MVP Dagen

mvpdagen.no

Takk til våre sponsorer

tendfor™ Microsoft Intranet Global power tilpasset norske forhold

mvpdagen.no

Hey!

Jeg er en MVP

Jeg er her fordi jeg ønsker å dele research som gir alle forutsetninger for å bygge sikre miljøer.

Du finner meg på @karimscloud

mvpdagen.no

My background

intility

Norges Bank Investment Management

O3 CYBER

Elevating Your Cloud Security Game.

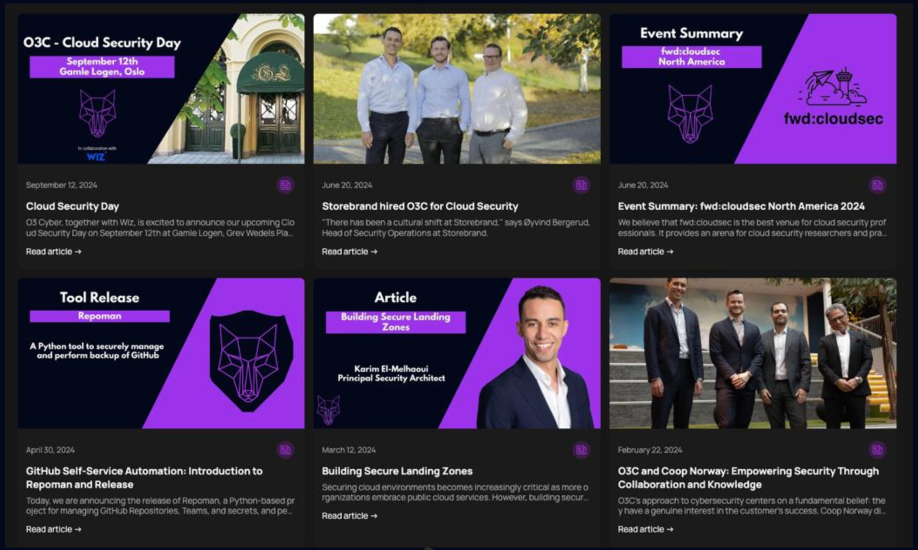
Trusted experts helping you raise the bar in cloud security.

fwd:cloudsec
Brussels 2024

O3 CYBER

To guide the global community toward a more secure utilization of cloud technologies

Community



September 12, 2024

Cloud Security Day

O3 Cyber, together with Wiz, is excited to announce our upcoming Cloud Security Day on September 12th at Gamle Logen, Grev Wedels Pla...

Read article →

June 20, 2024

Storebrand hired O3C for Cloud Security

"There has been a cultural shift at Storebrand," says Øyvind Bergerud, Head of Security Operations at Storebrand.

Read article →

June 20, 2024

Event Summary: fwd:cloudsec North America 2024

We believe that fwd:cloudsec is the best venue for cloud security professionals. It provides an arena for cloud security researchers and pra...

Read article →

April 30, 2024

GitHub Self-Service Automation: Introduction to Repoman and Release

Today, we are announcing the release of Repoman, a Python-based project for managing GitHub Repositories, Teams, and secrets, and pe...

Read article →

March 12, 2024

Building Secure Landing Zones

Securing cloud environments becomes increasingly critical as more organizations embrace public cloud services. However, building secur...

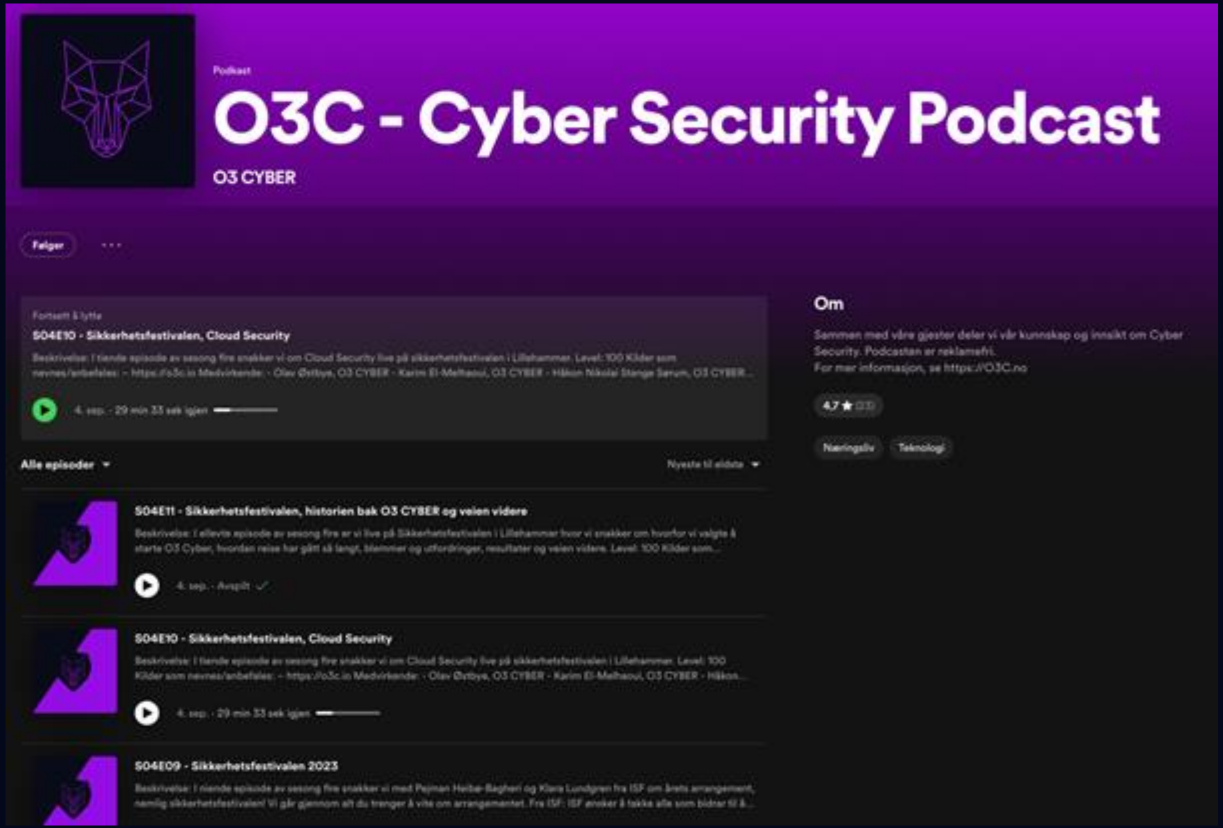
Read article →

February 22, 2024

O3C and Coop Norway: Empowering Security Through Collaboration and Knowledge

O3C's approach to cybersecurity centers on a fundamental belief: they have a genuine interest in the customer's success. Coop Norway di...

Read article →



Podkast

O3C - Cyber Security Podcast

O3 CYBER

Følger ···

Fortsett å lytte

S04E10 - Sikkerhetsfestivalen, Cloud Security

Beskrivelse: I tiende episode av sesong fire snakker vi om Cloud Security live på sikkerhetsfestivalen i Lillehammer. Level: 100 Kilder som nevnes/anbefales: – https://o3c.io Medvirkende: - Olav Østbye, O3 CYBER - Karim El-Melhaoui, O3 CYBER - Håkon Nikolai Stange Sørum, O3 CYBER...

4. sep. · 29 min 33 sek igjen

Om

Sammen med våre gjester deler vi vår kunnskap og innsikt om Cyber Security. Podcasten er reklamefri.
For mer informasjon, se https://O3C.no

4,7 ★ (23)

Næringsliv Teknologi

Alle episoder ▾ Nyeste til eldste ▾

S04E11 - Sikkerhetsfestivalen, historien bak O3 CYBER og veien videre

Beskrivelse: I ellevte episode av sesong fire er vi live på Sikkerhetsfestivalen i Lillehammer hvor vi snakker om hvorfor vi valgte å starte O3 Cyber, hvordan reise har gått så langt, blemmer og utfordringer, resultater og veien videre. Level: 100 Kilder som...

4. sep. · Avspilt ✓

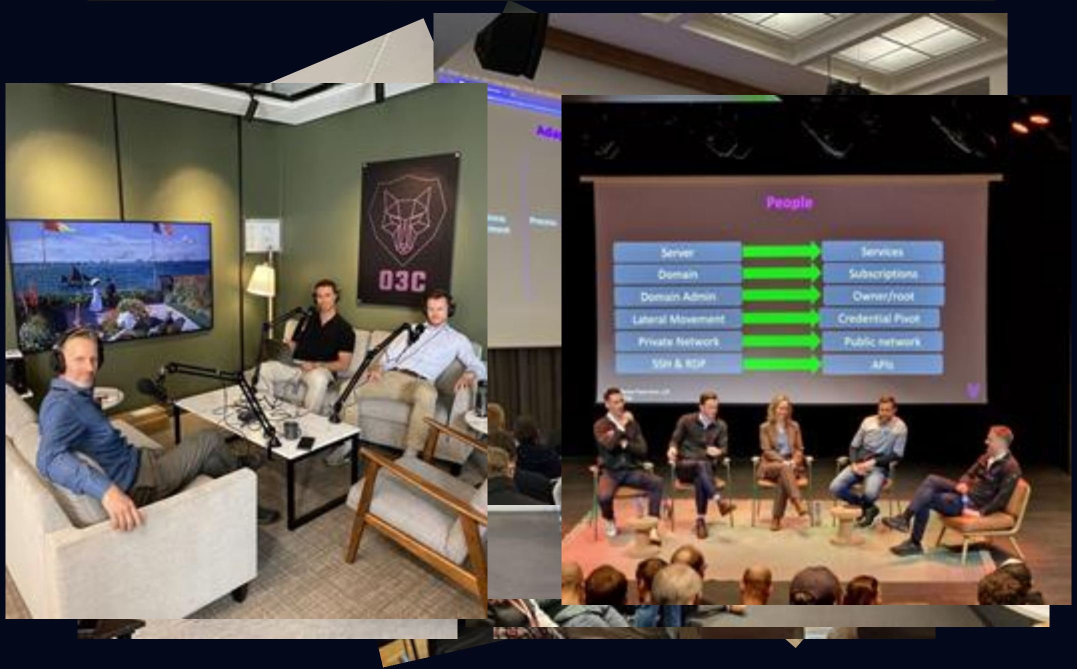
S04E10 - Sikkerhetsfestivalen, Cloud Security

Beskrivelse: I tiende episode av sesong fire snakker vi om Cloud Security live på sikkerhetsfestivalen i Lillehammer. Level: 100 Kilder som nevnes/anbefales: – https://o3c.io Medvirkende: - Olav Østbye, O3 CYBER - Karim El-Melhaoui, O3 CYBER - Håkon...

4. sep. · 29 min 33 sek igjen

S04E09 - Sikkerhetsfestivalen 2023

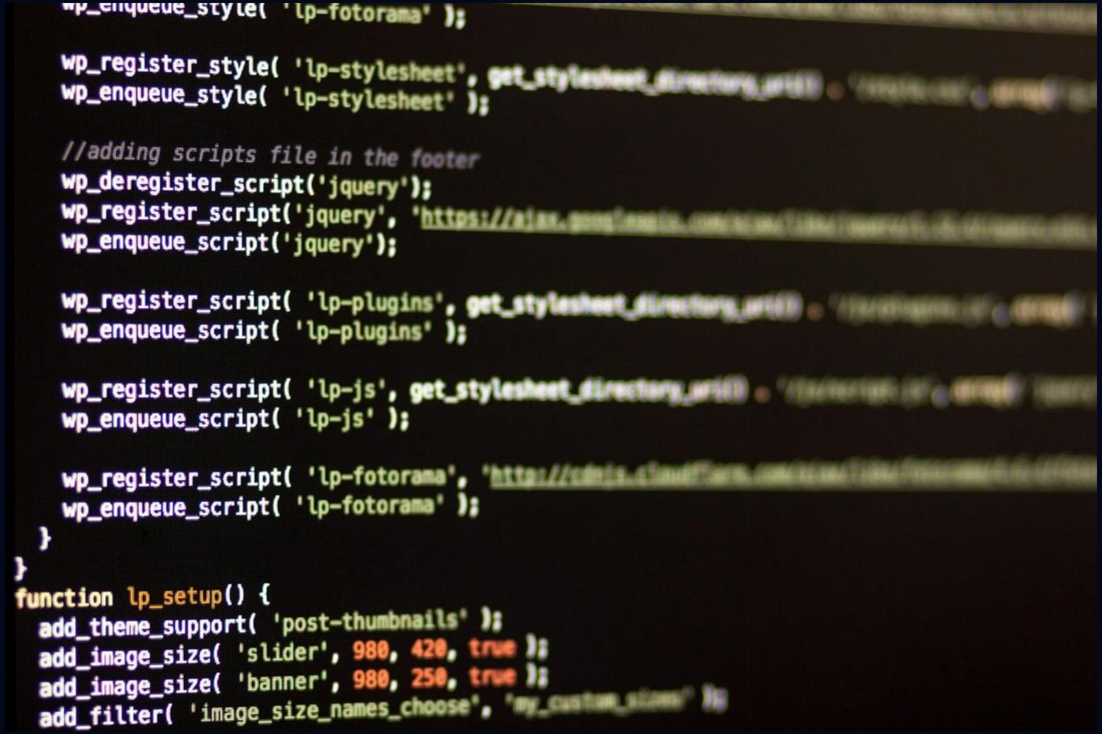
Beskrivelse: I niende episode av sesong fire snakker vi med Pejman Heibe-Bagheri og Klara Lundgren fra ISF om årets arrangement, nemlig sikkerhetsfestivalen! Vi går gjennom alt du trenger å vite om arrangementet. Fra ISF: ISF ønsker å takke alle som bidrar til å...



