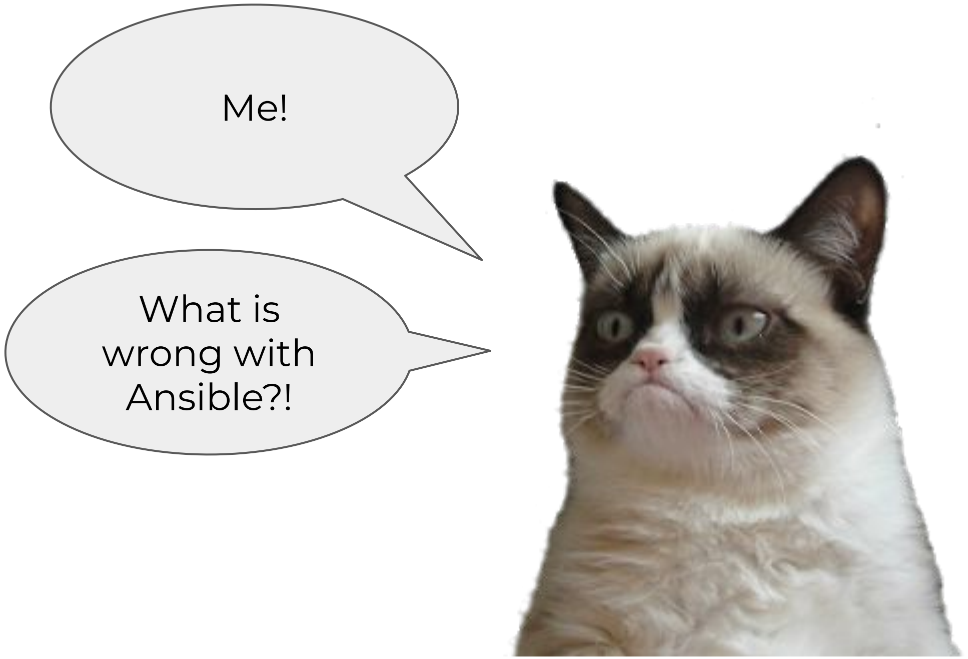
FivexL

Why I don’t recommend learning Ansible?

DevOops Piter 2020

Who is using Ansible and came to see what is wrong with me??? ✋

FivexL


Me!
What is wrong with Ansible?!

This talk is not

Ansible vs Terraform

Ansible vs CloudFormation

Ansible vs Chef

Ansible vs Puppet

Ansible vs Pulumi

Ansible vs SaltStack

Ansible vs Bash

Well, you got the idea...

This is talk is

based on our experience

what we recommend to customers

pragmatic and subjective

not a silver bullet

And not about Ansible?

Ansible Is...

Simple

Human readable automation

No special coding skills needed

Tasks executed in order

Get productive quickly

Powerful

App deployment

Configuration management

Workflow orchestration

Orchestrate the app lifecycle

Agentless

Agentless architecture

Uses OpenSSH and WinRM

No agents to exploit or update

Predictable, reliable and secure

https://www.ansible.com/overview/it-automation

PROVISIONING

Your apps have to live somewhere. If you're PXE booting and kickstarting bare-metal servers or VMs, or creating virtual or cloud instances from templates, Ansible and Red Hat® Ansible® Tower help streamline the process.

CONFIGURATION MANAGEMENT

Centralizing configuration file management and deployment is a common use case for Ansible, and it's how many power users are first introduced to the Ansible automation platform.

APPLICATION DEPLOYMENT

When you define your application with Ansible, and manage the deployment with Ansible Tower, teams are able to effectively manage the entire application lifecycle from development to production.

CONTINUOUS DELIVERY

Creating a CI/CD pipeline requires buy-in from numerous teams. You can't do it without a simple automation platform that everyone in your organization can use. Ansible Playbooks keep your applications properly deployed (and managed) throughout their entire lifecycle.

SECURITY AUTOMATION

When you define your security policy in Ansible, scanning and remediation of site-wide security policy can be integrated into other automated processes and instead of being an afterthought, it'll be integral in everything that is deployed.

ORCHESTRATION

Configurations alone don't define your environment. You need to define how multiple configurations interact and ensure the disparate pieces can be managed as a whole. Out of complexity and chaos, Ansible brings order.

