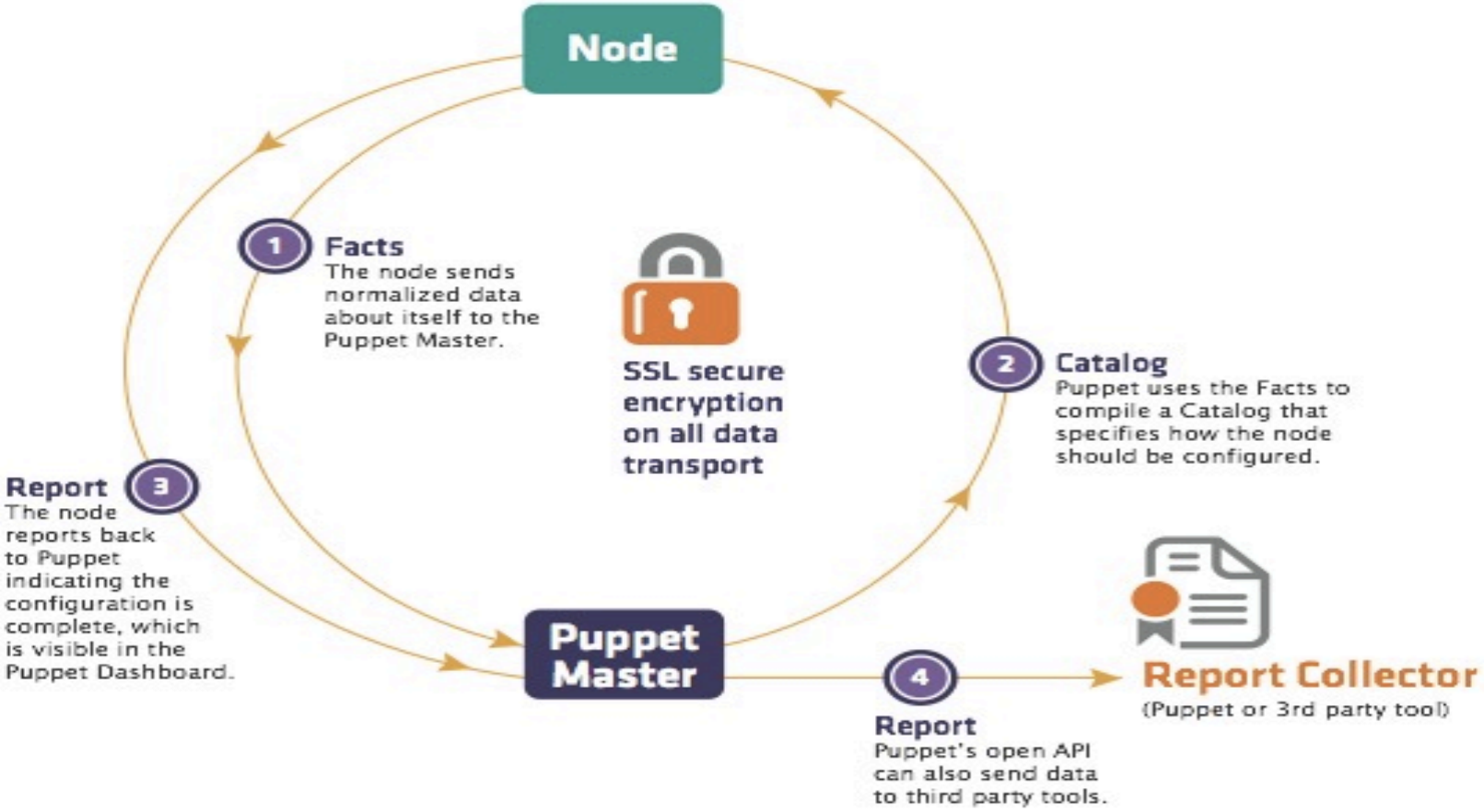
HashiCorp Stack/Suite

Puppet

- Open Source configuration management tool since 2005
- Desired State DSL on top of ruby
- Client – Server architecture
- Runs on pretty much any OS
- Combines node info (Facter), node config (Hiera), node model (DSL) into a catalogue that is applies at regular intervals

https://puppet.com/open-source/#osp

Puppet Workflow



Packer

- Open Source tool to make OS images
- Supports Cloud Providers, Docker, Virtualbox, … (builders)
- Has hooks to provision the base images (provisioners)
- Create artifacts (Post-Processors)

https://www.packer.io/

HCL2

- HashiCorp Configuration Language
- Yet another config management DSL
- Desired state
- Used by multiple HashiCorp tools but also 3rd party tools

https://github.com/hashicorp/hcl

Packer

```
source "azure-arm" "basic-example" {
  …
  vm_size = "Standard_A2"
}

build {
  sources = ["sources.azure-arm.basic-example"]
  provisioner "shell" {
    scripts = ["scripts/common/puppet_install.sh"]
  }
}
```

```
$ packer build template. hcl
```

https://github.com/petems/puppet-install-shell

Packer & Puppet

```
provisioner "puppet-masterless" {
  manifest_file = "site.pp"
}

provisioner "puppet-server" {
  puppet_server = ”puppet.example.com"
}
```