Set the stage

We’re on a cloud journey

Everything as Code



```
    wp_enqueue_style( 'lp-fotorama' );

    wp_register_style( 'lp-stylesheet', get_stylesheet_directory_uri() . [illegible]
    wp_enqueue_style( 'lp-stylesheet' );

    //adding scripts file in the footer
    wp_deregister_script('jquery');
    wp_register_script('jquery', 'https://ajax.googleapis.com/[illegible]
    wp_enqueue_script('jquery');

    wp_register_script( 'lp-plugins', get_stylesheet_directory_uri() . [illegible]
    wp_enqueue_script( 'lp-plugins' );

    wp_register_script( 'lp-js', get_stylesheet_directory_uri() . [illegible]
    wp_enqueue_script( 'lp-js' );

    wp_register_script( 'lp-fotorama', 'http://cdnjs.cloudflare.com/[illegible]
    wp_enqueue_script( 'lp-fotorama' );
  }
}
function lp_setup() {
  add_theme_support( 'post-thumbnails' );
  add_image_size( 'slider', 980, 420, true );
  add_image_size( 'banner', 980, 250, true );
  add_filter( 'image_size_names_choose', 'my_custom_sizes' );
```

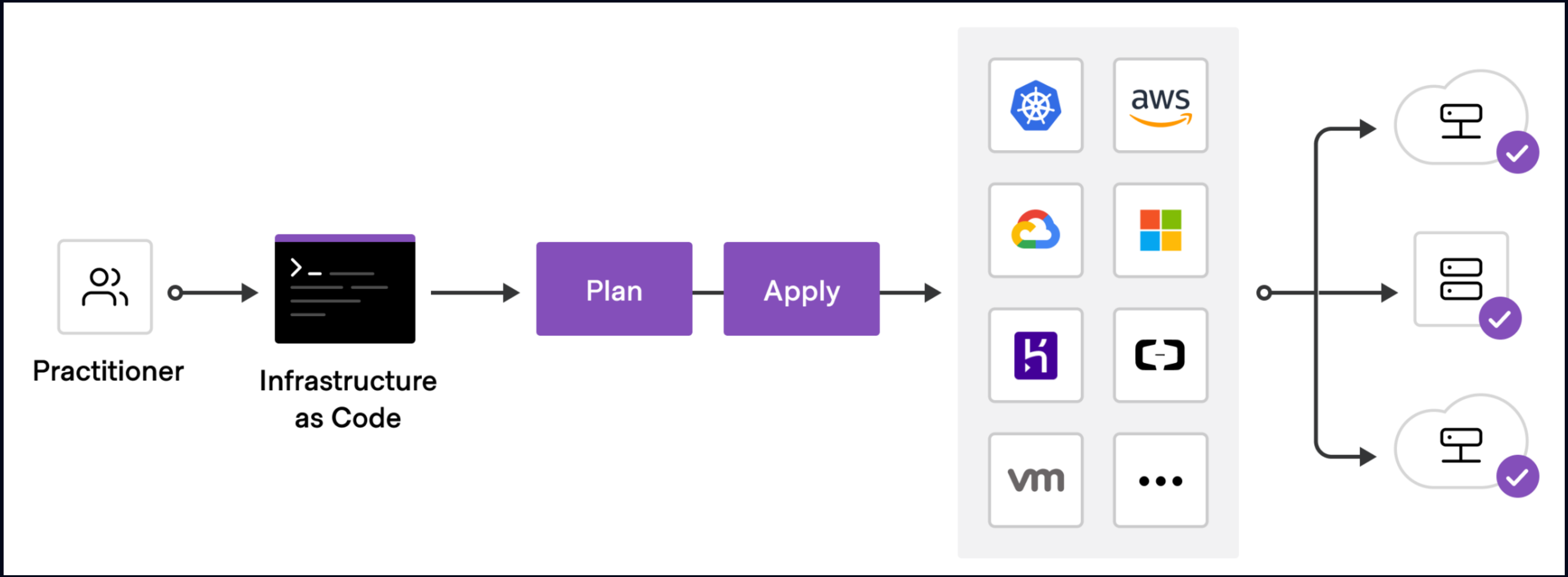
How about securing cloud?

- Segmentation
- Secrets Management
- Access Management
- Continuous Integration
- Resource Configuration

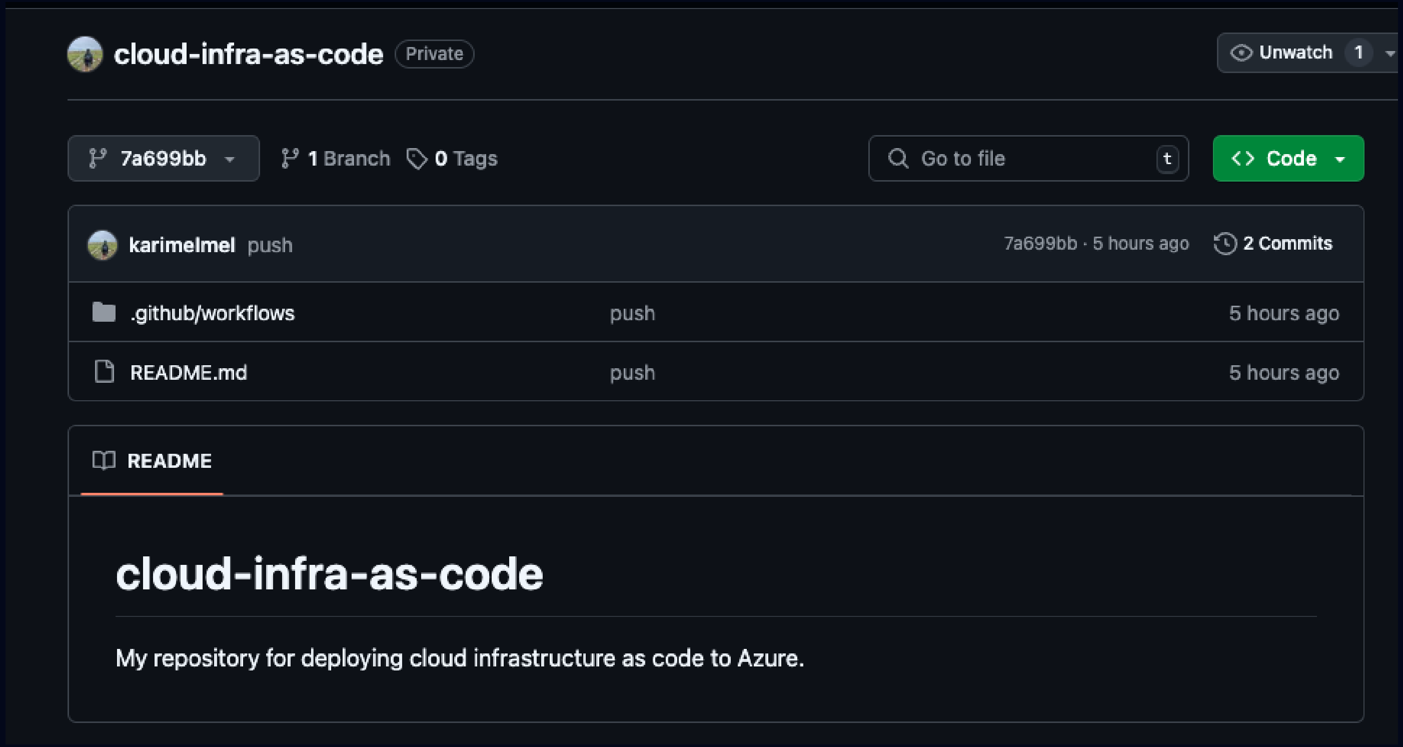
Anti-patterns

- Clickops
- Deploying from all laptops
- Overly permissive
- Poor configuration control

Continuous Integration



Git and CI



cloud-infra-as-code Private

karimelmel push 7a699bb · 5 hours ago 2 Commits

.github/workflows	push	5 hours ago
README.md	push	5 hours ago

README

cloud-infra-as-code

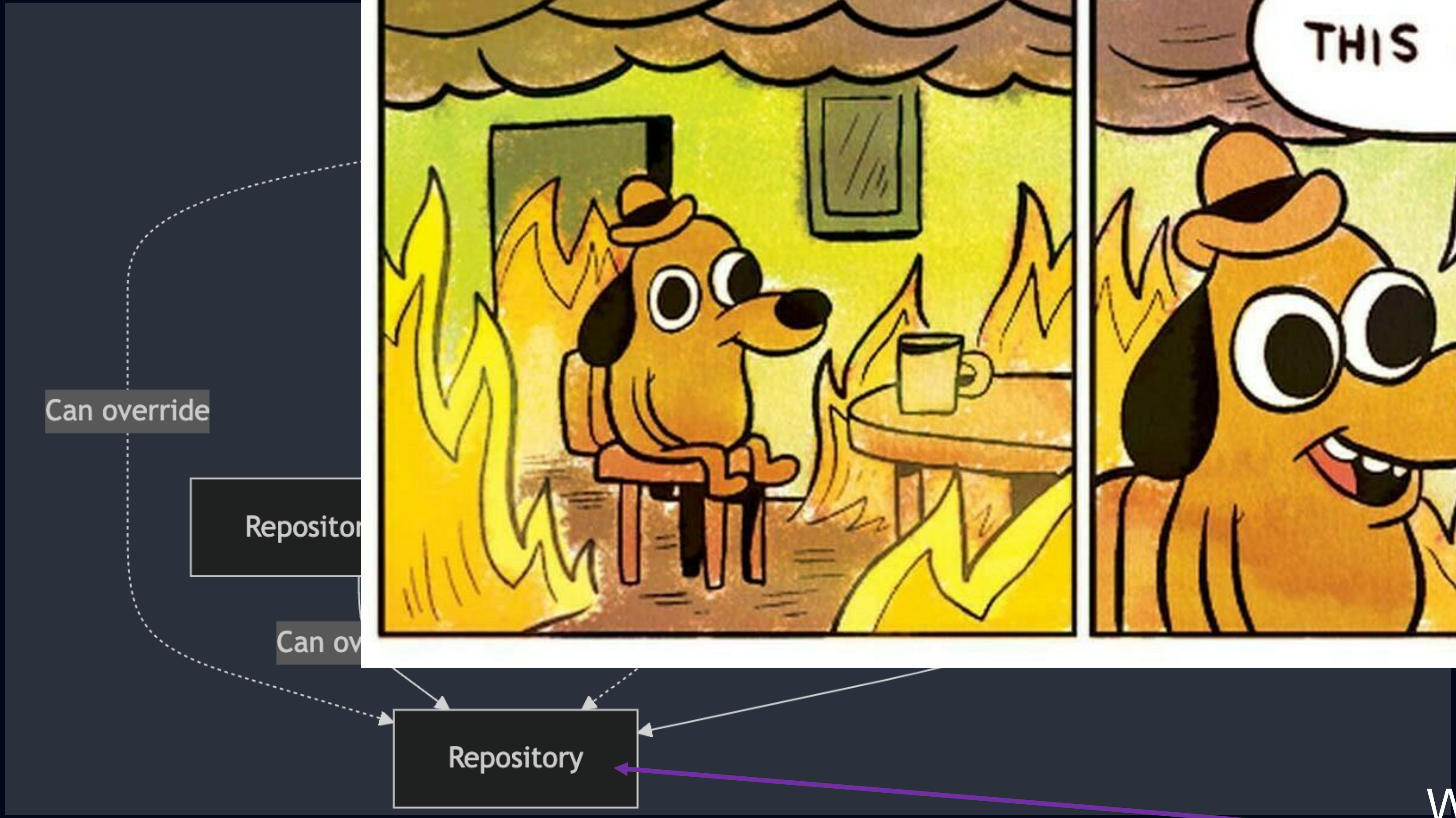
My repository for deploying cloud infrastructure as code to Azure.