https://www.ansible.com/use-cases

Problem

Problem

Context

Problem

Context

Method

“

Problem

Context

Method

Tool

Problem = IT change management automation

“

Problem = IT change management automation

Context = bare metal / vm

FivexL

“

Problem = IT change management automation

Context = bare metal / vm

Method = infra as code / configuration sync

Problem = IT change management automation

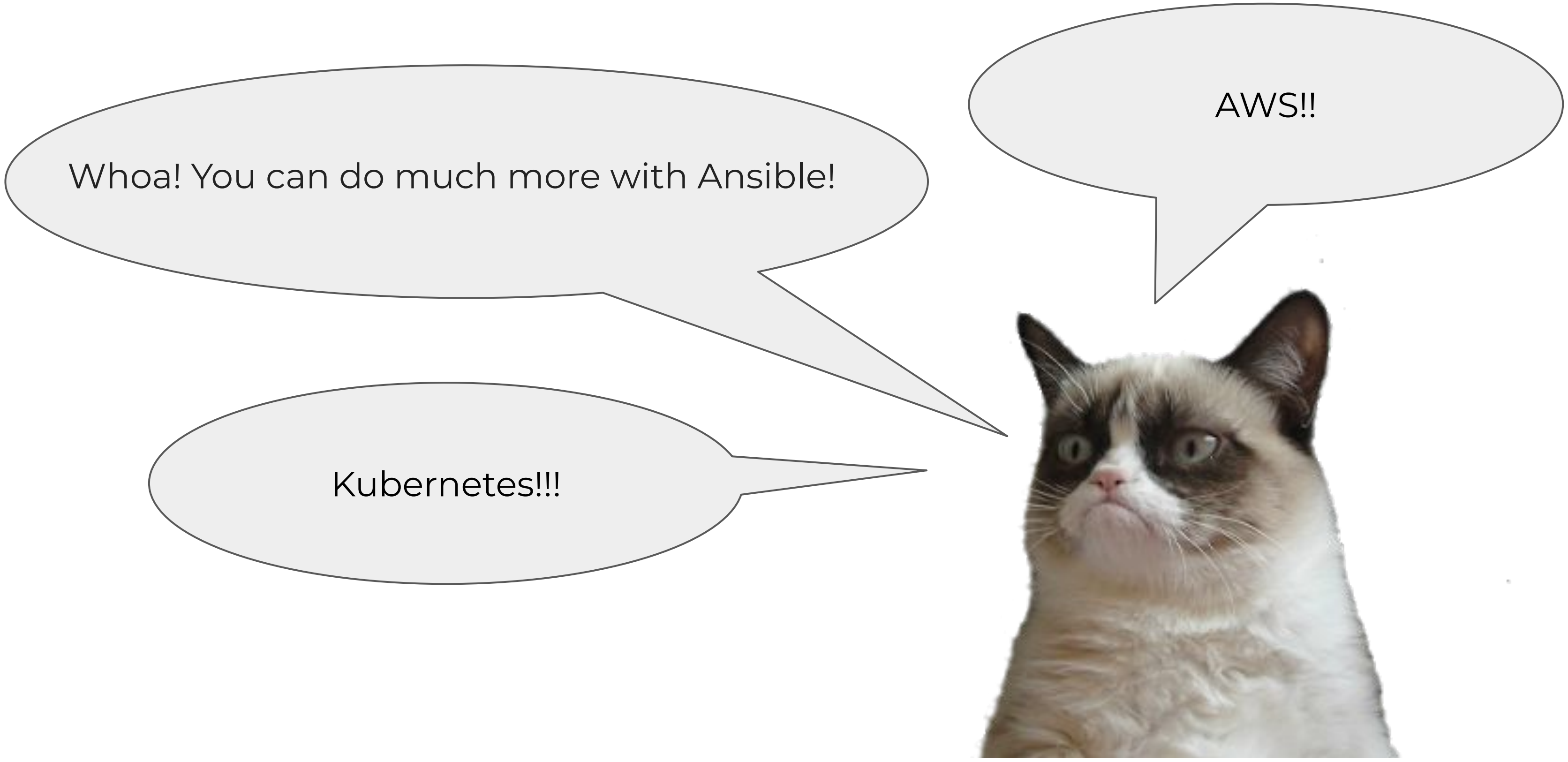
Context = bare metal / vm

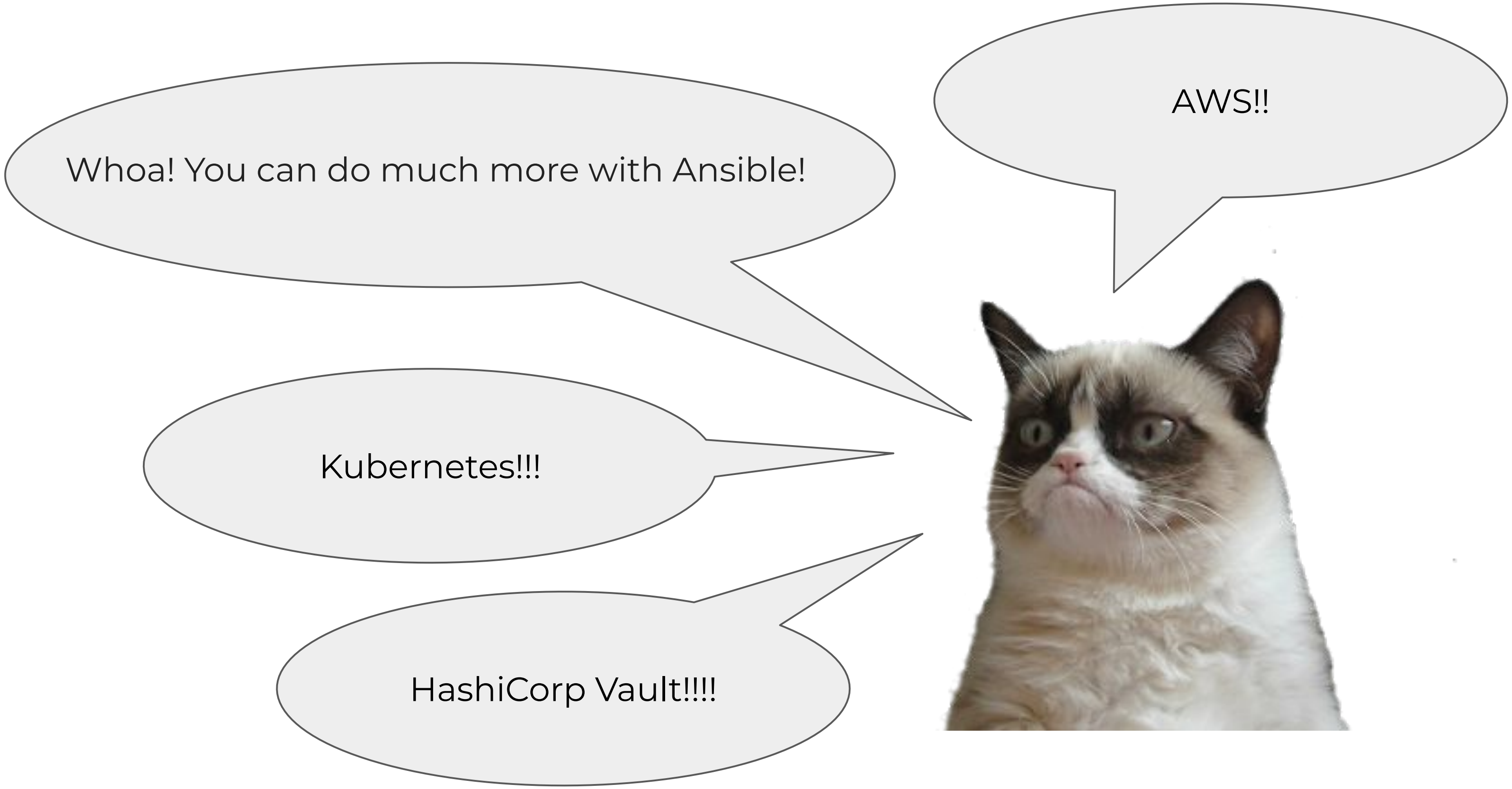
Method = infra as code / configuration sync

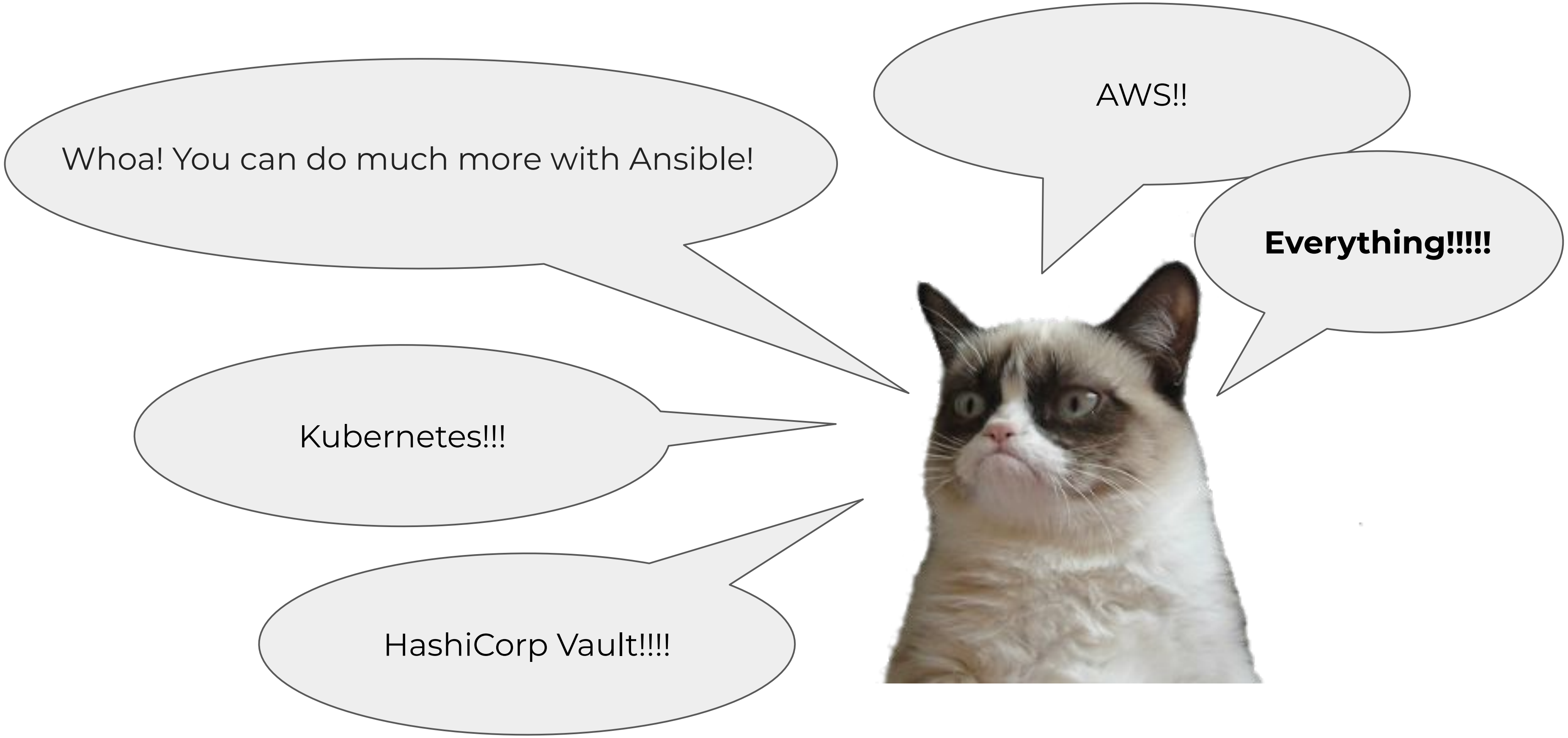
Tool = Ansible

Whoa! You can do much more with Ansible!