Vagrant

- Open Source tool to bootstrap local vms
  - Build by packer!
- Supports many vm Providers, Docker, Virtualbox, …
- Has hooks to provision the base images (provisioners), Puppet, Chef, Ansible, Bash

https://www.vagrantup.com/

Vagrantfile

```
Vagrant.configure("2") do |config|
  config.vm.box = "base"
  # config.vm.box_check_update = false
  # config.vm.network "forwarded_port", guest: 80, host: 8080
  # config.vm.network "private_network", ip: "192.168.33.10"
  # config.vm.network "public_network"
  # config.vm.synced_folder "../data", "/vagrant_data"
  # config.vm.provider "virtualbox" do |vb|
  #   vb.gui = true
  #   vb.memory = "1024"
  # end
  # config.vm.provision "shell", inline: <<-SHELL
  #   apt-get update
  #   apt-get install -y apache2
  # SHELL
end
```

Vagrant & Puppet

```
$ vagrant plugin install vagrant-puppet-install
```

https://github.com/petems/vagrant-puppet-install

Vagrant & Puppet

```
cat Vagrantfile
case node["provision_type"]
when 'masterless'
  srv.vm.synced_folder "#{environment}/hieradata",
"/etc/puppetlabs/code/environments/#{environment}/hieradata"
  srv.vm.provision :puppet do |puppet|
      puppet.environment_path = "."
      puppet.hiera_config_path = "#{environment}/hiera.yaml"
  end
```

$ vagrant (up,halt,destroy)

Vagrant & Puppet

```
cat Vagrantfile
case node["provision_type"]
when 'puppet_agent'

  srv.vm.provision "puppet_server" do |puppet|
    puppet.options       = "-t --environment #{environment}"
    puppet.puppet_server = "puppetmaster.#{environment}.vagrant"
  end

  srv.trigger.after :destroy do |trigger|
    trigger.name = "Cleaning puppet certificate"
    trigger.run = {inline: "vagrant ssh puppetmaster -c 'sudo /opt/puppetlabs/bin/puppetserver ca clean --certname
#{node["hostname"]}'"}
  end
```

$ vagrant (up,halt,destroy)

Terraform

- Open Source Automation Tool
- “cloud” oriented
- Cloud are API’s
- API’s oriented

- Terraform is an open source automation tool which can deal with any kind of CRUD api’s – including major cloud providers

https://www.terraform.io/

The Terraform Model

- You model your infrastructure
- You make a plan
- If ok, you apply that plan
- Current state is saved for future changes

$ terraform (fmt, validate, plan, apply)

Lets magic a node

```
resource "azurerm_virtual_machine" "puppetmaster" {
  name                = "vm-${var.node_name}"
  location            = "${var.azure_region_name}"
  resource_group_name = "${var.resource_group}"
  network_interface_ids  = ["${azurerm_network_interface.puppetmaster.id}"]
  vm_size             = "${var.vm_size}"

  storage_image_reference {
    id = "${var.packerimage_id}" # ← your packer image id
  }

  os_profile {
    ...
  }
}
```

$ terraform (fmt, validate, plan, apply)

Lets magic a node

```
resource "azurerm_virtual_machine" "puppetmaster" {
  ...

  os_profile {
    custom_data    = "${data.template_file.bootstrap_sh.rendered}"
  }

}
output "private_ip" {
  value = "${azurerm_network_interface.network_interface.private_ip_address}"
}
```

```
$ terraform (fmt, validate, plan, apply)
```