Reality

Probably 2 – 5 developers

Finance guy

ho quit 2 years ago



s

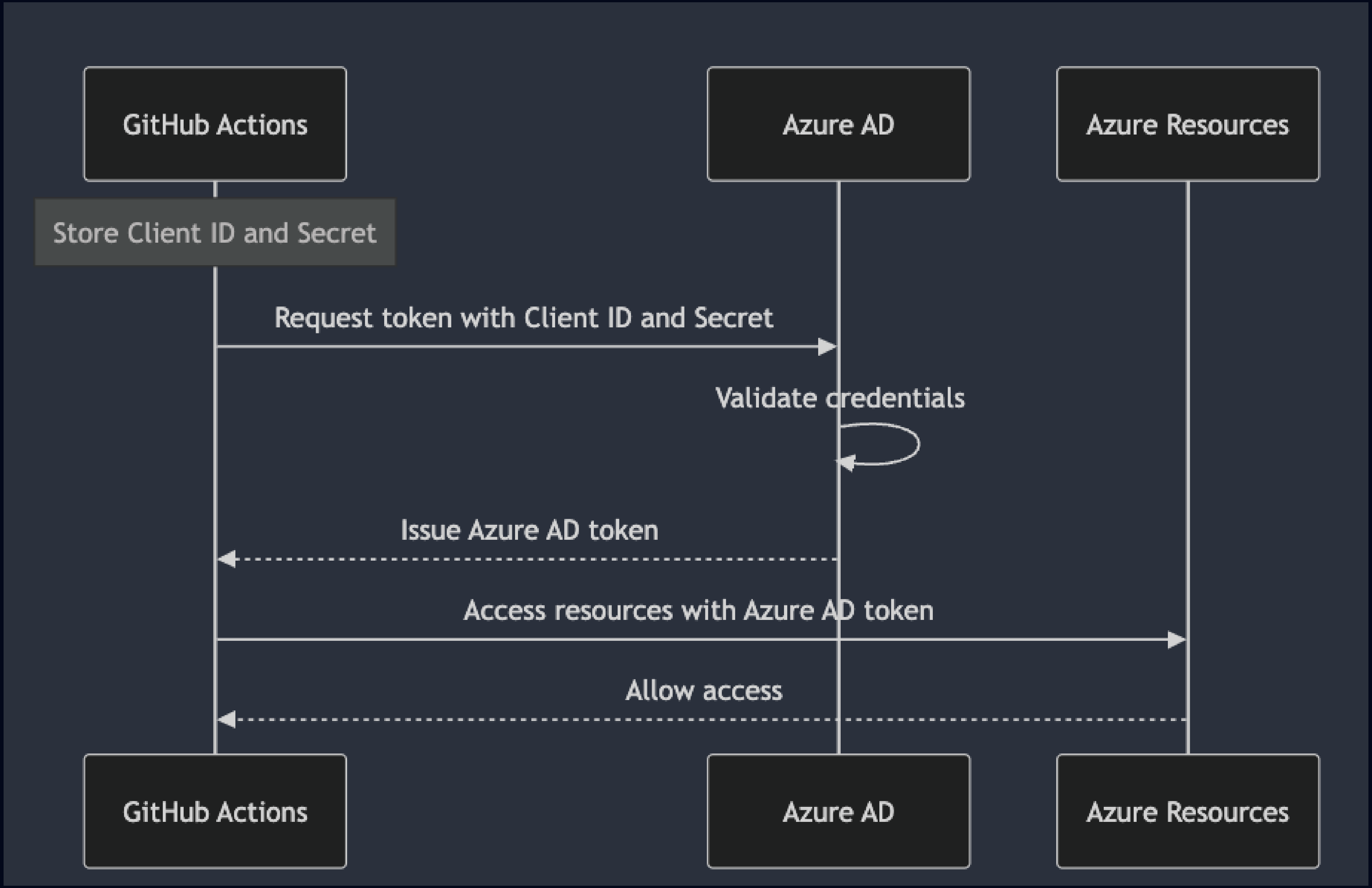
Usually least 4-6 people

What is Branch Protection??🤷‍♂️

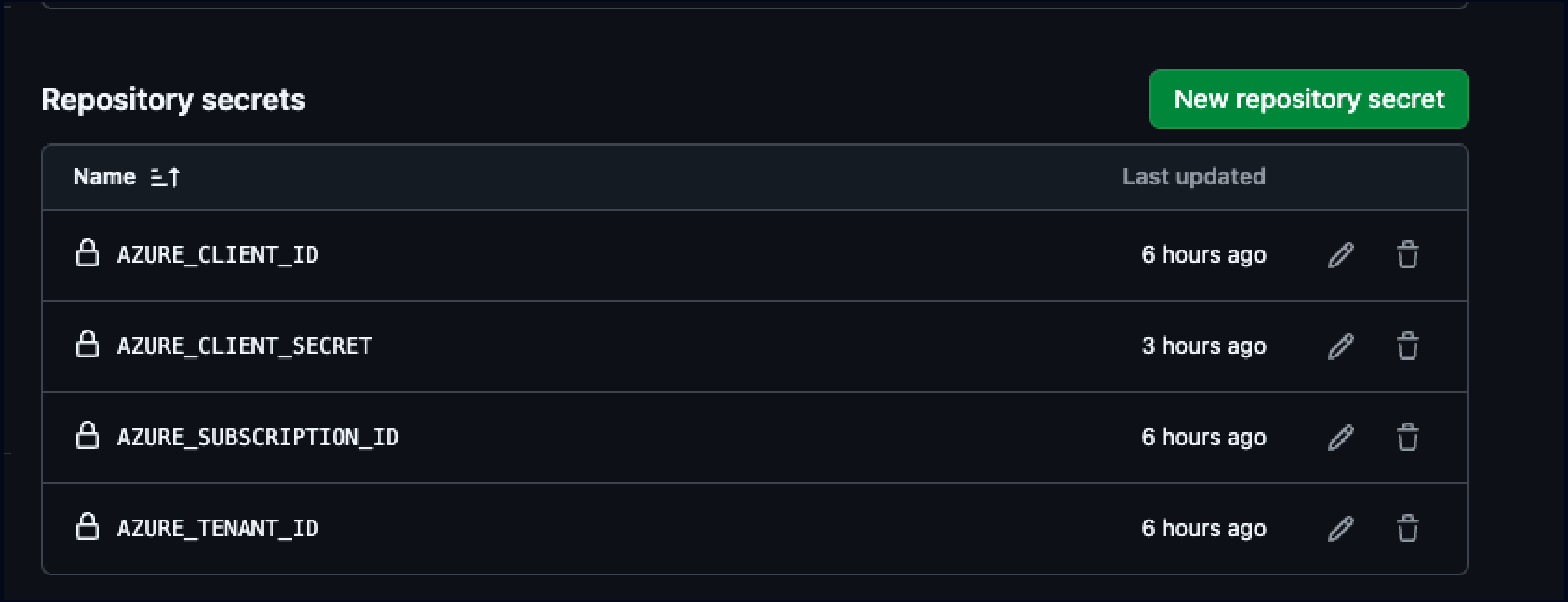
Secrets 🔑

Secrets Management and CI

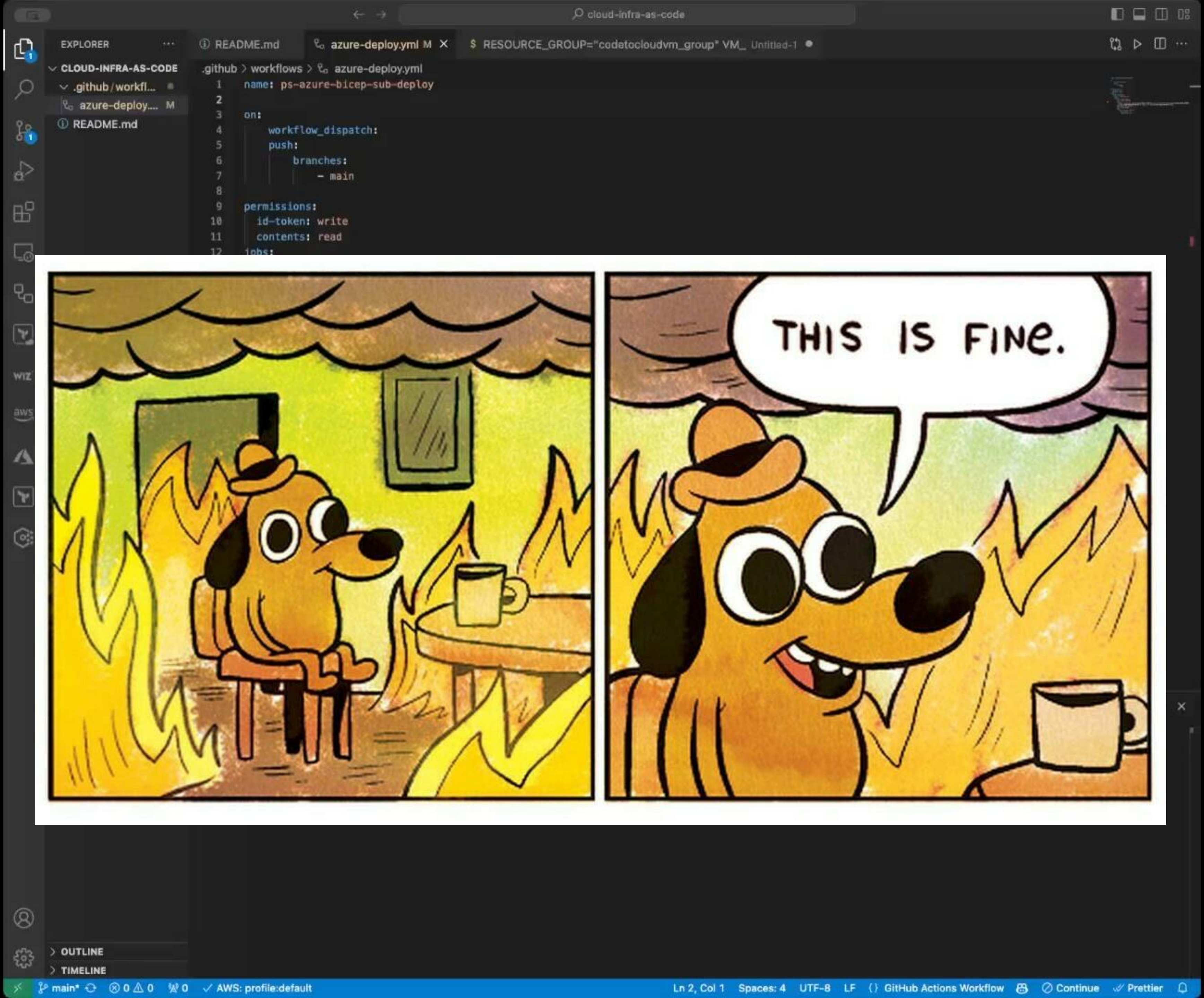
How do we authenticate from our CI?


GitHub Actions
Azure AD
Azure Resources
Store Client ID and Secret
Request token with Client ID and Secret
Validate credentials
Issue Azure AD token
Access resources with Azure AD token
Allow access
GitHub Actions
Azure AD
Azure Resources

Client Credentials Flow


Repository secrets

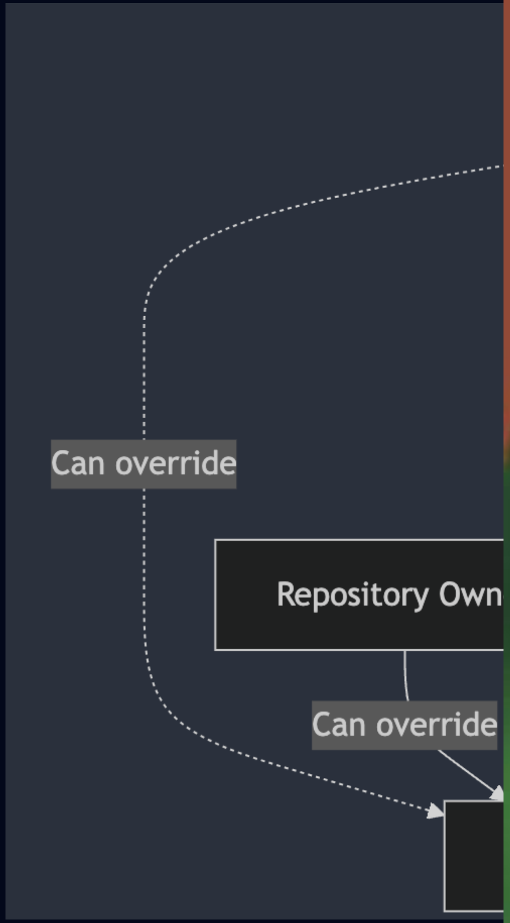
Name	Last updated
AZURE_CLIENT_ID	6 hours ago
AZURE_CLIENT_SECRET	3 hours ago
AZURE_SUBSCRIPTION_ID	6 hours ago
AZURE_TENANT_ID	6 hours ago


cloud-infra-as-code
EXPLORER
CLOUD-INFRA-AS-CODE
.github / workfl...
azure-deploy.... M
README.md
README.md
azure-deploy.yml M
RESOURCE_GROUP="codetocloudvm_group" VM_ Untitled-1
.github > workflows > azure-deploy.yml
OUTLINE
TIMELINE
main* ⊗ 0 ⚠ 0 ✓ AWS: profile:default
Ln 2, Col 1 Spaces: 4 UTF-8 LF {} GitHub Actions Workflow Continue Prettier

```
name: ps-azure-bicep-sub-deploy

on:
    workflow_dispatch:
    push:
        branches:
            - main

permissions:
  id-token: write
  contents: read
jobs:
```

The actual problem

Delete al
Push to



Delete all Code
Push to any CI

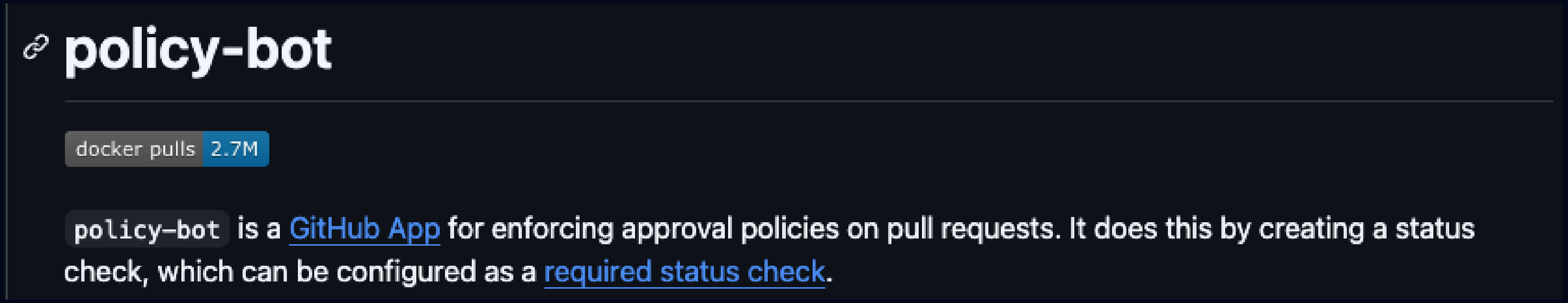
Own all code and rules
Push to any system

Exfil secrets
Resource Hijacking

Branch Protection

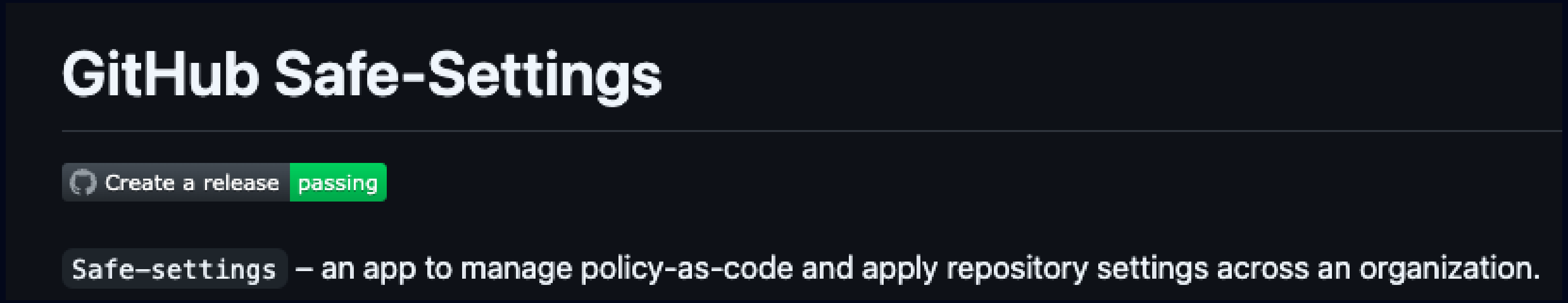
- Enforce a specific workflow
- PR reviews
- Status checks
- Push protection
- Code quality

Managing GitHub config at scale



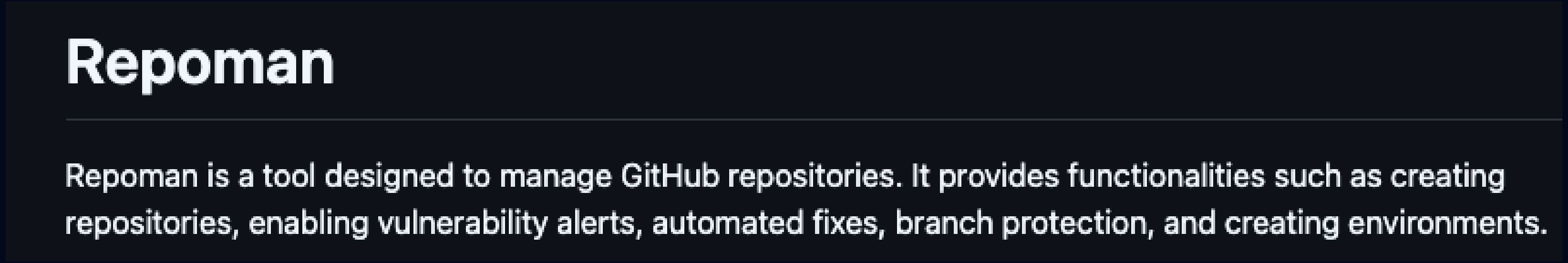
policy-bot

`policy-bot` is a GitHub App for enforcing approval policies on pull requests. It does this by creating a status check, which can be configured as a required status check.