Whoa! You can do much more with Ansible!
AWS!!


Whoa! You can do much more with Ansible!
AWS!!
Kubernetes!!!


AWS!!
Whoa! You can do much more with Ansible!
Kubernetes!!!
HashiCorp Vault!!!!


Whoa! You can do much more with Ansible!
AWS!!
Everything!!!!!
Kubernetes!!!
HashiCorp Vault!!!!

Problem ???

Context ???

Method ???

Tool ???

"Everything looks like a nail."

Community

Long-term support

Hiring

Business focus

Configuration management

Server provisioning

Application deployment

Task scheduling/orchestration

"

Problem = IT change management automation

Context = ???

Method = ???

Tool = ???

Andrey Devyatkin

Cloud Engineering Specialist

AWS and HashiStack

Co-Founder at FivexL

Public speaker

Co-Host at DevSecOps Talks podcast

www.fivexl.io | hello@fivexl.io

The only constant is change

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Five theses

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

The future is already here - it is just not evenly distributed

FivexL

Five theses

→ The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

The future is already here - it is just not evenly distributed

FivexL

Big picture - reasons for change

Consumer

Has a problem or desire

Ready to exchange cash for solution or satisfaction

Big picture - reasons for change

Consumer	Entrepreneur
Has a problem or desire	Identifies business opportunity
Ready to exchange cash for solution or satisfaction	Tests business model
	Builds and offers a product

Big picture - reasons for change

Consumer

Has a problem or desire

Ready to exchange cash for solution or satisfaction

Entrepreneur

Identifies business opportunity

Tests business model

Builds and offers a product

Money machine?

Big picture - reasons for change

Consumer

Has a problem or desire

Ready to exchange cash for solution or satisfaction

Entrepreneur

Identifies business opportunity

Tests business model

Builds and offers a product

Competition

Offers similar or superior product

Tries to win market share

Big picture - reasons for change

Consumer

Has a problem or desire

Ready to exchange cash for solution or satisfaction

Entrepreneur

Identifies business opportunity

Tests business model

Builds and offers a product

Competition

Offers similar or superior product

Tries to win market share

Market evolution

Consumer preferences change

Market saturation

Global economy shifts

Big picture - reasons for change

Consumer

Has a problem or desire

Ready to exchange cash for solution or satisfaction

Entrepreneur

Identifies business opportunity

Tests business model

Builds and offers a product

Competition

Offers similar or superior product

Tries to win market share

Market evolution

Consumer preferences change

Market saturation

Global economy shifts

**Business have to constantly adapt to market change
in order to survive and stay relevant
because the cheese is constantly moving**

“

The change is inevitable and change is causing entropy

“

Entropy - the degradation of the matter and energy in the universe to an ultimate state of inert uniformity

Merriam-Webster dictionary

https://www.merriam-webster.com/dictionary/entropy

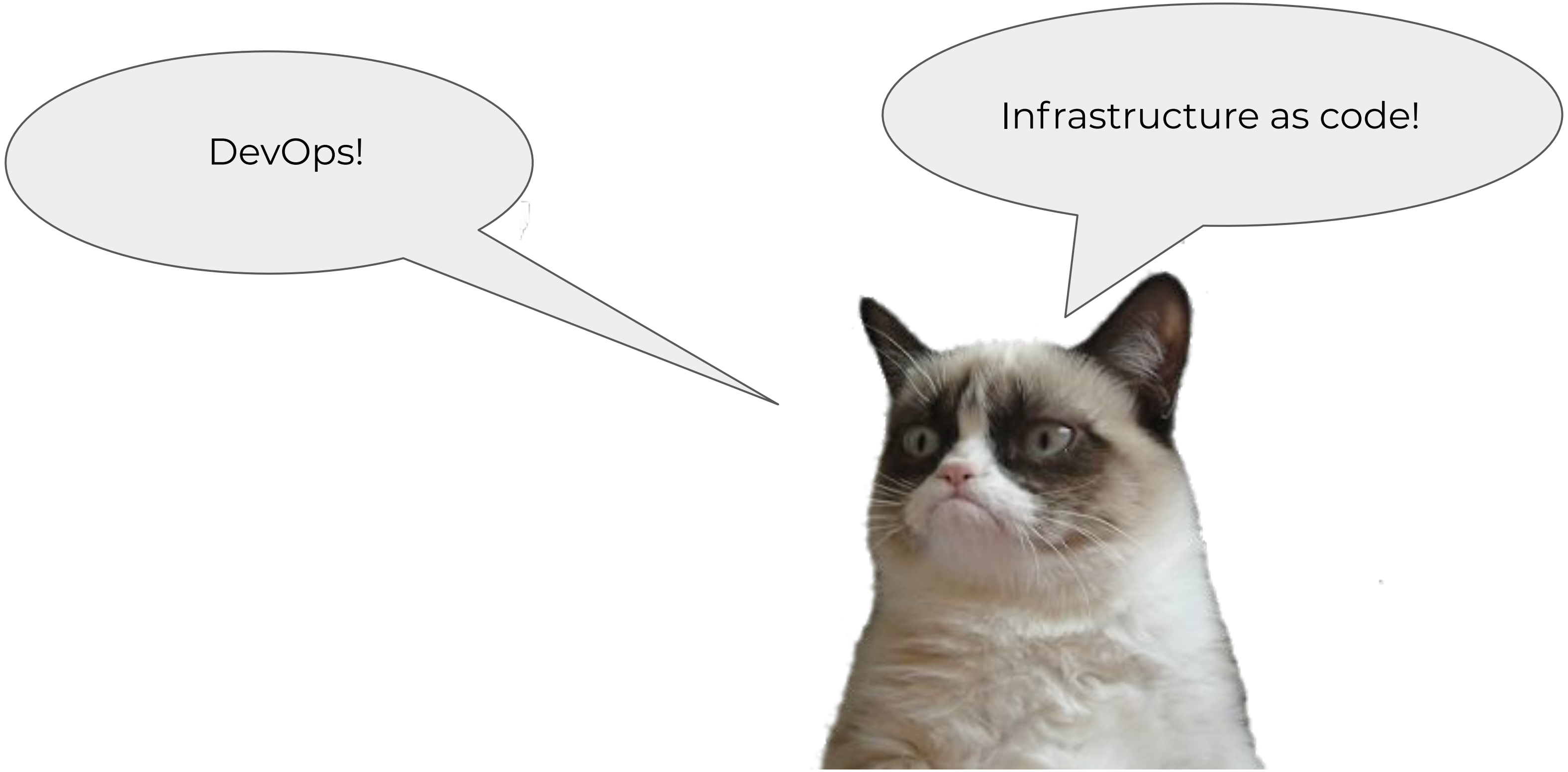
Yo! What all of this have to do with Ansible?!!