Bootstrap your Puppet Server

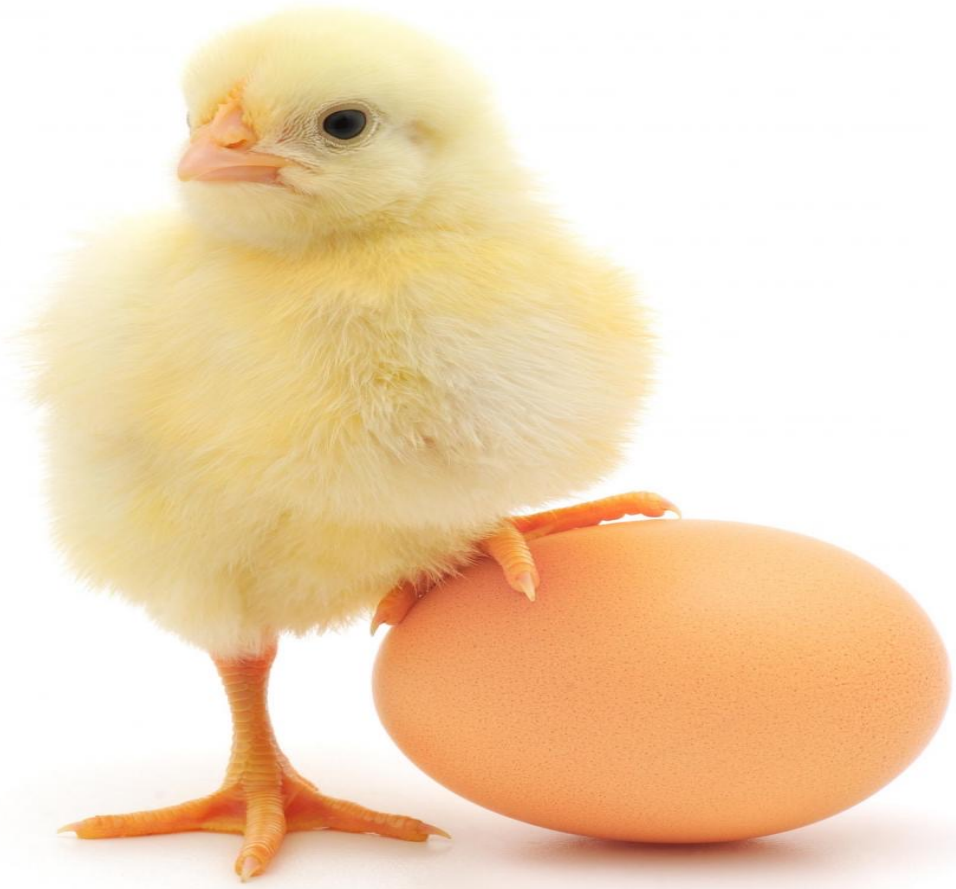
```
#!/usr/bin/env bash

apt-get install puppetserver
puppet config set --section agent environment ${environment}
systemctl start  puppetserver
```

Move it all to a module

```
module "web" {
  source        = "../modules/azure/instance"
  domain        = "${var.domain}"
  environment   = "${var.environment}"
  node_name     = "web"
  private_subnet = "${module.inuits_play_bootstrap.subnet_id}"
  puppet_master  = "${var.puppet_master}"
}
```

It’s a DNS problem



Consul

- Open Source Service Discovery Tool
- Build-in KV store
- Service Mesh tool

https://www.consul.io/

Consul & Puppet

```
class { '::consul':
  config_hash   => $config,
  version       => $version,
}
::consul::service { 'puppetmaster':
  port   => 8140,
}
::consul::check { 'puppetmaster_tcp':
  interval     => '10s',
  tcp          => 'localhost:8140',
  notes        => 'Puppetmasters listen on port 8140',
  service_id => 'puppetmaster’,
}
```

```
dig @127.0.0.1 -p 8600 puppetmaster.service.consul ANY
```

https://forge.puppet.com/KyleAnderson/consul

Consul~Icinga(2) Exit Codes

```
::consul::check { 'puppetmaster':
  interval     => '10s',
  script       => '/usr/lib64/nagios/plugins/puppetmaster',
  notes        => 'Puppetmasters listen on port 8140',
  service_id => 'puppetmaster',
}
```

Geolocation Stuff

```
consul_prepared_query { 'puppetmaster':
  ensure                 => 'present',
  service_name           => 'puppetmaster',
  service_failover_n     => 1,
  service_failover_dcs   => [ 'failover', 'dr' ],
  service_only_passing => true,
  ttl                    => 10,
}
```

```
dig @127.0.0.1 -p 8600 puppetmaster.query.consul ANY
```

X509 is hard

```
#!/usr/bin/env bash

apt-get install puppetserver
puppet config set --section agent environment ${environment}
puppet config set --section master dns_alt_names \
puppet,puppetmaster.service.consul,puppetmaster.query.consul
puppet config set --section master autosign /etc/puppetlabs/puppet/autosign-policy.rb
systemctl start  puppetserver
```

So what about that chicken?

```
#!/usr/bin/env bash
consul agent -retry-join "provider=azure \
  tag_name=consul \
  tenant_id=${tid} \
  client_id=${cid} \
  subscription_id=${sid} \
  secret_access_key=${ro_key}”
cat > /etc/puppetlabs/puppet/csr_attributes.yaml << YAML
---
extension_requests:
  pp_preshared_key: ${psk}
  pp_role: ${role}
YAML

puppet config set --section agent environment ${environment}
puppet config set --section agent server puppet.query.consul
/opt/puppetlabs/bin/puppet agent -t
```