GitHub Safe-Settings

`Safe-settings` – an app to manage policy-as-code and apply repository settings across an organization.



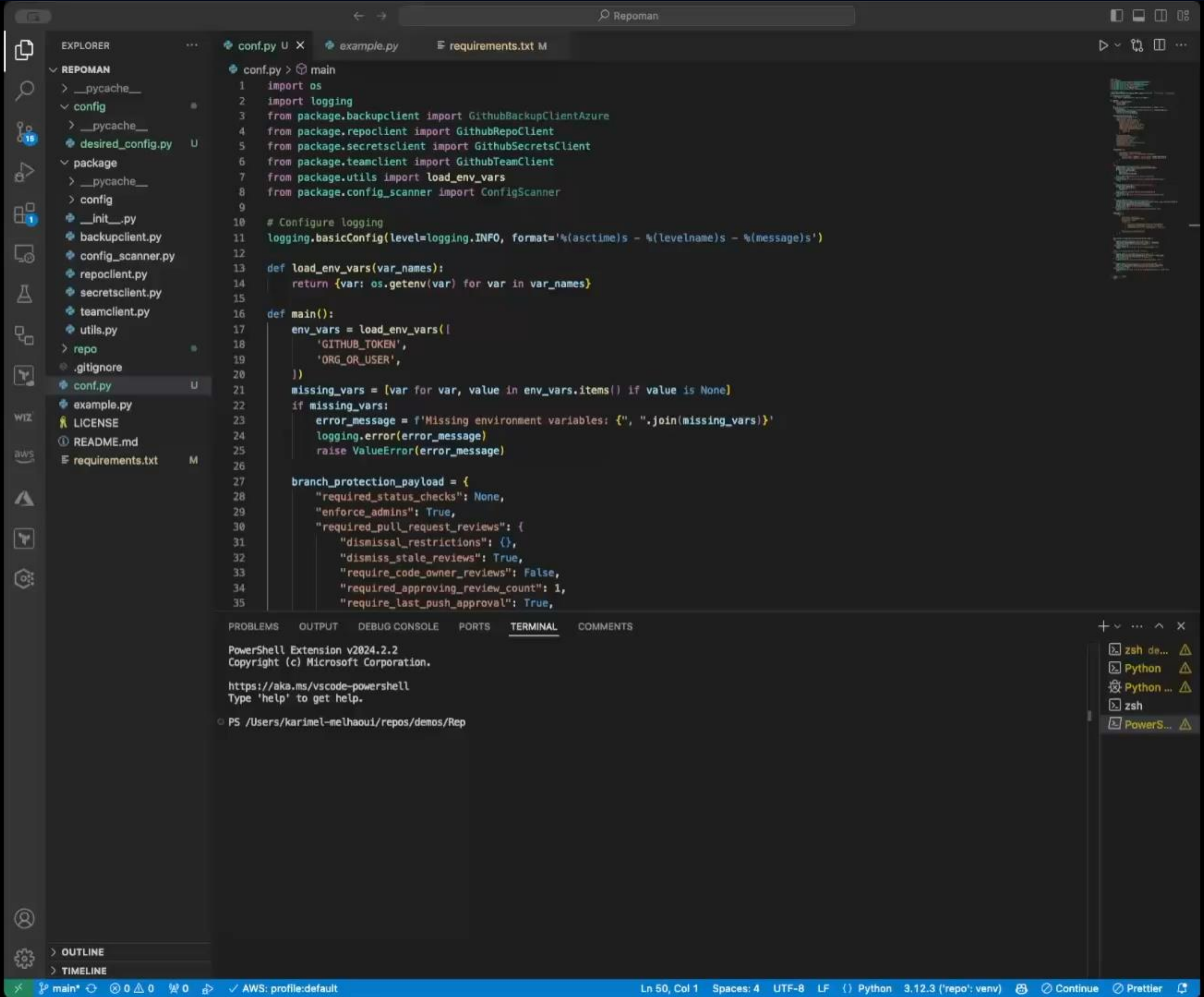
Repoman

Repoman is a tool designed to manage GitHub repositories. It provides functionalities such as creating repositories, enabling vulnerability alerts, automated fixes, branch protection, and creating environments.

Introducing Repoman

Why Repoman?

- Allows central control of repositories
- Allows encrypting secrets with public key from a repository or environment before uploading
- Allow associating GitHub Teams and IdP Groups
- **Remove the need for GitHub Admins**



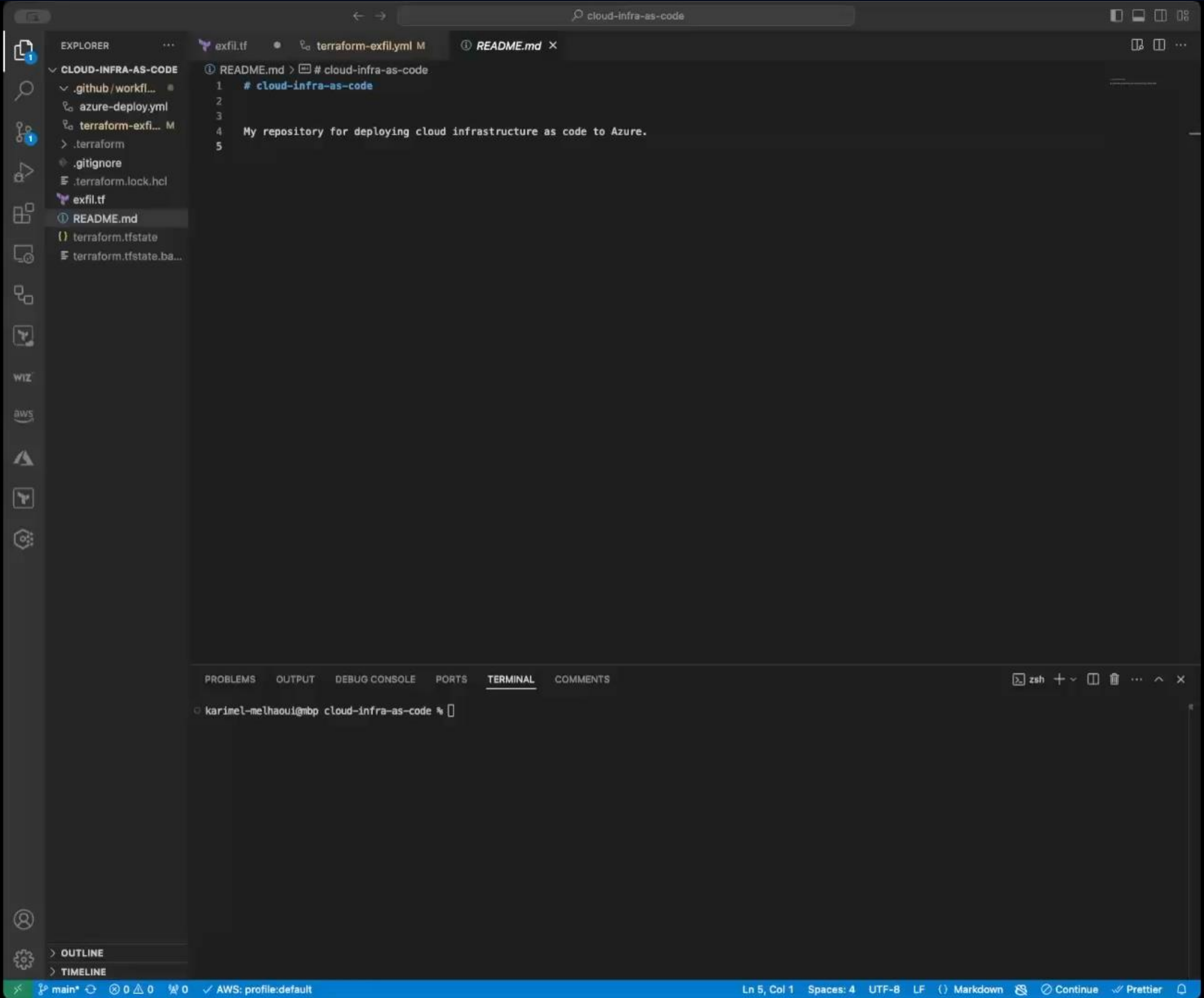
```python
import os
import logging
from package.backupclient import GithubBackupClientAzure
from package.repoclient import GithubRepoClient
from package.secretsclient import GithubSecretsClient
from package.teamclient import GithubTeamClient
from package.utils import load_env_vars
from package.config_scanner import ConfigScanner

# Configure logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

def load_env_vars(var_names):
    return {var: os.getenv(var) for var in var_names}

def main():
    env_vars = load_env_vars([
        'GITHUB_TOKEN',
        'ORG_OR_USER',
    ])
    missing_vars = [var for var, value in env_vars.items() if value is None]
    if missing_vars:
        error_message = f'Missing environment variables: {", ".join(missing_vars)}'
        logging.error(error_message)
        raise ValueError(error_message)

    branch_protection_payload = {
        "required_status_checks": None,
        "enforce_admins": True,
        "required_pull_request_reviews": {
            "dismissal_restrictions": {},
            "dismiss_stale_reviews": True,
            "require_code_owner_reviews": False,
            "required_approving_review_count": 1,
            "require_last_push_approval": True,
```

```
PowerShell Extension v2024.2.2
Copyright (c) Microsoft Corporation.

https://aka.ms/vscode-powershell
Type 'help' to get help.

PS /Users/karimel-melhaoui/repos/demos/Rep
```

Fresh repository with a branch protection

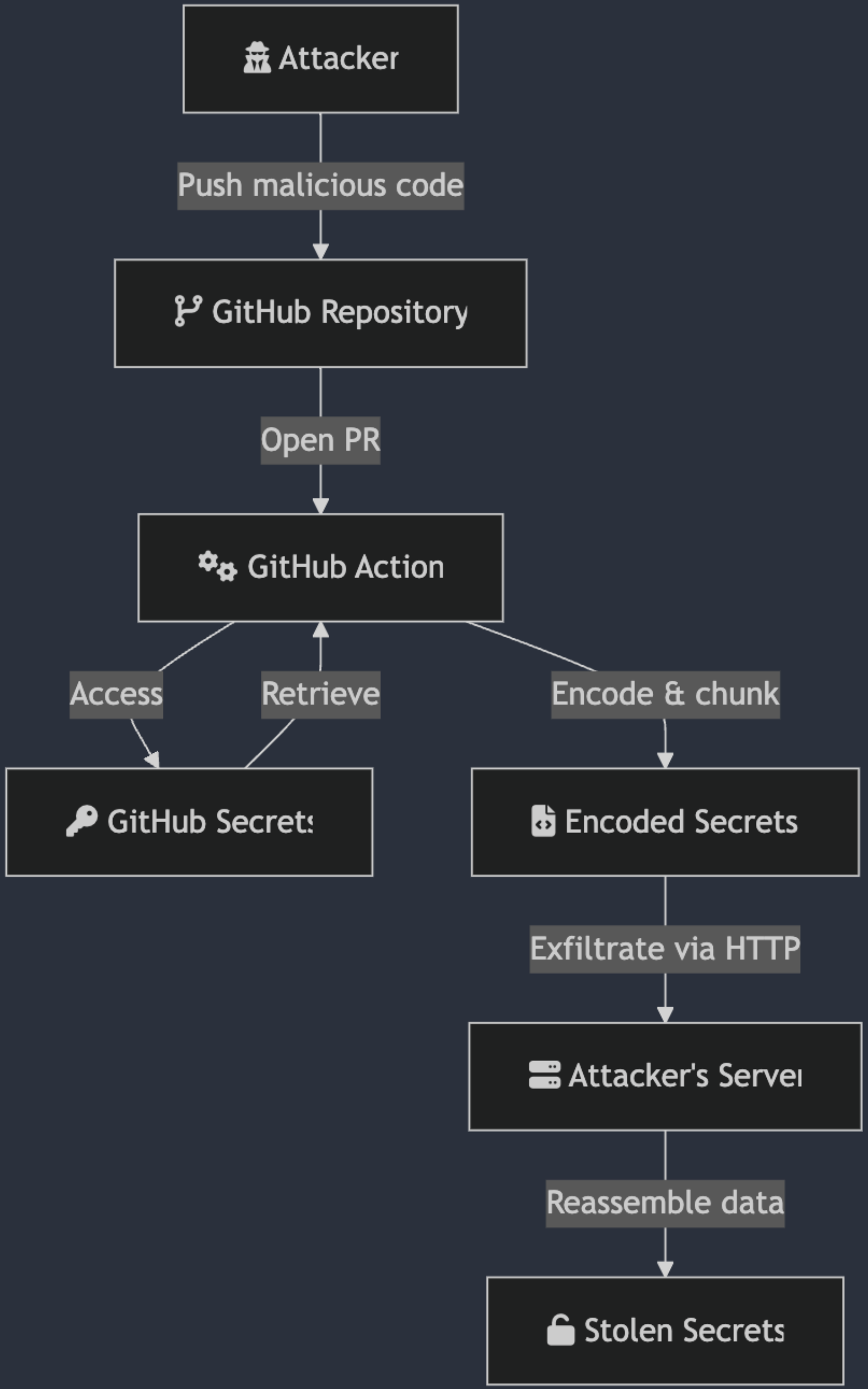
Control Bypass with a twist

- Bypass the Branch Protection by triggering from a Pull Request
- Use Terraform to exfiltrate secret over HTTP
    - … Because we can



```
# cloud-infra-as-code


My repository for deploying cloud infrastructure as code to Azure.