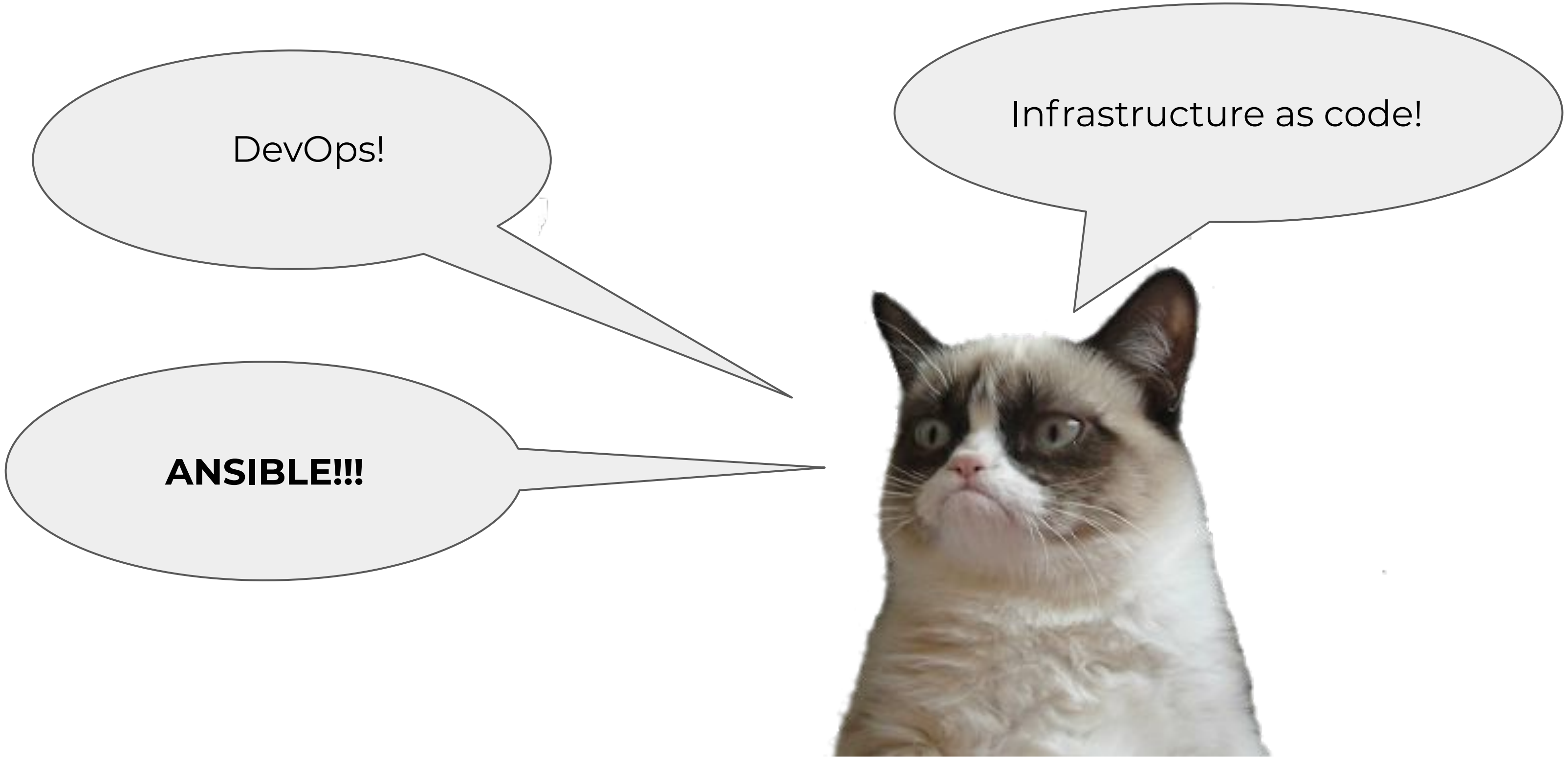
“

How do we allow for the change while minimizing entropy?

FivexL

FivexL


DevOps!
Infrastructure as code!


DevOps!
Infrastructure as code!
ANSIBLE!!!

Problem = IT change management automation

Context = ???

Method = ???

Tool = ???

Five theses

The only constant is change

→ Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

The future is already here - it is just not evenly distributed

Configuration Drift is the phenomenon where servers in an infrastructure **become more and more different** from one another as time goes on, due to manual **ad-hoc changes and updates, and general entropy**.

Keif Morris

http://kief.com/configuration-drift.html

Configuration changes are regularly needed to tweak the environment so that it runs efficiently and communicates properly with other systems. This requires some mix of command-line invocations, jumping between GUI screens, and editing text files.

The result is a **unique snowflake** - good for a ski resort, bad for a data center.

Martin Fowler

https://martinfowler.com/bliki/SnowflakeServer.html

Why is this a problem?

Stability of the system

Diverts resources from creating value for a customer

Impacts our ability to reliably deliver new software

Impacts MTTR

Why is this a problem?

AWS EC2 instance shutdown

Emergency security patching

GPU drivers update

Security incident investigation

“Legacy” system reconfiguration

FivexL

Problem = IT change management automation

Context = bare metal / vm

Method = ???

Tool = ???

So what do we do about it?

Four principles

Infrastructure elements should be

Reproducible

Disposable

Consistent

No end state

Infrastructure as code is the process of managing and **provisioning** computer data centers **through machine-readable definition files**, rather than physical hardware configuration or interactive configuration tools.

https://en.wikipedia.org/wiki/Infrastructure_as_code

“

Infrastructure as code means applying tools and practices from software engineering to infrastructure management

https://www.youtube.com/watch?v=K843Ukiw3d8

Problem = IT change management automation

Context = bare metal / vm

Method = infra as code / configuration sync

Tool = ???

Server Configuration Management tools

Puppet
Chef
Ansible
SaltStack

Automated provisioning

Tools are focused on applying configuration after server start

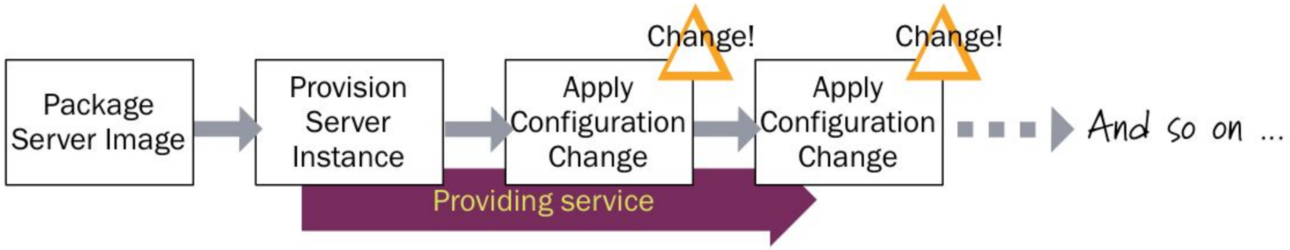
Requires additional configuration

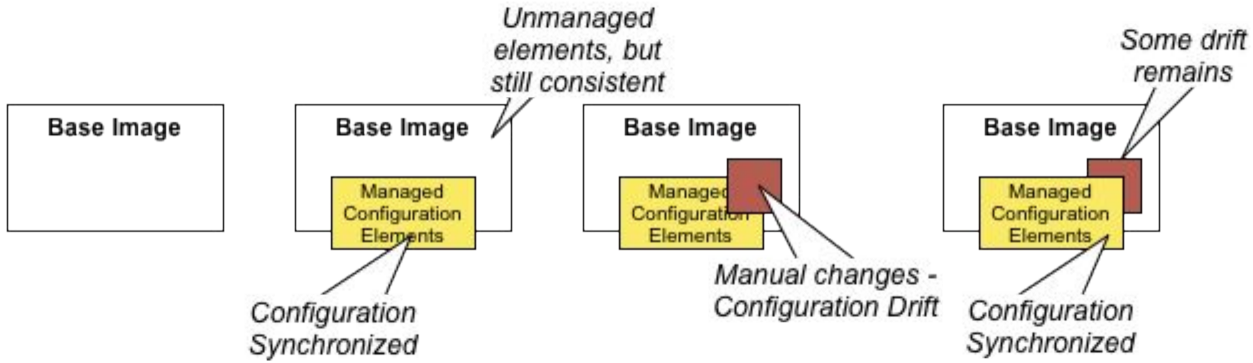
Either command server or ability to directly connect to running server or server to have a connection to control center

Configuration drift tracking

Some tools provide an ability to track changes done to files on server and automatically handle them

