PURESTORAGE®

Pure Storage Wavemakers Infrastructure-as-code and Terraform

This Session

- Infrastructure-as-code – laying the foundations
- Infrastructure-as-code – landscape
- An introduction to Terraform
- What Terraform means for Pure Storage
- Demonstrations
- Useful resources

About Me

- Microsoft Data Platform Solutions Architect for EMEA
- 20+ years experience of being a database focused developer either working as a developer or working within a team of developers
- Twitter: ChrisAdkin8
- GitHub: chrisadkin

Essential Terminology

- Imperative

    1. Do this
    2. Do that
    3. Do the other

- Declarative

    1. Provision me what I want in the exact state I declare it in

Essential Terminology

- Idempotency

The ability to apply something multiple times without the result changing

The Infrastructure-As-Code Landscape

AWS CloudFormation

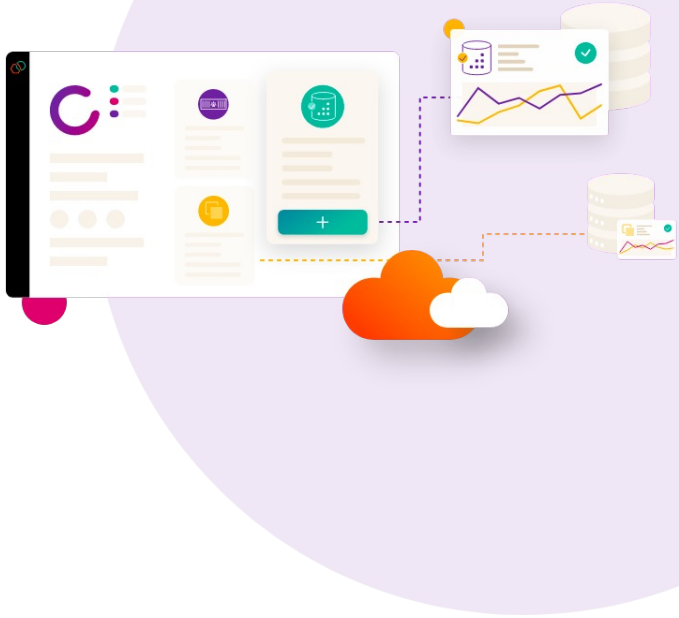
Azure Resource Manager Templates (ARM)

Azure Bicep

The Infrastructure-As-Code Landscape

What Does Terraform Mean for Pure Storage ?