```

```
karimel-melhaoui@mbp cloud-infra-as-code %
```


Attacker
Push malicious code
GitHub Repository
Open PR
GitHub Action
Access
Retrieve
Encode & chunk
GitHub Secrets
Encoded Secrets
Exfiltrate via HTTP
Attacker's Server
Reassemble data
Stolen Secrets

What’s the issue here?

- Secret is accessible to a workflow triggered by a Pull Request!

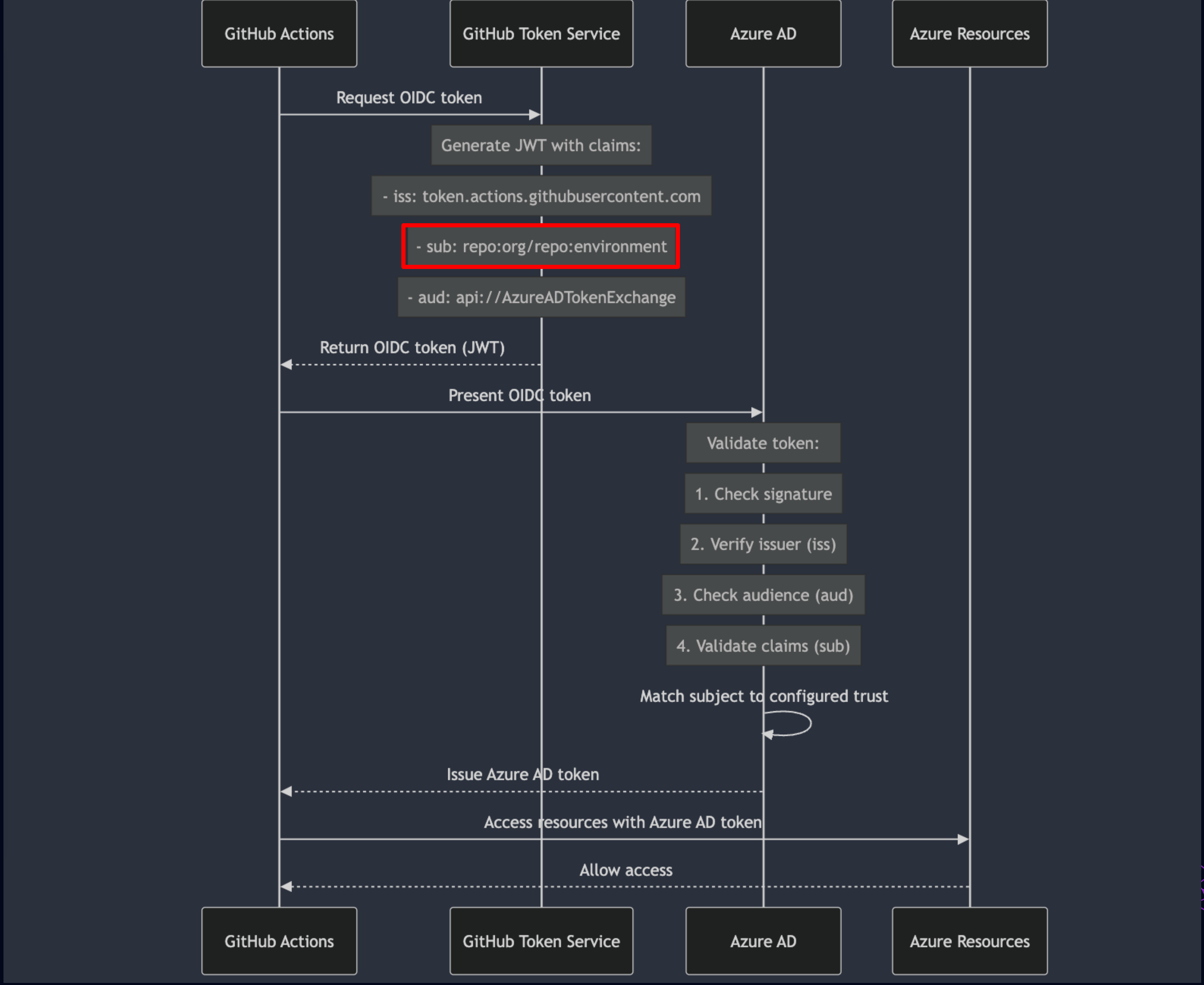
Solution? OIDC

- OpenID Connect
- Enables short-lived, auto-rotated creds
- Eliminates the need for a secret!
- Native support by all major cloud providers

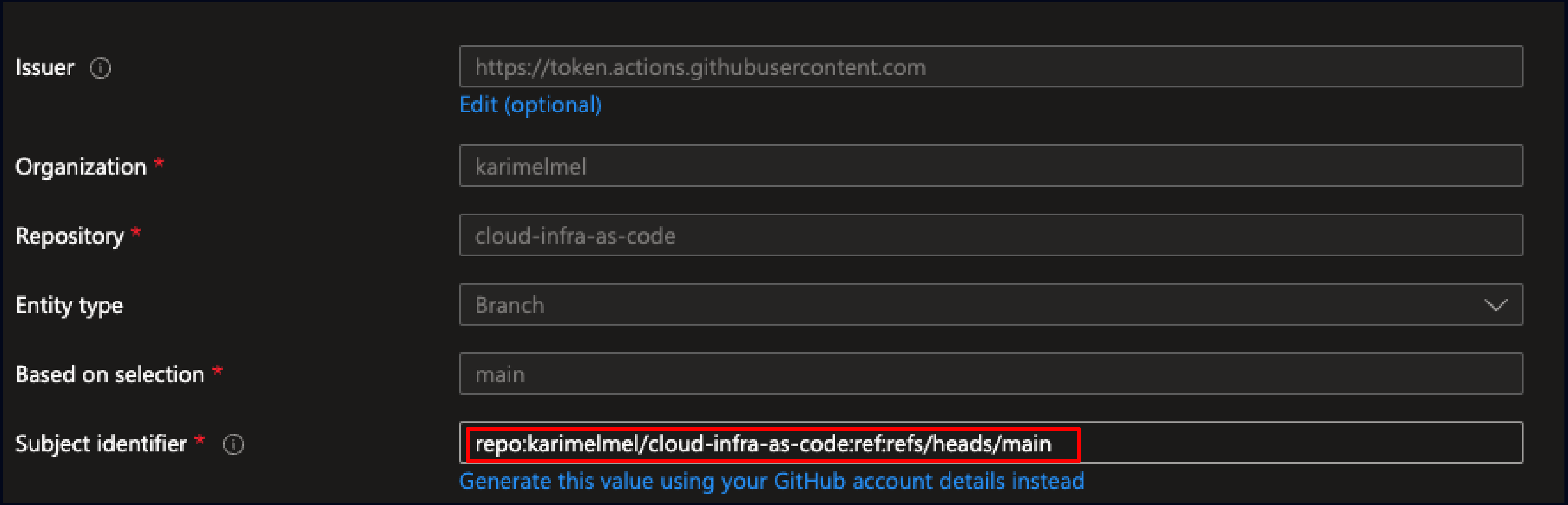
repo:org/repo:pull_request

repo:org/repo:<tag>

repo:org/repo:<branch>



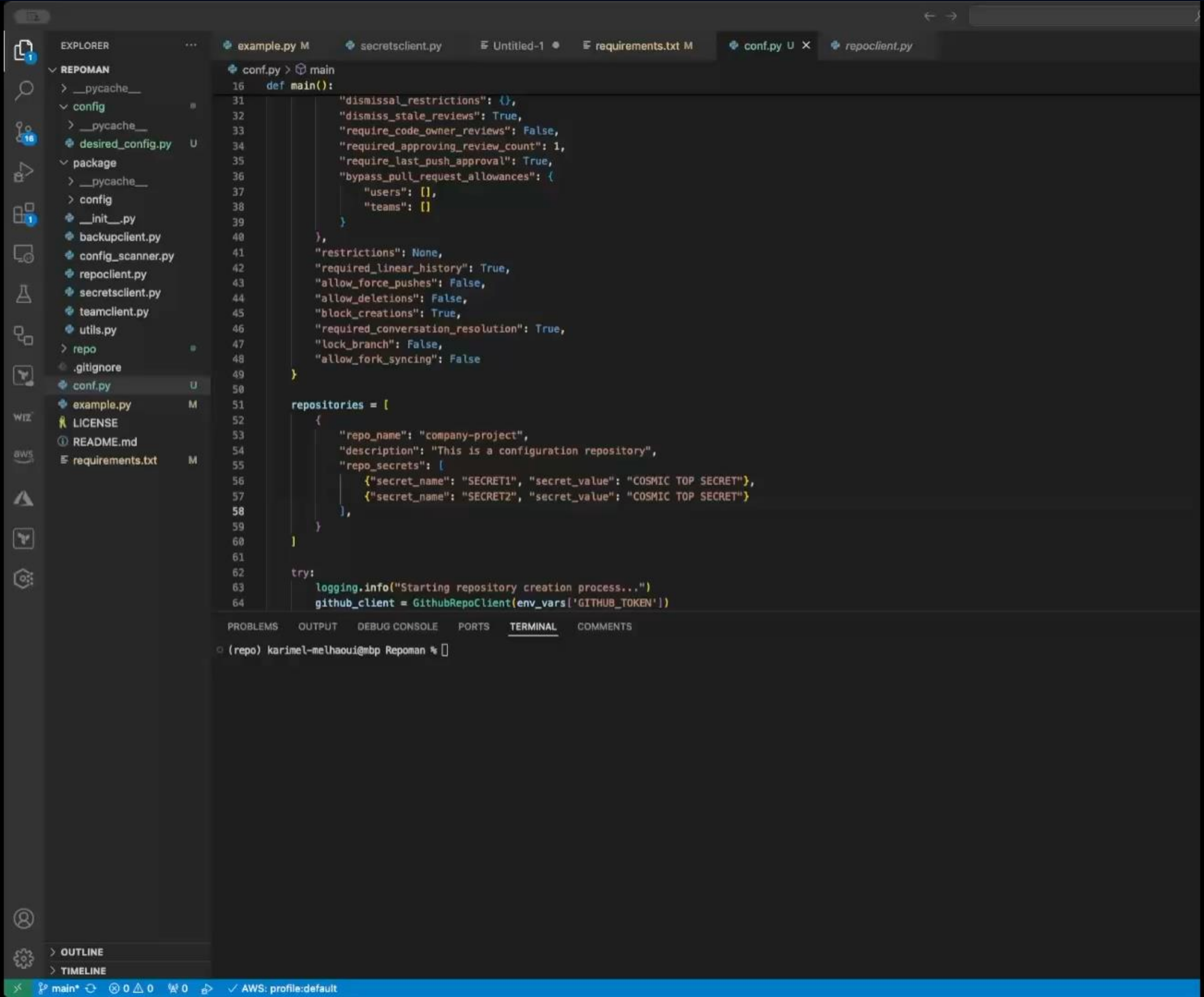
OIDC Flow



Issuer ⓘ	https://token.actions.githubusercontent.com
	Edit (optional)
Organization *	karimelmel
Repository *	cloud-infra-as-code
Entity type	Branch
Based on selection *	main
Subject identifier * ⓘ	repo:karimelmel/cloud-infra-as-code:ref:refs/heads/main
	Generate this value using your GitHub account details instead

What other measures? Environments

- Isolated context for deployment
- Allows granular controls
- Combines with branch protection



```python
def main():
            "dismissal_restrictions": {},
            "dismiss_stale_reviews": True,
            "require_code_owner_reviews": False,
            "required_approving_review_count": 1,
            "require_last_push_approval": True,
            "bypass_pull_request_allowances": {
                "users": [],
                "teams": []
            }
        },
        "restrictions": None,
        "required_linear_history": True,
        "allow_force_pushes": False,
        "allow_deletions": False,
        "block_creations": True,
        "required_conversation_resolution": True,
        "lock_branch": False,
        "allow_fork_syncing": False
    }

    repositories = [
        {
            "repo_name": "company-project",
            "description": "This is a configuration repository",
            "repo_secrets": [
                {"secret_name": "SECRET1", "secret_value": "COSMIC TOP SECRET"},
                {"secret_name": "SECRET2", "secret_value": "COSMIC TOP SECRET"}
            ],
        }
    ]

    try:
        logging.info("Starting repository creation process...")
        github_client = GithubRepoClient(env_vars['GITHUB_TOKEN'])
```

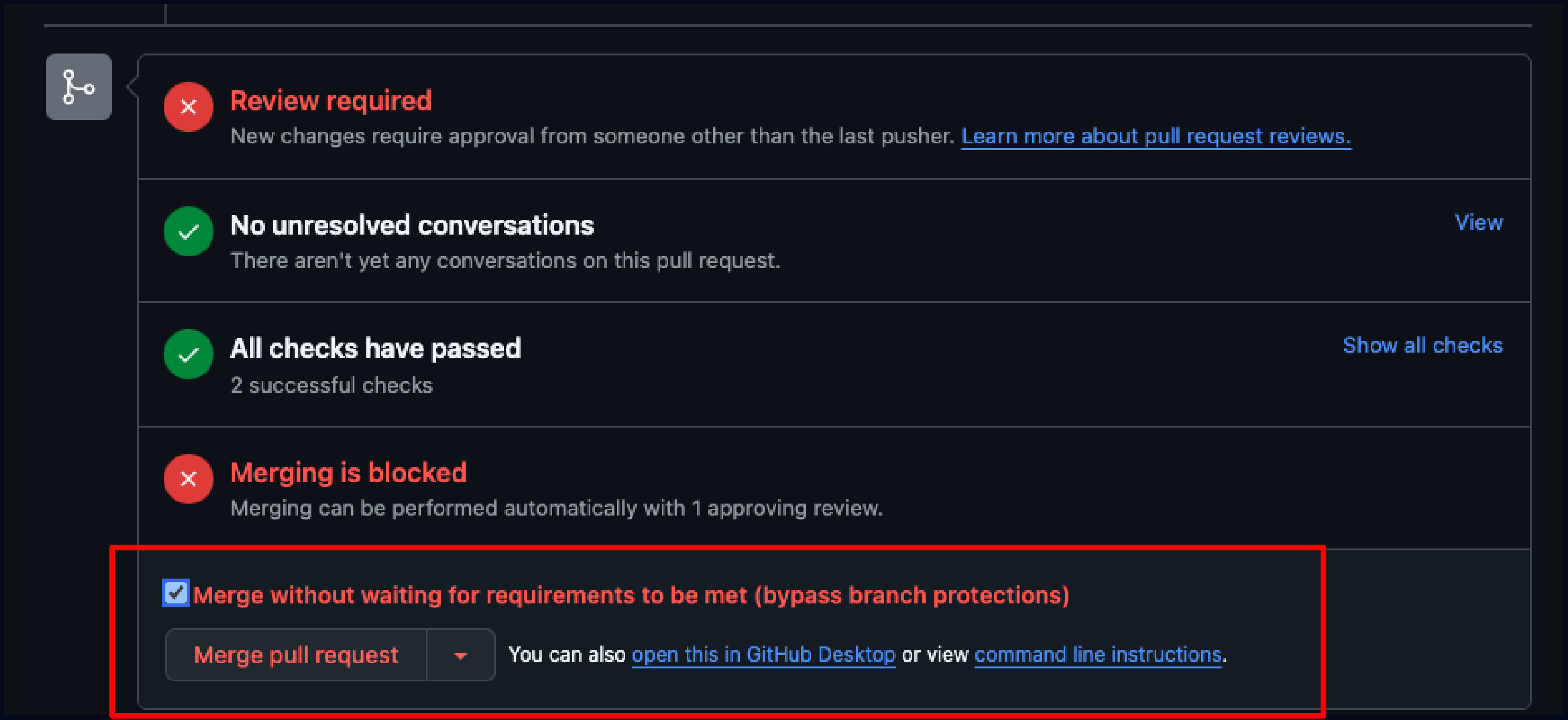
```
(repo) karimel-melhaoui@mbp Repoman % 
```

How do we effectively protect our CI secrets?