Configuration synchronization

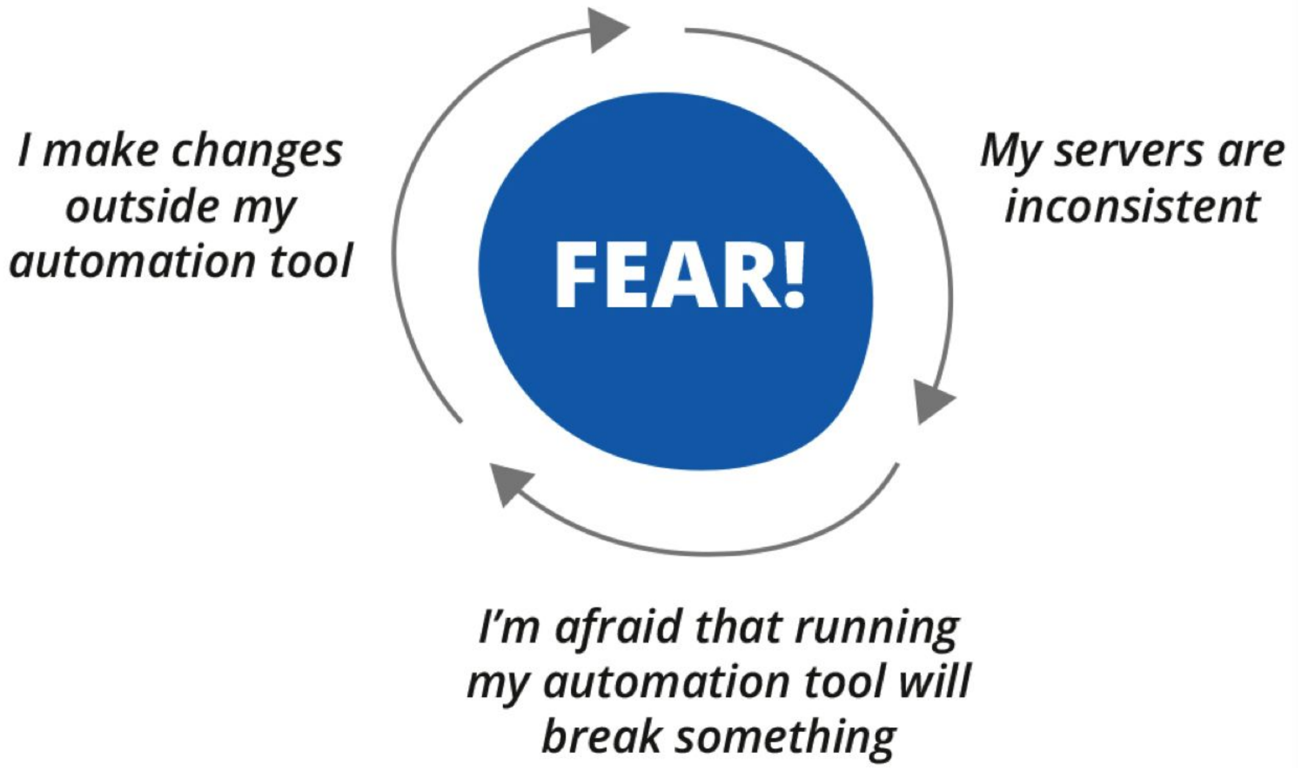

Package Server Image
Provision Server Instance
Apply Configuration Change
Change!
Apply Configuration Change
Change!
And so on ...
Providing service


Base Image
Base Image
Managed Configuration Elements
Unmanaged elements, but still consistent
Configuration Synchronized
Base Image
Managed Configuration Elements
Manual changes - Configuration Drift
Some drift remains
Base Image
Managed Configuration Elements
Configuration Synchronized

https://martinfowler.com/bliki/ConfigurationSynchronization.html

The Automation Fear Spiral

When infra synchronization fails


I make changes outside my automation tool
FEAR!
My servers are inconsistent
I'm afraid that running my automation tool will break something

https://www.thoughtworks.com/insights/blog/infrastructure-code-automation-fear-spiral