portworx
data services

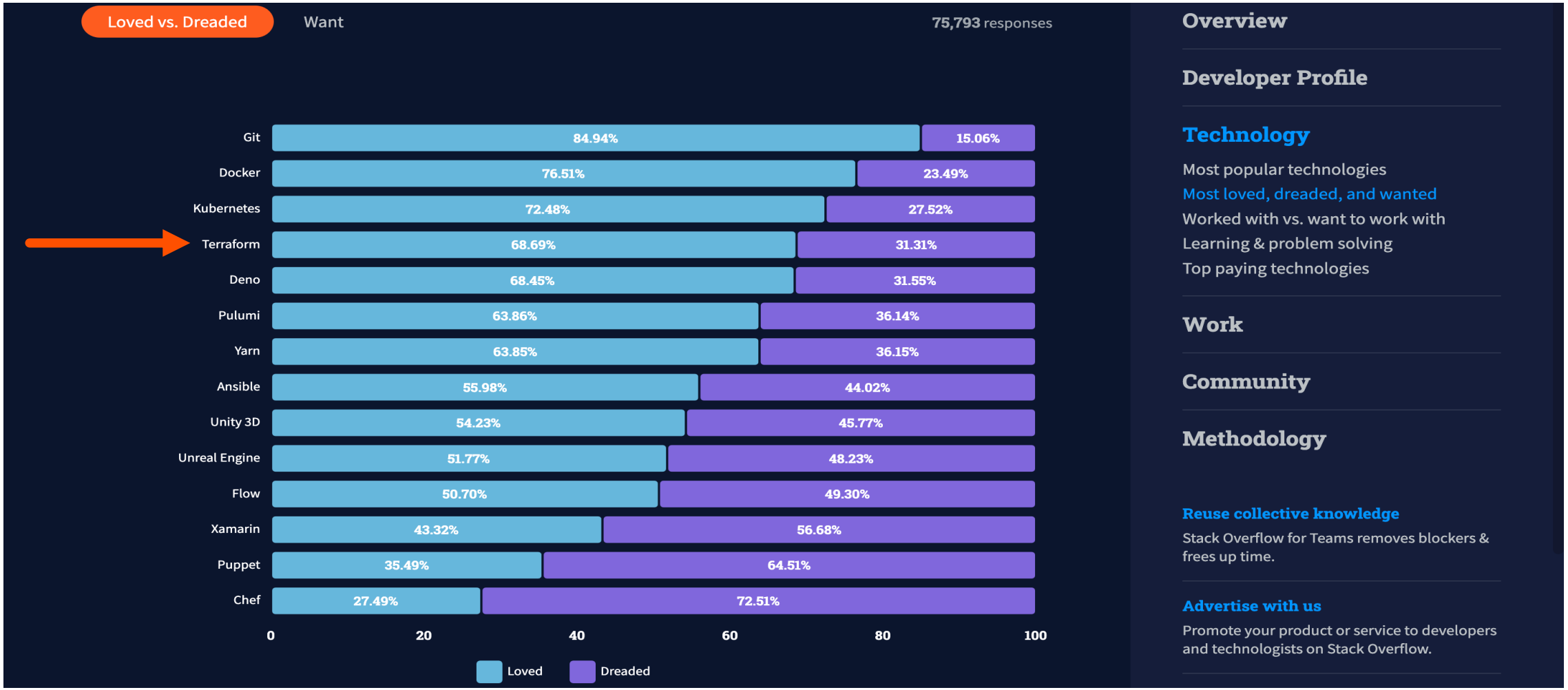


Terraform – Why Should You Care ?

- Consistency
- Flexibility
- Cloud spend reduction
- Source code control

Terraform Traction In The Developer Community

The Stackoverflow developer survey 2021



Download Terraform

macOS | Windows | Linux | FreeBSD | OpenBSD | Solaris

PACKAGE MANAGER

```
$ brew tap hashicorp/tap
$ brew install hashicorp/tap/terraform
```