- Use Environments
- Protect workflow file
- Use OIDC with **secure configuration**
- Use Protected Branches
  - Require approvals
  - Dismiss when new commits are pushed
  - Require approval of recent reviewable push
  - Disallow Force pushes

OIDC for short-lived token

Branch Protection for integrity



Review required

New changes require approval from someone other than the last pusher. Learn more about pull request reviews.

No unresolved conversations View

There aren't yet any conversations on this pull request.

All checks have passed Show all checks

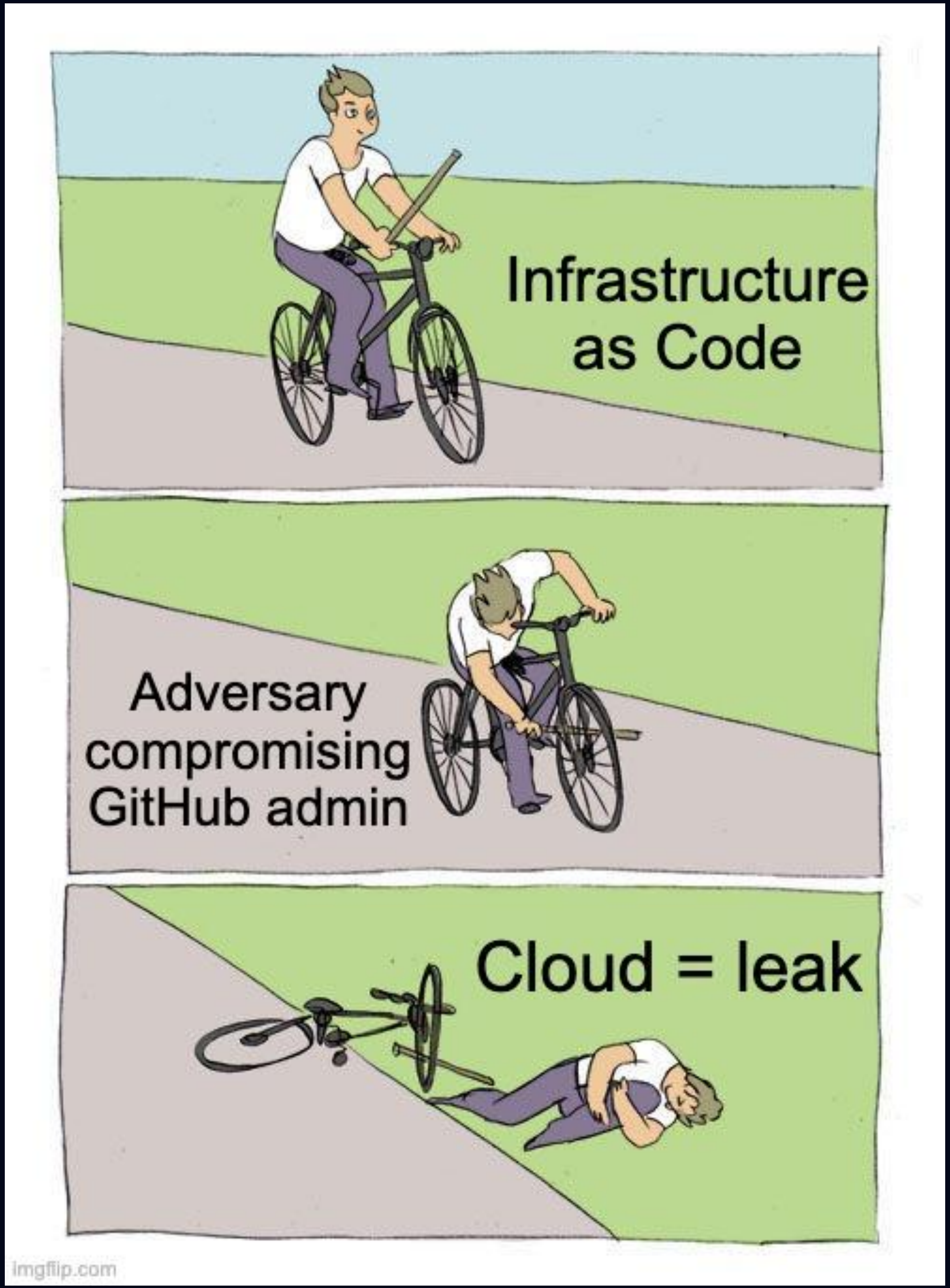
2 successful checks

Merging is blocked

Merging can be performed automatically with 1 approving review.

[✓] Merge without waiting for requirements to be met (bypass branch protections)

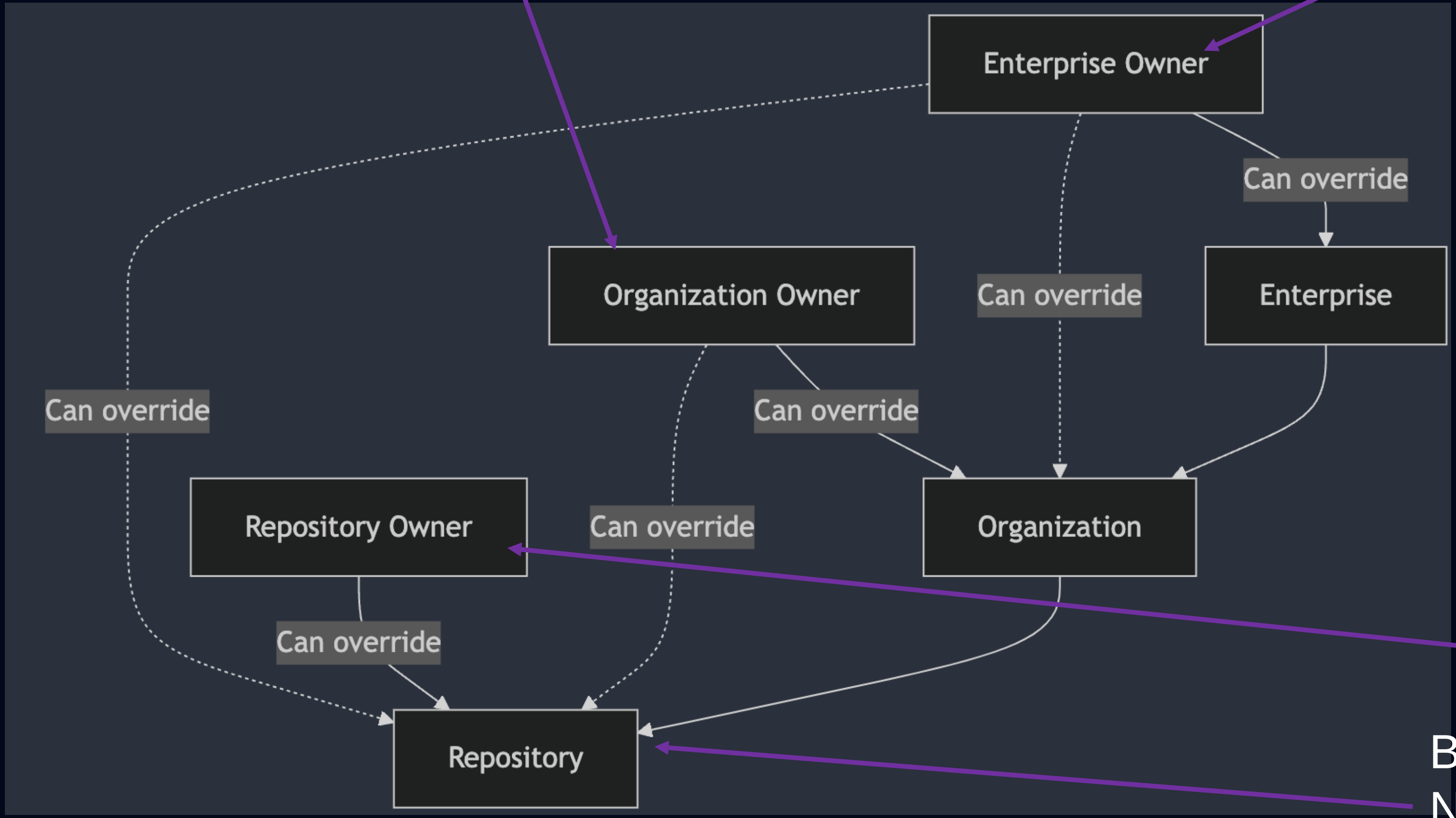
You can also open this in GitHub Desktop or view command line instructions.


Infrastructure as Code
Adversary compromising GitHub admin
Cloud = leak
imgflip.com

Ideal state

1-2 dedicated admin accounts
Regular backups

2 dedicated admin accounts



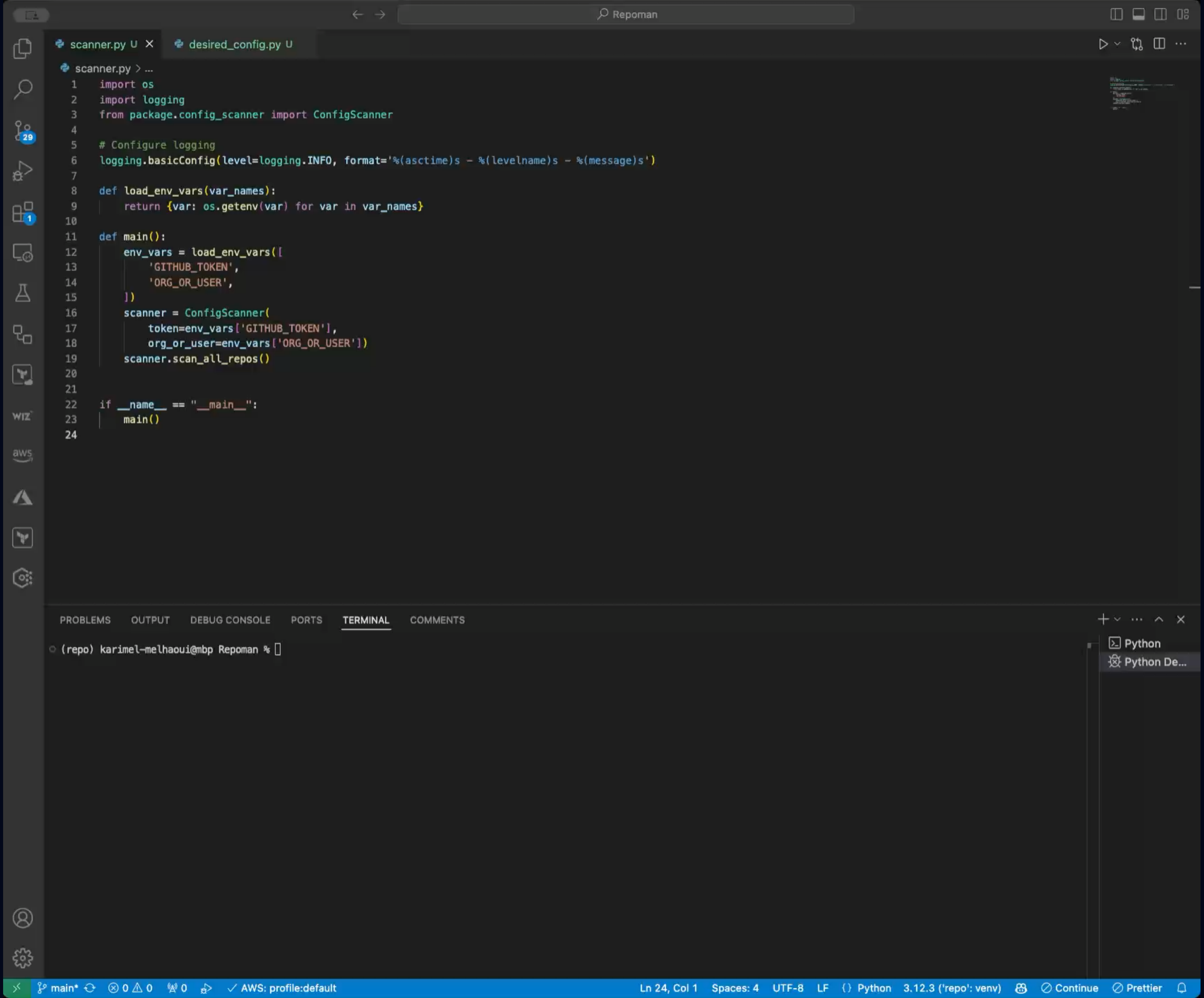
Regular backups

Nobody

Branch protection with peer review
No misconfigured OIDC
No secrets

What more do we care about?