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

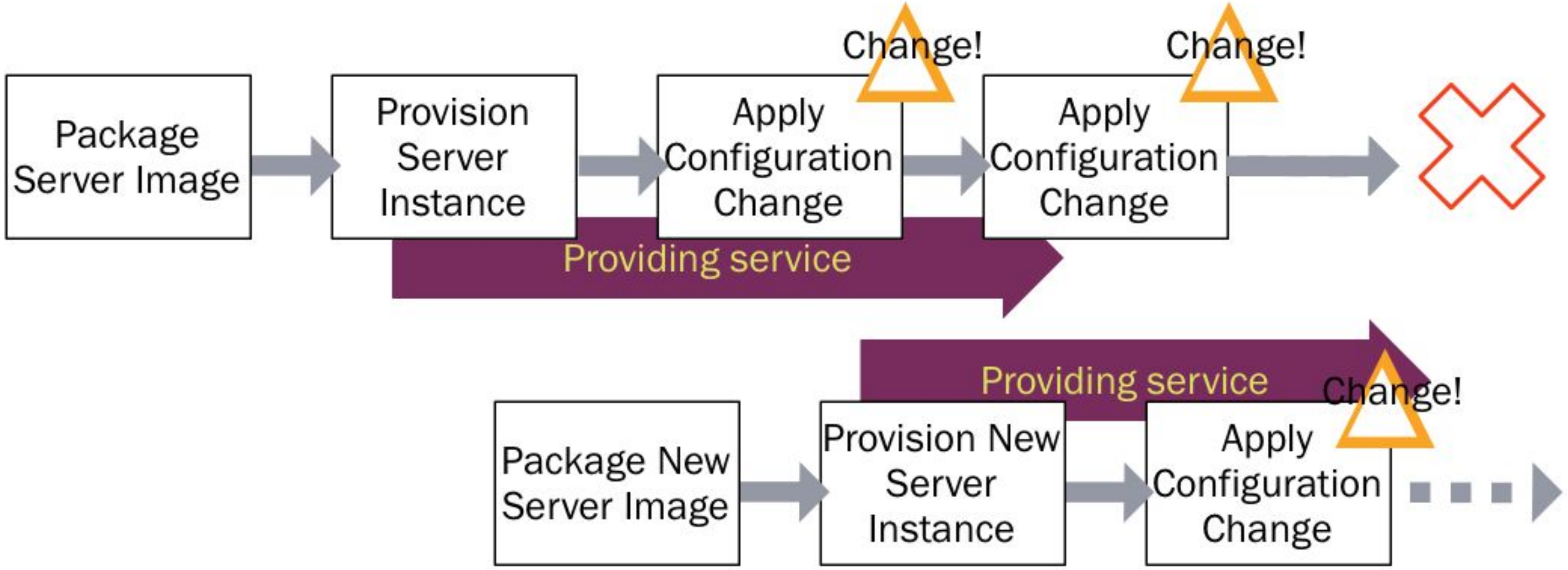
There are new ways that are emerging

The future is already here - it is just not evenly distributed

Phoenix server

Taking it a step further

Phoenix server

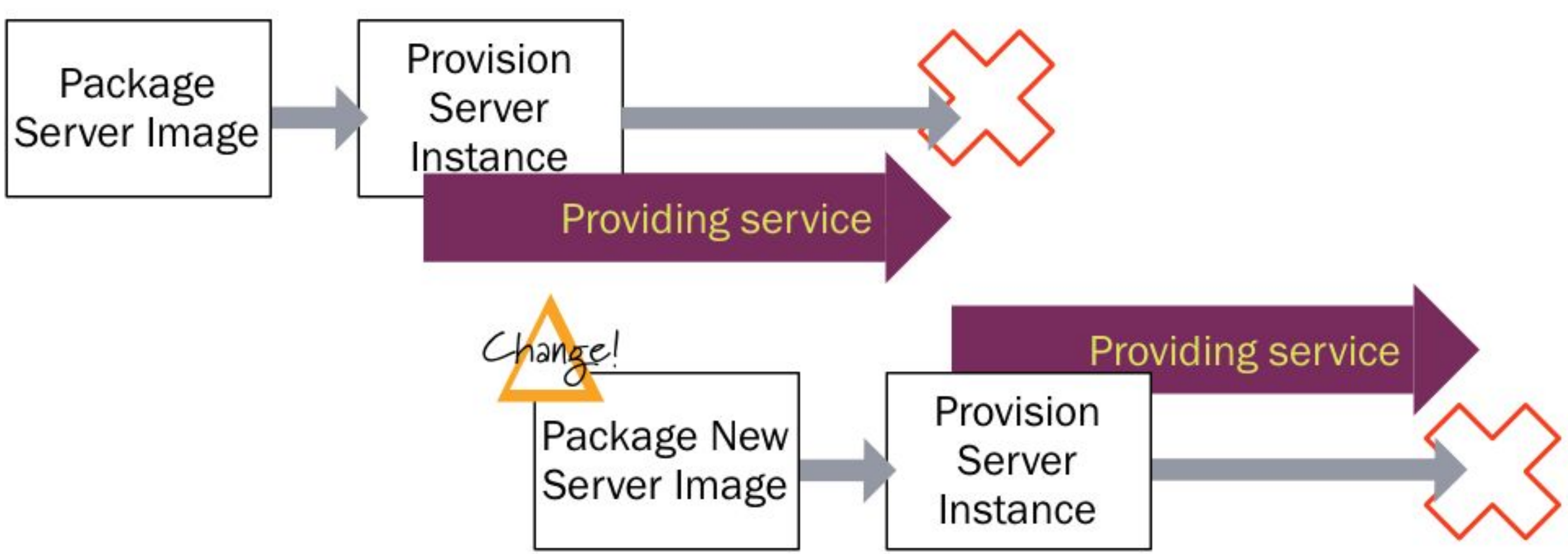

Package Server Image
Provision Server Instance
Apply Configuration Change
Change!
Apply Configuration Change
Change!
Providing service
Providing service
Change!
Package New Server Image
Provision New Server Instance
Apply Configuration Change

https://martinfowler.com/bliki/ImmutableServer.html

Immutable server

Leaving no chance for configuration drift to accumulate

Immutable server



https://martinfowler.com/bliki/ImmutableServer.html

Benefits of immutable infra

Minimal configuration drift

Scalability

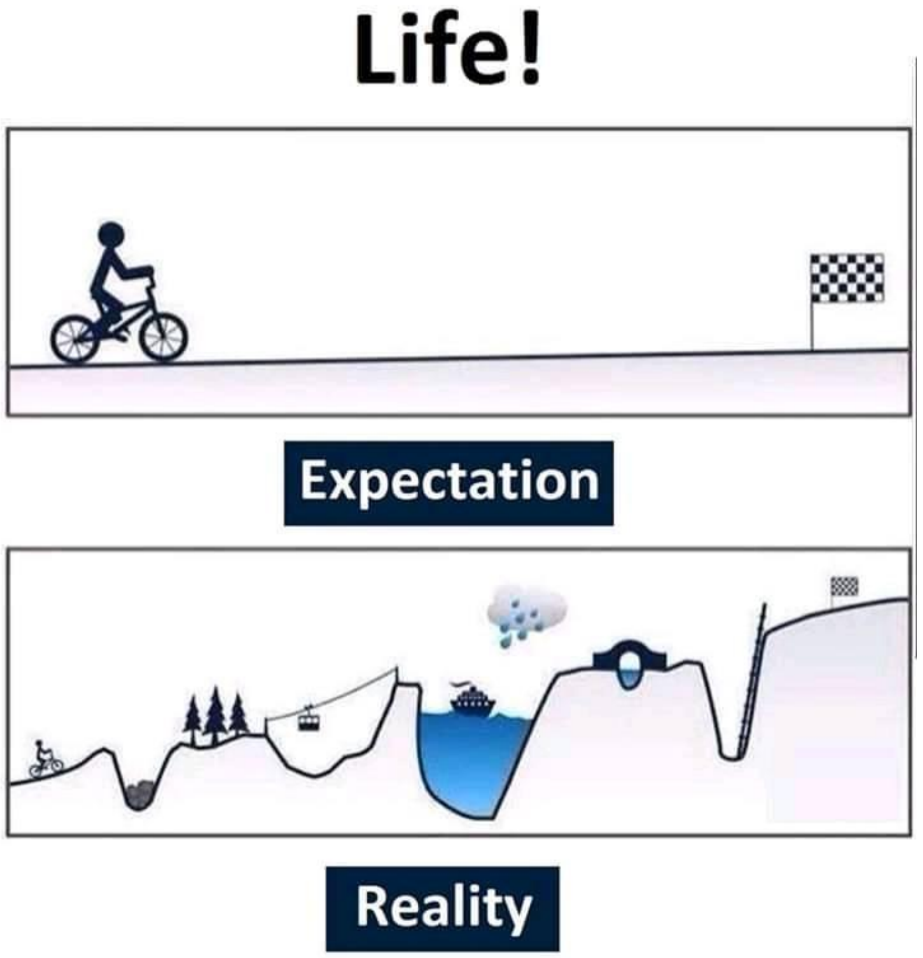
Security

Cost

So if I kill my servers often enough and provision them with Ansible then I’m doing immutable configuration as code?

So if I kill my servers often enough and provision them with Ansible then I'm doing immutable configuration as code?

Is it good enough?



https://memezila.com/wp-content/Life-expectation-vs-reality-meme-1556.png

Problem = IT change management automation

Context = bare metal / vm

Method = immutable infra as code

Tool = ???

Implementing immutable infa

12 factor app

III. Config

Store config in the environment

IV. Backing services

Treat backing services as attached resources

V. Build, release, run

Strictly separate build and run stages

XI. Logs

Treat logs as event streams

https://12factor.net/

Things to consider

How to deal with secrets and dynamic configuration?

How do we add sensitive information into the image. What if it is being stolen? How to deliver dynamic config?

How to keep people away?

People is a primary source of configuration drift

What if I need to troubleshoot?

Getting ready to things not going according to the plan

Secrets

Secrets

TLS certs

Cloud access

Identity

3rd party services creds

Solving secret zero problem

- Use IAM instance profiles
- Encrypt and bake in (don’t bake in encryption key!)
- Pull secrets in (Vault, AWS SSM)
- Generate temporary credentials where possible (Vault, AWS IAM)
- Pull in environment specific configuration (Consul, AWS SSM) or pass it as user data

People

People

SSH

RDP

VNC

EC2 Instance Connect

Make them trace

- Keep the possibility to login but remove the access
- Opening access should be a traceable event
- Kill the VM after someone touched it
- Automate purification process

Debug

Get out everything you need

- Stream out everything you might need (logs, metrics)
- Record system calls/sessions (think auditd)
- Boot up test VM to test what you need
- If you have to login to production VM but make sure to kill it after

Best practices

Build

Test

Version

Automate

CI-thinking for infra

- Keep all dependencies versioned
- Automate the build and let machines produce images, keep people out of the loop
- Automate testing of your infra
- Build and promote instead of rebuild
- Deploy often (change-driven)
- Deploy regularly (schedule-driven)

Putting all of it to practice

Build

HashiCorp Packer - to produce images (AMI, VMDK, ISO, QCOW etc)

Deploy

Terraform (cloud, vCenter Server, ESXi, OpenStack), PXE (baremetal)

Secure

Auditbeat, HashiCorp Vault, AWS Secrets Manager

Debug

Prometheus, Logstash/Beats, CloudWatch Logs/Agent

Configuration management

Part of server/vm image build

Server provisioning

Server/vm image configured during image creation

Application deployment

Deployed as part of server/vm image

Task scheduling/orchestration

???

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

The future is already here - it is just not evenly distributed

Containers

You are not configuring your running containers using Ansible, don’t ya?

“

Problem = IT change management automation

Context = containers

Method = ???

Tool = ???

Containers

Immutable by default

Enclose app related entropy

Stateless

Disposable

Container Orchestrators

Kubernetes

AWS ECS

HashiCorp Nomad

Container OS

AWS Bottlerocket

Red Hat Atomic

CoreOS 💀

k3OS

“

Problem = IT change management automation

Context = containers / k8s

Method = GitOps(Infra as Data) / Infra as code

Tool = ArgoCD / Helm / Terraform

Configuration management

Part of container build

Server provisioning

Server/vm image configured during image creation, not much to configure, self updating, minimal

Application deployment

Deployed by container orchestrator

Task scheduling/orchestration

???

Serverless?

MicroVM Unikernels

AWS Firecracker

OSv

includeOS

MirageOS

What can we learn from K8S?

A possible next step? Will cloud providers automate us out of our jobs?

Are we going backwards?

Is GitOps a new Chef?

How do we do immutable infra for K8S cluster?