View Tutorials at HashiCorp Learn

MACOS BINARY DOWNLOAD

Terraform 1.1.8

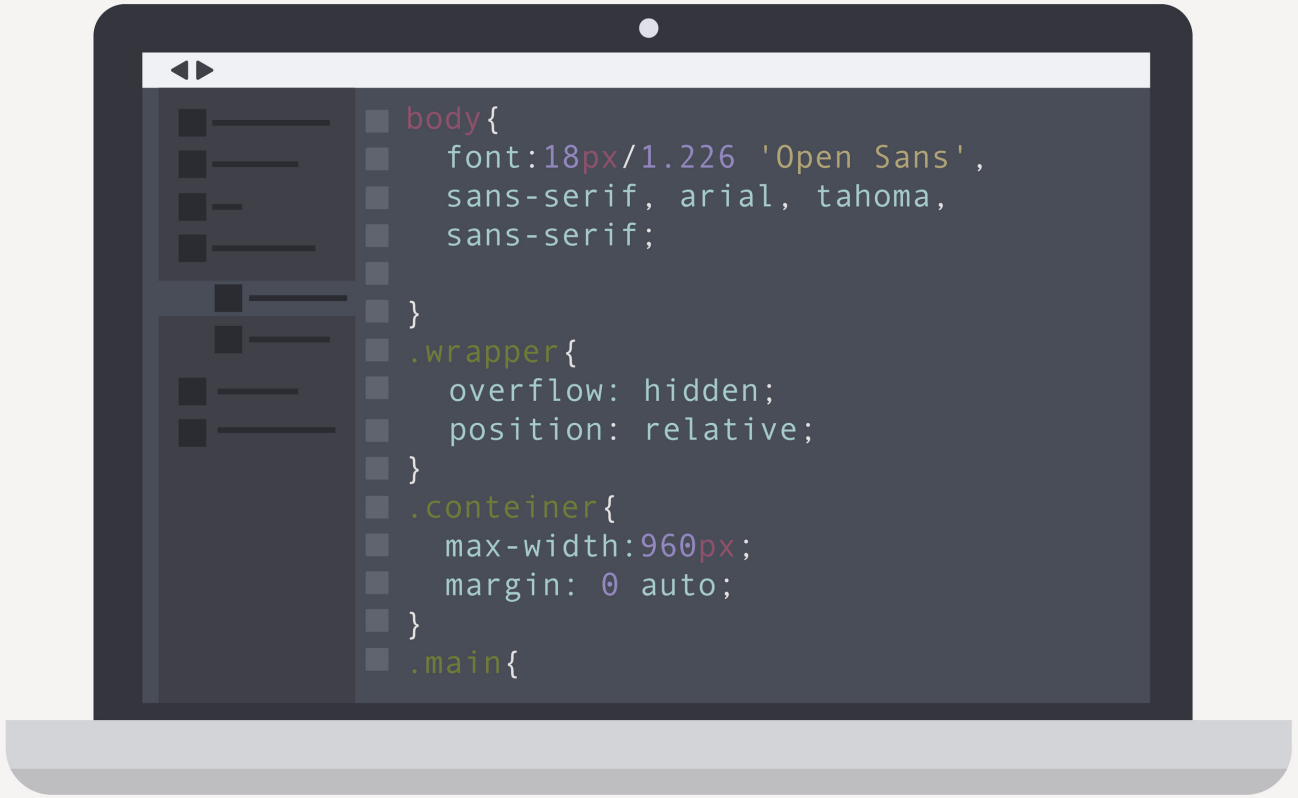
Amd64 Arm64

Bandwidth courtesy of

fastly

Want it hosted? Deploy on Terraform Cloud. Sign up for Terraform Cloud →

Demonstrations



Source sample text

```
(base) cadkin@cadkin--MacBookPro16 demo-basics %
```

We change the name of the virtual machine in the terraform.tfvars file

Re-run terraform apply

What state do we have now ?

```
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
```

I want different state for different projects

How do I achieve this ?

```
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
(base) cadkin@cadkin--MacBookPro16 demo-basics %
```

Can I store my state
more securely
than on my laptop ?

```
(base) cadkin@cadkin--MacBookPro16 tutorial-02 %
(base) cadkin@cadkin--MacBookPro16 tutorial-02 %
(base) cadkin@cadkin--MacBookPro16 tutorial-02 %
(base) cadkin@cadkin--MacBookPro16 tutorial-02 %
```

Can I deploy **multiple** virtual machines in one configuration?

```
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
(base) cadkin@cadkin--MacBookPro16 demo-foreach %
```

How do I perform text substitution on a file ?