Unprotected repos



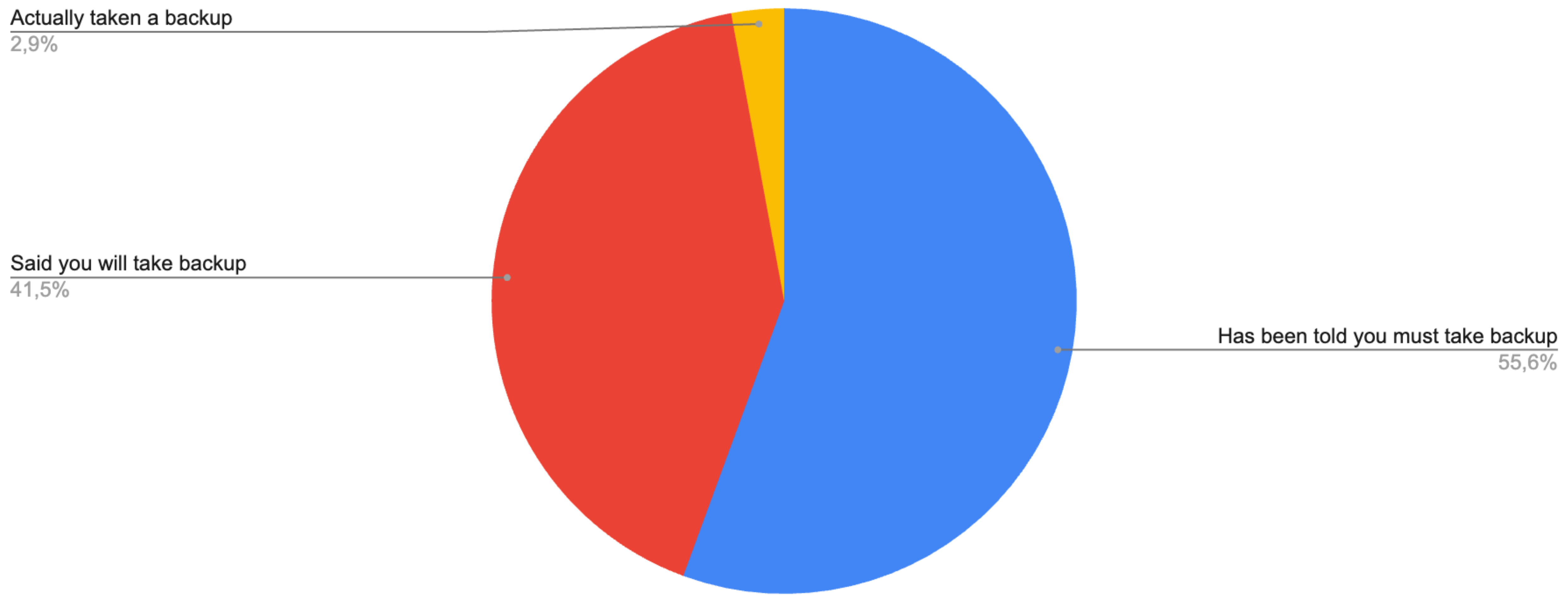
```python
import os
import logging
from package.config_scanner import ConfigScanner

# Configure logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

def load_env_vars(var_names):
    return {var: os.getenv(var) for var in var_names}

def main():
    env_vars = load_env_vars([
        'GITHUB_TOKEN',
        'ORG_OR_USER',
    ])
    scanner = ConfigScanner(
        token=env_vars['GITHUB_TOKEN'],
        org_or_user=env_vars['ORG_OR_USER'])
    scanner.scan_all_repos()


if __name__ == "__main__":
    main()

```

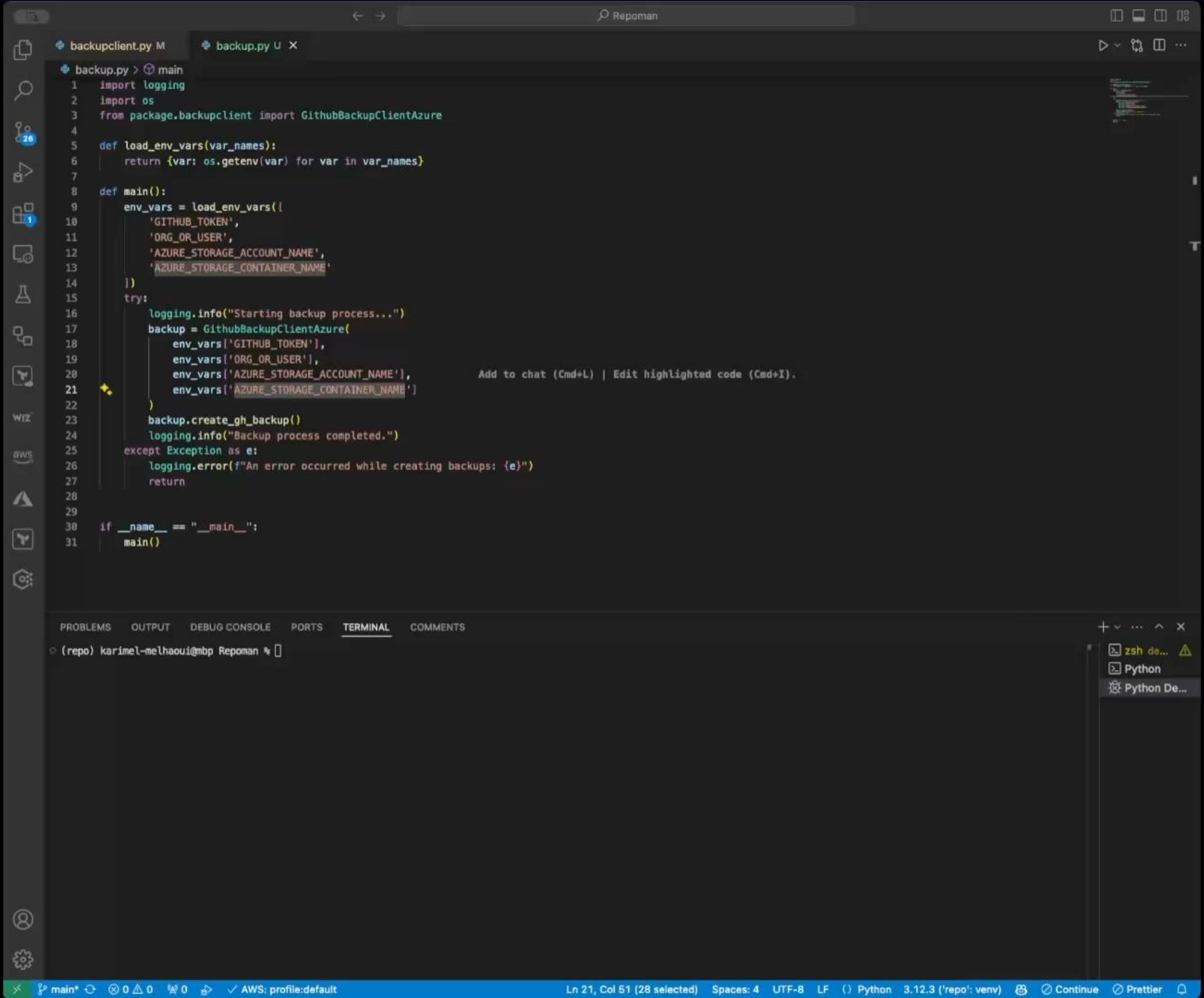
```
(repo) karimel-melhaoui@mbp Repoman %
```

Backing up your IaC

”We do IaC so we can recover fast”

Backup


Actually taken a backup
2,9%
Said you will take backup
41,5%
Has been told you must take backup
55,6%

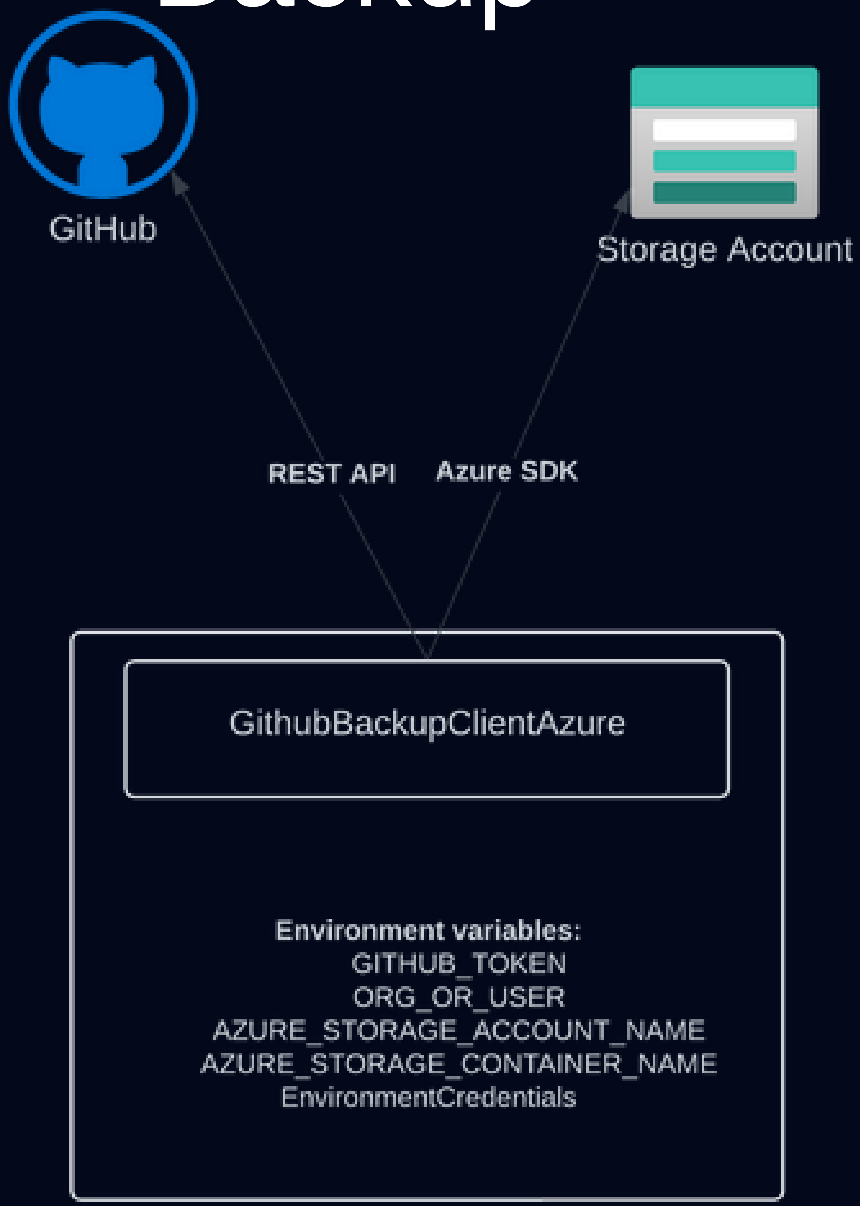


```python
import logging
import os
from package.backupclient import GithubBackupClientAzure

def load_env_vars(var_names):
    return {var: os.getenv(var) for var in var_names}

def main():
    env_vars = load_env_vars([
        'GITHUB_TOKEN',
        'ORG_OR_USER',
        'AZURE_STORAGE_ACCOUNT_NAME',
        'AZURE_STORAGE_CONTAINER_NAME'
    ])
    try:
        logging.info("Starting backup process...")
        backup = GithubBackupClientAzure(
            env_vars['GITHUB_TOKEN'],
            env_vars['ORG_OR_USER'],
            env_vars['AZURE_STORAGE_ACCOUNT_NAME'],
            env_vars['AZURE_STORAGE_CONTAINER_NAME']
        )
        backup.create_gh_backup()
        logging.info("Backup process completed.")
    except Exception as e:
        logging.error(f"An error occurred while creating backups: {e}")
        return


if __name__ == "__main__":
    main()
```

Add to chat (Cmd+L) | Edit highlighted code (Cmd+I).

```
(repo) karimel-melhaoui@mbp Repoman %
```