Five theses

The only constant is change

Configuration synchronization is a necessary evil

Immutable infrastructure is a current best practice for today

There are new ways that are emerging

The future is already here - it is just not evenly distributed

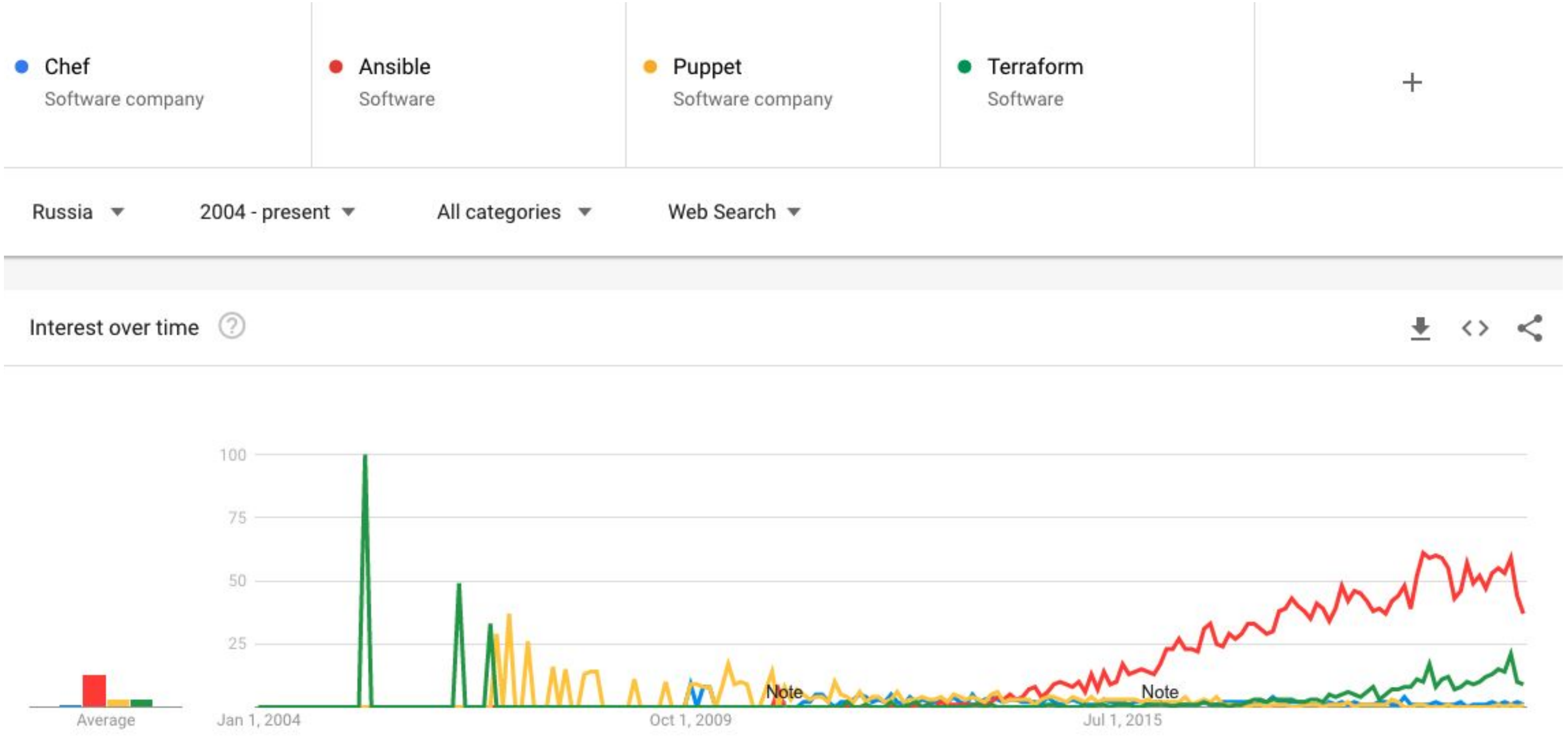
You decide what you do

Hard to remove tools

High demand for specialists

Technology bets are hard

Cloud is coming for you



● Chef
Software company

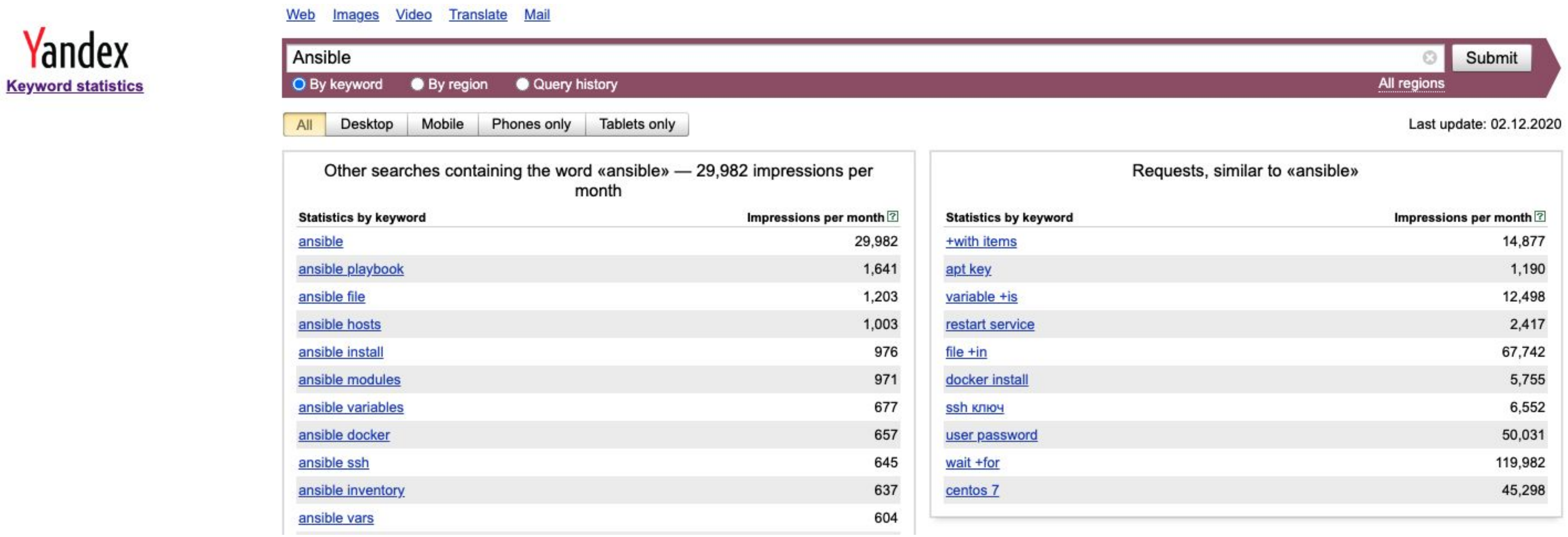
● Ansible
Software

● Puppet
Software company

● Terraform
Software

Russia | 2004 - present | All categories | Web Search

Interest over time



Yandex
Keyword statistics

Web Images Video Translate Mail

Ansible

By keyword [✓] By region Query history All regions

Submit

[✓ All] Desktop Mobile Phones only Tablets only

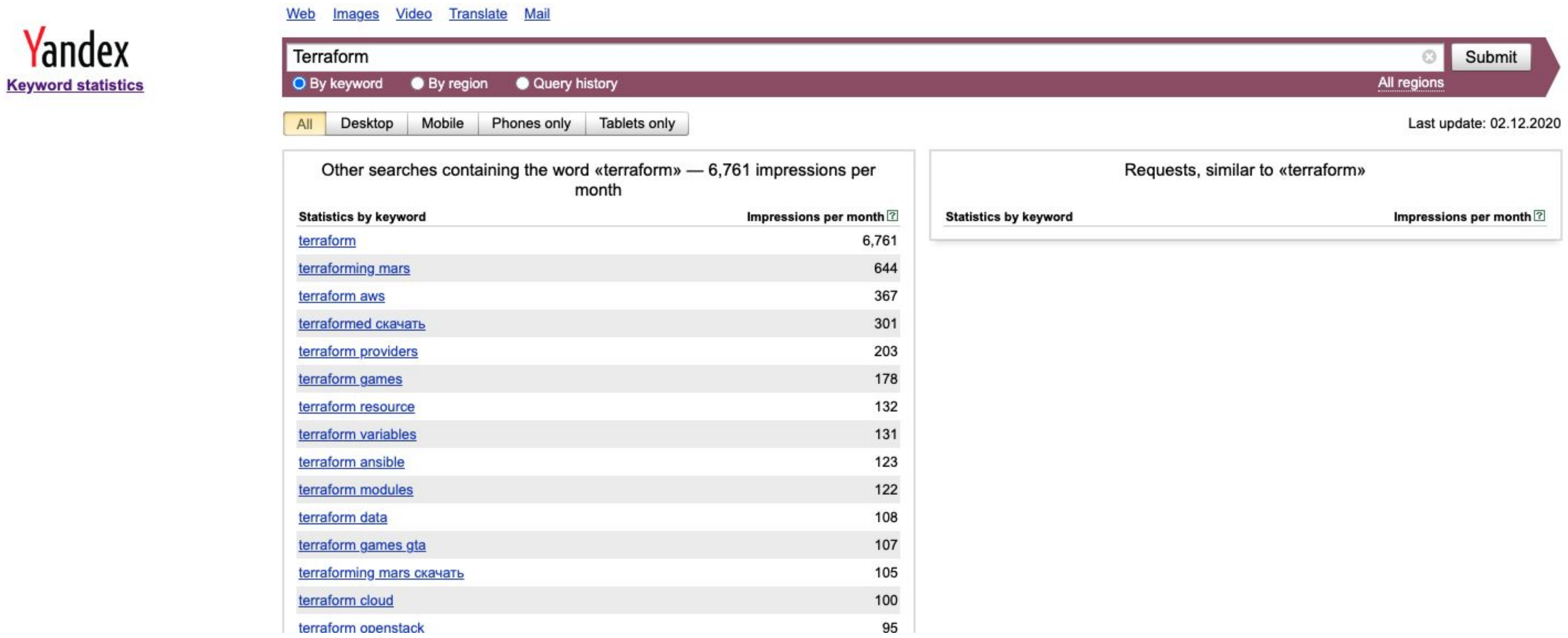
Last update: 02.12.2020

Other searches containing the word «ansible» — 29,982 impressions per month

Statistics by keyword	Impressions per month
ansible	29,982
ansible playbook	1,641
ansible file	1,203
ansible hosts	1,003
ansible install	976
ansible modules	971
ansible variables	677
ansible docker	657
ansible ssh	645
ansible inventory	637
ansible vars	604

Requests, similar to «ansible»

Statistics by keyword	Impressions per month
+with items	14,877
apt key	1,190
variable +is	12,498
restart service	2,417
file +in	67,742
docker install	5,755
ssh ключ	6,552
user password	50,031