Fast Exported Resources

```
::consul::watch { 'detect_backend_changes':
  type           => 'service',
  handler        => '/usr/bin/runpuppet.sh',
  service        => 'node_exporter',
  passingonly => true,
  require        => File['/usr/bin/runpuppet.sh'],
}
```

Vault

- Open Source tool to do secrets management
- Secure, store and tightly control access to tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data using a UI, CLI, or HTTP API.
- Certificate management
- Password rotation

https://www.vaultproject.io/

Pesky Passwords

```
class profiles::security::vault (
  Variant[Hash, Array[Hash]] $listener = {
    'tcp' => {
      'address'            => '127.0.0.1:8200',
      'cluster_address' => '127.0.0.1:8201',
    },
  },
  Hash $storage = { 'consul' => { 'address' => '127.0.0.1:8500', 'path' => 'vault/' }},
){
  class {'vault':
    enable_ui => true,
    listener     => $listener,
    storage     => $storage,
}
```

https://forge.puppet.com/jsok/vault

Pesky Passwords

```
$ vault unseal
$ vault write kv/my-secret value="s3c(eT"
$ vault read kv/mysecret
Key                 Value
---                 -----
refresh_interval    768h
mysecret            s3c(eT
```

Vault & Hiera

```
cat hiera.yaml
---
version: 5
hierarchy:
  - name: "Hiera-vault lookup”
    lookup_key: hiera_vault
    options:
      confine_to_keys:
        - '^sensitive_.*’
        - '^password.*’
      address: http://active.vault.service.consul:8200
      mounts:
        generic:
          - secret/puppet/%{::trusted.certname}/
          - secret/puppet/common/
```

https://github.com/petems/hiera_backend_vault

Puppet > 6

```
$password = Deferred('vault_lookup::lookup',["password/mysql"], 'https://active.vault.service.consul:8200',)

class { '::mysql::server':
  root_password => $password,
}
```

Nomad

- Open Source tool to do dynamic workload scheduling
- Batch, containerized, and non-containerized applications.
- Has native Consul and Vault integrations.

https://www.nomadproject.io/

Nomad & Puppet

```
class { '::nomad':
  config_hash = {
    'client’          => { 'enabled' => true, },
    'consul’         => { 'address' => '127.0.0.1:8500’, },ß
    'server'          => {
      'enabled'                => true
      'bootstrap_expect' => 3,
    },
    'vault’          => {
      'enabled’                => true,
      'address’                => 'https://active.vault.service.consul:8200',
      'create_from_role' => 'nomad-cluster',
      'token’                  => 's.krHYYcd8PRzpSBO59AE6sawO',
     },
  }
}
```

https://forge.puppet.com/modules/puppet/nomad

Waypoint

- Modern workflow to build, deploy, and release across platforms.
  - Build
  - Deploy
  - Release

https://www.waypointproject.io/
[WIP] https://github.com/attachmentgenie/attachmentgenie-waypoint

Boundary

- Identity-based access for zero trust security
  - Authenticate & authorize
  - Connect
  - Access

https://www.boundaryproject.io/
[WIP] https://github.com/attachmentgenie/attachmentgenie-boundary

Bolt

- Open Source tool to do task orchestration
- Aimed at fire and forget 2nd day operations
- Support tasks written in Puppet or YAML

https://puppet.com/open-source/bolt/

Bolt

```
# bolt-project.yaml
modulepath:
  - modules
modules:
  - name: puppetlabs/apt

# inventory.yaml
groups:
- name: all-of-azure
  targets:
    - uri: 1.2.3.4:22
      name: prod
  config:
    transport: ssh
    ssh:
      user: ubuntu
```

Bolt

bolt project init

bolt module install

bolt command run apt action=update --targets servers

Bolt & Terraform

```
groups:
  - name: all-of-azure
    targets:
      - _plugin: terraform
        dir: /path/to/terraform/project
        resource_type: azurerm_virtual_machine
        target_mapping:
          uri: public_ip

plan do_important_stuff (TargetSpec $targets){
  run_task('terraform::initialize', 'dir' => '/path/to/terraform/project')
  $apply_result = run_plan('terraform::apply', 'dir' => '/path/to/terraform/project', 'return_output' => true)
  run_task('important::stuff', $targets, 'task_var' => $apply_result)
}
```

https://forge.puppet.com/modules/puppetlabs/terraform

Bolt & Vault

```
targets:
  - ...
config:
  ssh:
    user: root
    private-key:
      key-data:
        _plugin: vault
        server_url: https://active.vault.service.consul:8200
        auth:
          method: token
          token: xxxxx-xxxxx
        path: secrets/bolt
        field: private-key
```

https://forge.puppet.com/modules/puppetlabs/vault

Contact

bram@attachmentgenie.com

@attachmentgenie

https://www.slideshare.net/attachmentgenie

The Floor is yours…

Questions ?