https://raw.githubusercontent.com/kubernetes-sigs/kubespray/master/inventory/sample/group_vars/all/all.yml

```
---
## Directory where the binaries will be installed
bin_dir: /usr/local/bin

## The access_ip variable is used to define how other nodes should access
## the node.  This is used in flannel to allow other flannel nodes to see
## this node for example.  The access_ip is really useful AWS and Google
## environments where the nodes are accessed remotely by the "public" ip,
## but don't know about that address themselves.
# access_ip: 1.1.1.1


## External LB example config
## apiserver_loadbalancer_domain_name: "elb.some.domain"
# loadbalancer_apiserver:
#   address: 1.2.3.4
#   port: 1234

## Internal loadbalancers for apiservers
# loadbalancer_apiserver_localhost: true
# valid options are "nginx" or "haproxy"
# loadbalancer_apiserver_type: nginx  # valid values "nginx" or "haproxy"

## If the cilium is going to be used in strict mode, we can use the
## localhost connection and not use the external LB. If this parameter is
## not specified, the first node to connect to kubeapi will be used.
# use_localhost_as_kubeapi_loadbalancer: true

## Local loadbalancer should use this port
## And must be set port 6443
loadbalancer_apiserver_port: 6443

## If loadbalancer_apiserver_healthcheck_port variable defined, enables proxy liveness check for nginx.
loadbalancer_apiserver_healthcheck_port: 8081

### OTHER OPTIONAL VARIABLES

## Upstream dns servers
# upstream_dns_servers:
#   - 8.8.8.8
#   - 8.8.4.4

## There are some changes specific to the cloud providers
## for instance we need to encapsulate packets with some network plugins
## If set the possible values are either 'gce', 'aws', 'azure', 'openstack', 'vsphere', 'oci', or 'external'
```

Where To Find The Demo Code

How do I perform text substitution on a file **using variables** ?

```
(base) cadkin@cadkin--MacBookPro16 demo-advanced-template % ls
README.md       main.tf         templates       variables.tf
(base) cadkin@cadkin--MacBookPro16 demo-advanced-template %
```

Can I deploy multiple resources as **in a single configuration** ?

```
fsauser@z-fsa-util:~/Arc-PX-VMware-Faststart/azure$
fsauser@z-fsa-util:~/Arc-PX-VMware-Faststart/azure$
fsauser@z-fsa-util:~/Arc-PX-VMware-Faststart/azure$
fsauser@z-fsa-util:~/Arc-PX-VMware-Faststart/azure$
fsauser@z-fsa-util:~/Arc-PX-VMware-Faststart/azure$
```

Where To Find The Demo Code

```
git clone https://github.com/chrisadkin/Pure-Wavemakers-Terraform.git
```

1. Introductory tour of terraform – `demo-basics`
2. State demonstration – `demo-basics`
3. Workspace demo – `demo-basics`
4. Foreach demonstration – `demo-foreach`
5. Backend demonstration – `demo-backend`
6. Simple template demo – `demo-replace-text`
7. Advanced template demo – `demo-advanced-template`

```
git clone https://github.com/chrisadkin/Arc-PX-VMware-Faststart.git
```

8. Introductory tour of terraform – `azure`

Ansible Versus Terraform

	Ansible	Terraform
Unit of deployment	Playbook	Configuration
Focus area	Configuration management	Orchestration
Declarative / Imperative	Imperative	Declarative
Language	YAML	HCL
Stateful ?	No	Yes
Codeless resource destruction ?	No	Yes

A Good Analogy

Terraform is like a tool for building a house

Ansible is like a tool for decorating a house

Provisioners

```
resource "aws_instance" "web" {
  .
  .
  .
  provisioner "local-exec" {
    command = "echo The server's IP address is ${self.private_ip}"
  }

  provisioner "local-exec" {
    when.   = destroy
    command = "echo Destroy time"
  }
}
```


Provisioners

- A provisioner is a means of configuring a resource once it has been created
- TL;DR you can specify imperative code to be executed against the resource once it has been created
- Prefer the use resources and use provisioners as a last resort
- This is Hashicorp's official stance on provisioners

Best Practices

- Use .tfvars files
- Use .gitignore files to prevent sensitive information from being pushed to your GitHub repo
- Avoid storing state in local files
- Use provisioners sparingly
- In organizations with rigorous change control procedures, use fixed plans
- Prefer the use of HCL’s built in string handling capabilities over embedding perl, sed, awk etc . . .
- Group related configurations into modules
- Leverage source code control system such as GitHub

Useful Resources

- Hashicorp documentation for Cloud Block Store Terraform providers
- Hashicorp self-guided tutorials
- Terraform download page
- Terraform best practices
- GitHub repo for code used in the demonstrations

Source sample text