wait +for	119,982
centos 7	45,298



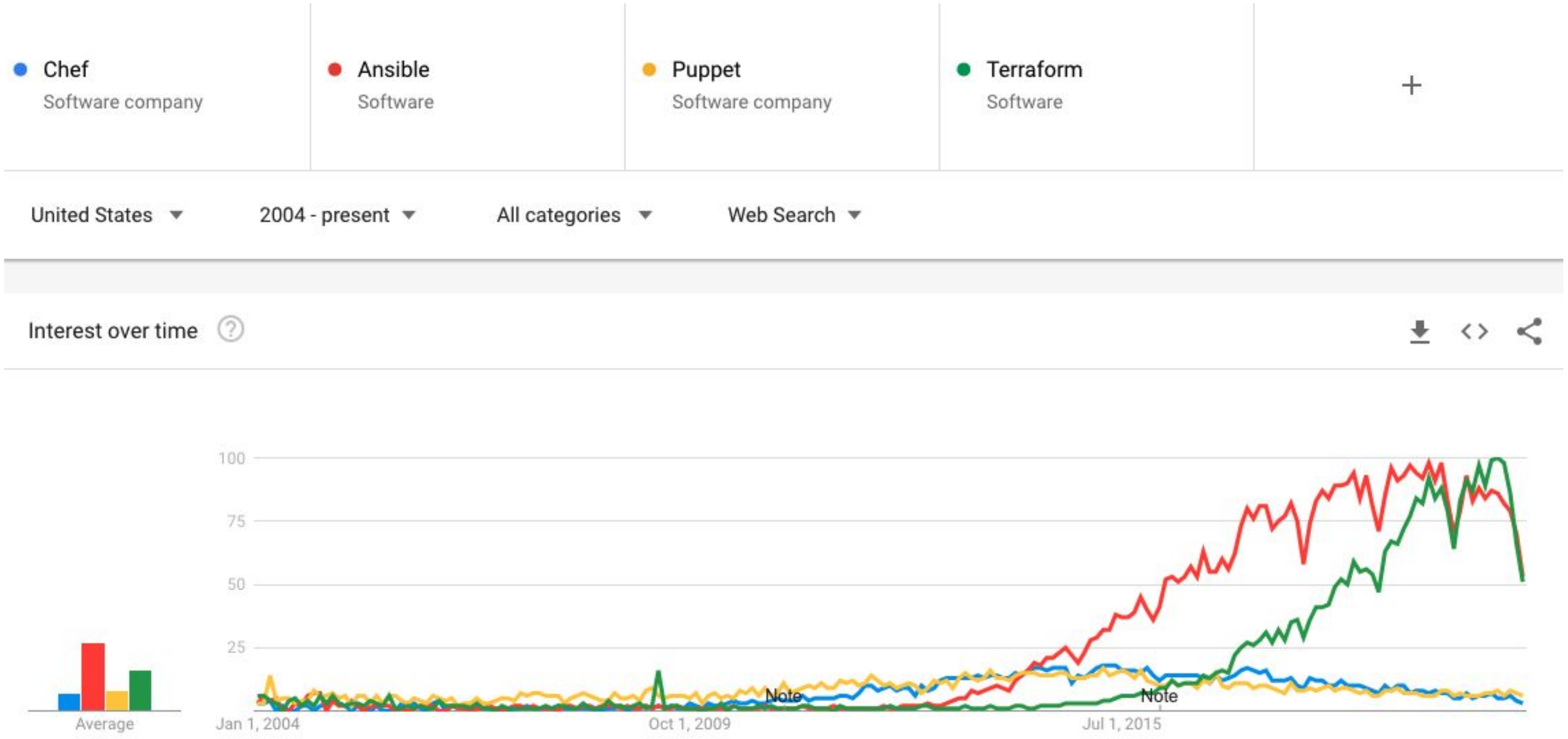
Last update: 02.12.2020

Other searches containing the word «terraform» — 6,761 impressions per month

Statistics by keyword	Impressions per month
terraform	6,761
terraforming mars	644
terraform aws	367
terraformed скачать	301
terraform providers	203
terraform games	178
terraform resource	132
terraform variables	131
terraform ansible	123
terraform modules	122
terraform data	108
terraform games gta	107
terraforming mars скачать	105
terraform cloud	100
terraform openstack	95

Requests, similar to «terraform»

Statistics by keyword	Impressions per month


● Chef
Software company

● Ansible
Software

● Puppet
Software company

● Terraform
Software

United States | 2004 - present | All categories | Web Search

Interest over time

Recap

“

Problem

Context

Method

Tool

FivexL

Problem = IT change management automation

Context = bare metal / vm

Method = infra as code / configuration sync

Tool = Ansible / Puppet / Chef etc

Problem = IT change management automation

Context = bare metal / vm

Method = immutable infra as code

Tool = Packer / PXE / Terraform

Problem = IT change management automation

Context = containers / k8s

Method = GitOps(Infra as Data) / Infra as code

Tool = ArgoCD / Helm / Terraform

Thank you

FivexL

@andrey9kin
https://fivexl.io
https://andreydevyatkin.com
https://www.linkedin.com/in/andreydevyatkin/
https://devsecops.fm