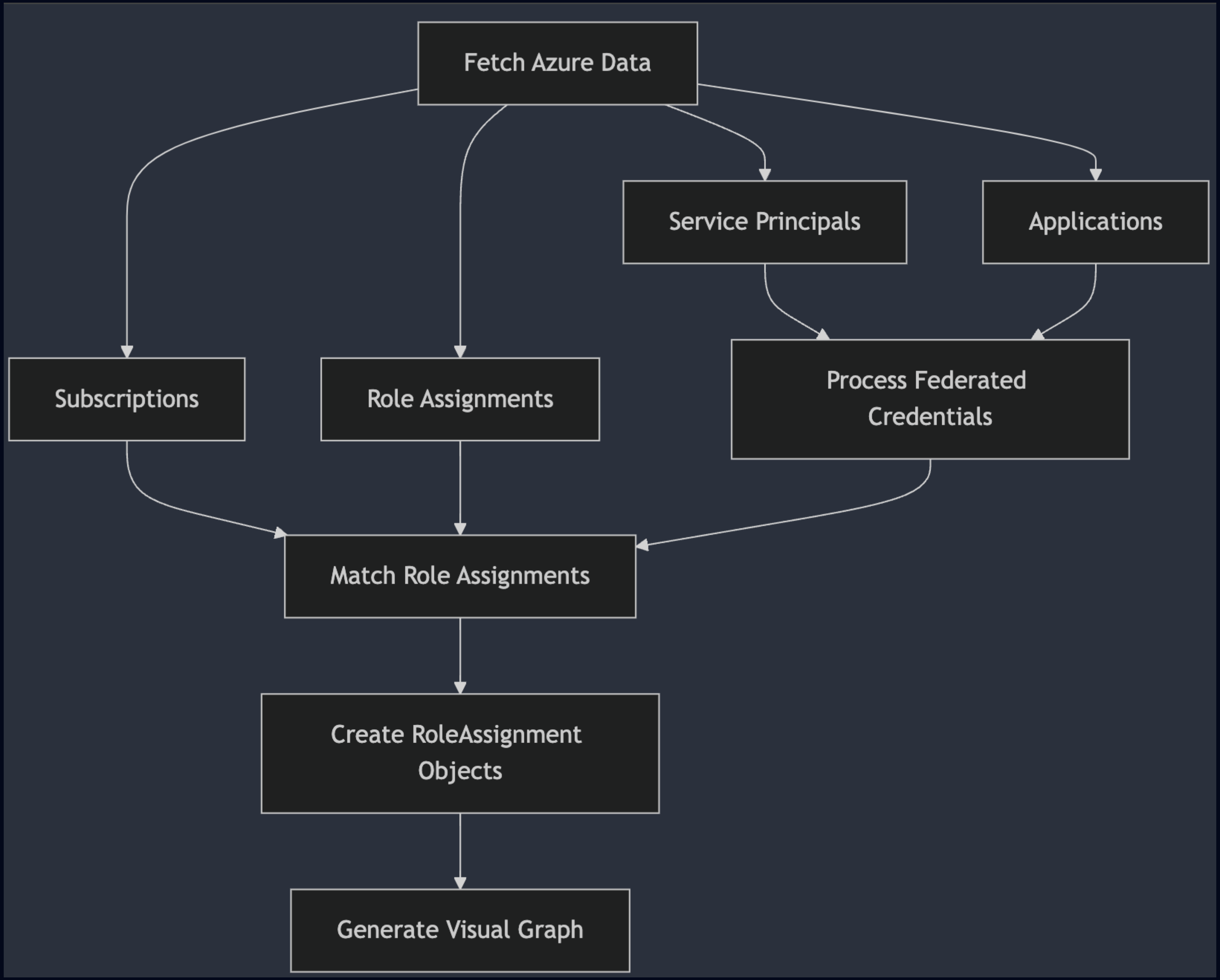
Backup

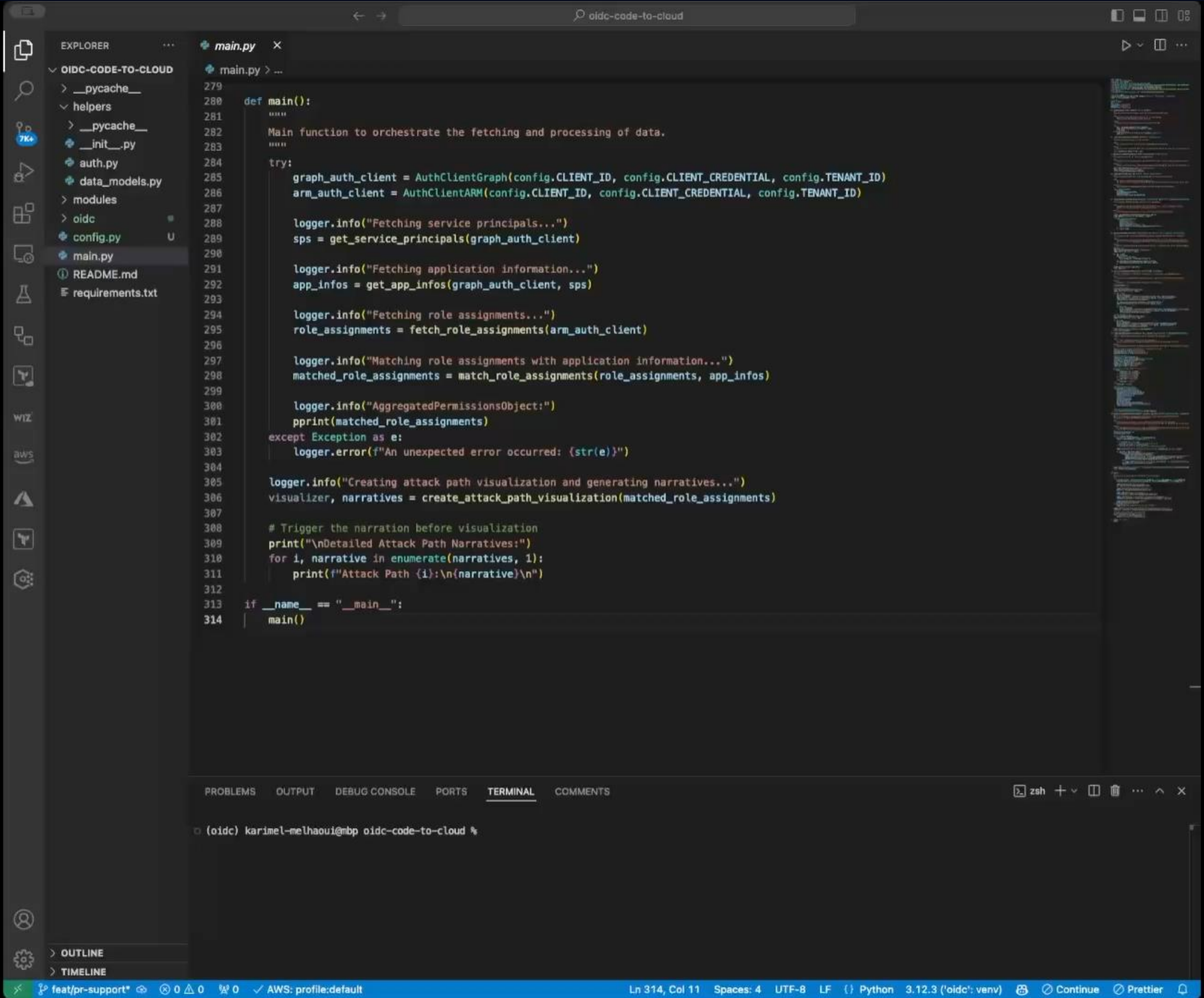

GitHub
Storage Account
REST API
Azure SDK
GithubBackupClientAzure
Environment variables:
GITHUB_TOKEN
ORG_OR_USER
AZURE_STORAGE_ACCOUNT_NAME
AZURE_STORAGE_CONTAINER_NAME
EnvironmentCredentials

bonus… ?

Finding Misconfigured OIDC

Early research preview*

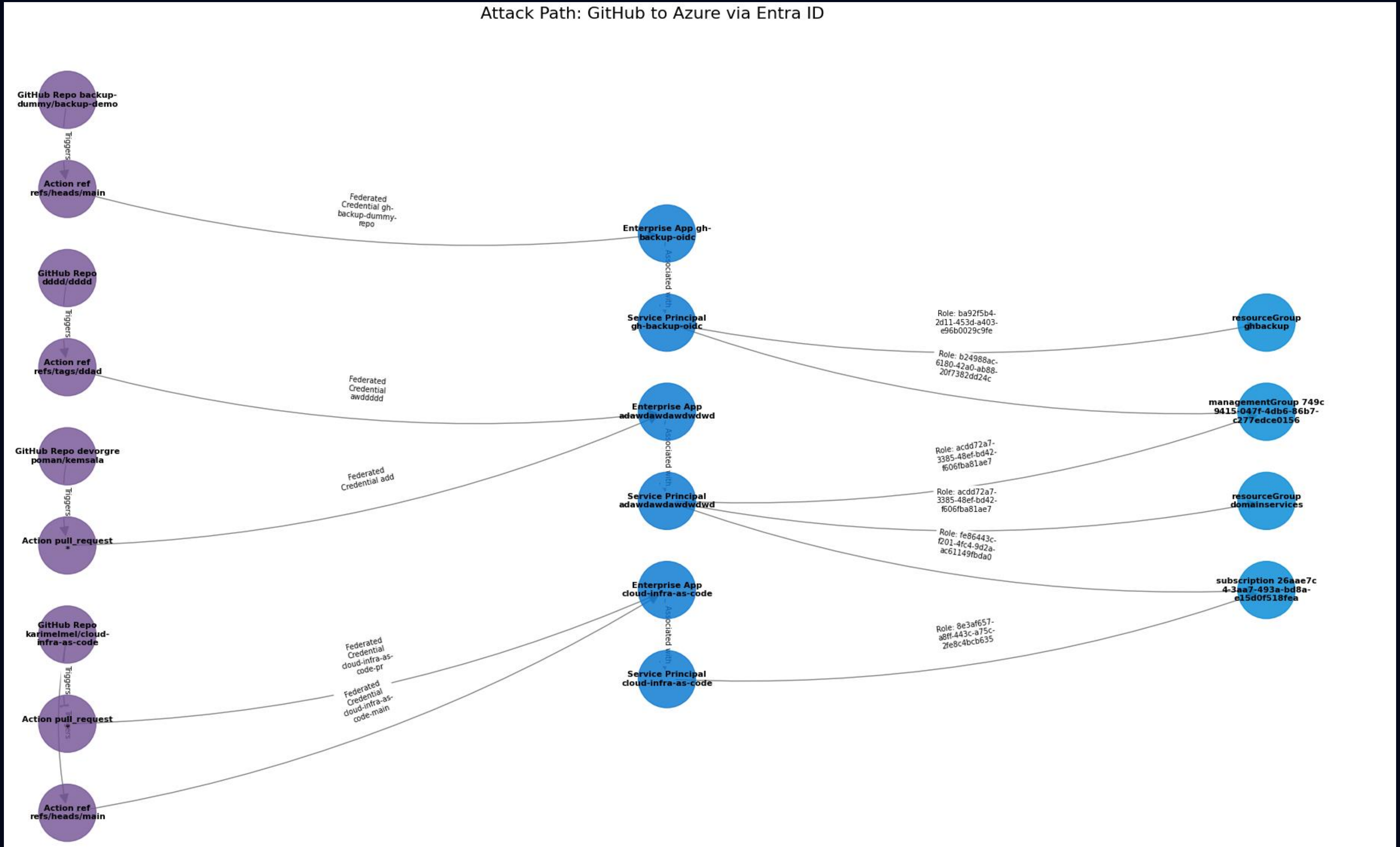

Fetch Azure Data
Service Principals
Applications
Subscriptions
Role Assignments
Process Federated Credentials
Match Role Assignments
Create RoleAssignment Objects
Generate Visual Graph



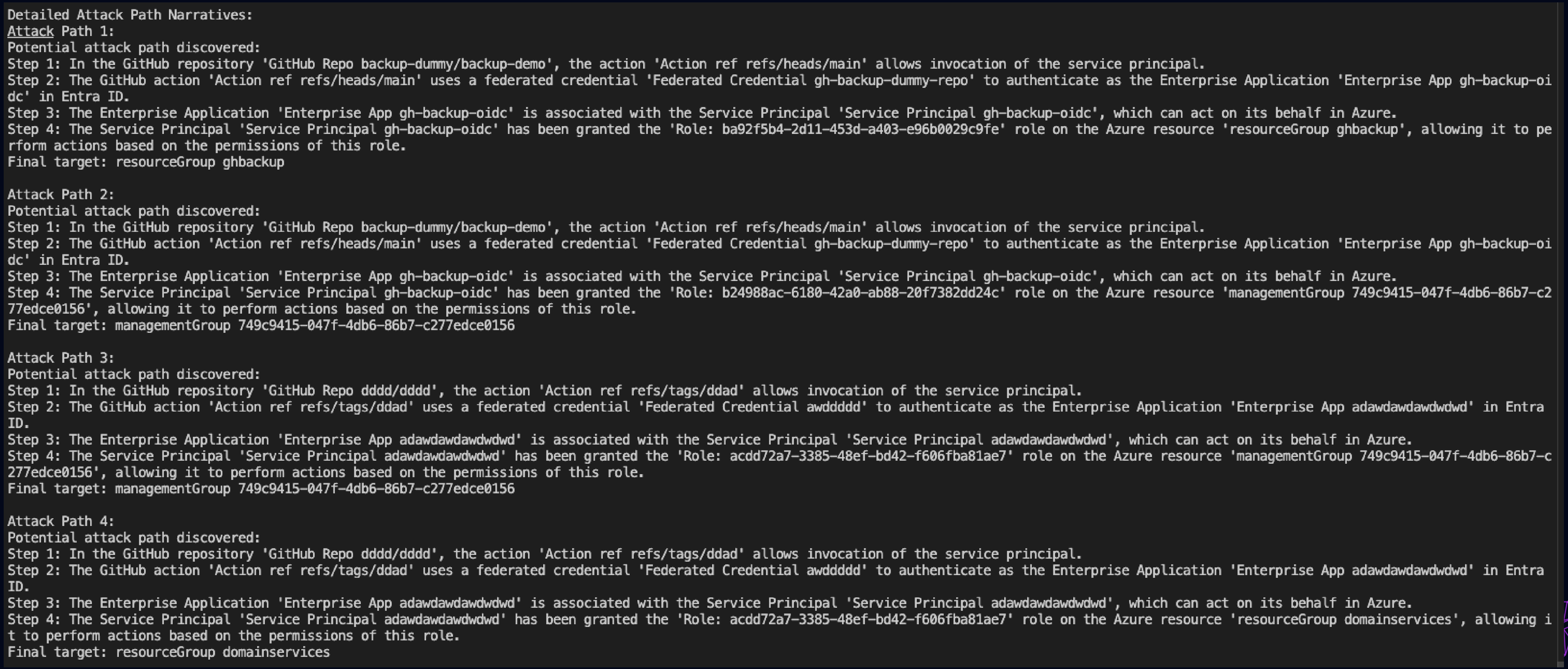
```python

def main():
    """
    Main function to orchestrate the fetching and processing of data.
    """
    try:
        graph_auth_client = AuthClientGraph(config.CLIENT_ID, config.CLIENT_CREDENTIAL, config.TENANT_ID)
        arm_auth_client = AuthClientARM(config.CLIENT_ID, config.CLIENT_CREDENTIAL, config.TENANT_ID)

        logger.info("Fetching service principals...")
        sps = get_service_principals(graph_auth_client)

        logger.info("Fetching application information...")
        app_infos = get_app_infos(graph_auth_client, sps)

        logger.info("Fetching role assignments...")
        role_assignments = fetch_role_assignments(arm_auth_client)

        logger.info("Matching role assignments with application information...")
        matched_role_assignments = match_role_assignments(role_assignments, app_infos)

        logger.info("AggregatedPermissionsObject:")
        pprint(matched_role_assignments)
    except Exception as e:
        logger.error(f"An unexpected error occurred: {str(e)}")

    logger.info("Creating attack path visualization and generating narratives...")
    visualizer, narratives = create_attack_path_visualization(matched_role_assignments)

    # Trigger the narration before visualization
    print("\nDetailed Attack Path Narratives:")
    for i, narrative in enumerate(narratives, 1):
        print(f"Attack Path {i}:\n{narrative}\n")

if __name__ == "__main__":
    main()
```

```
(oidc) karimel-melhaoui@mbp oidc-code-to-cloud %
```

Attack Path Visualizer


Attack Path: GitHub to Azure via Entra ID
GitHub Repo backup-dummy/backup-demo
Triggers
Action ref refs/heads/main
Federated Credential gh-backup-dummy-repo
Enterprise App gh-backup-oidc
Associated with
Service Principal gh-backup-oidc
Role: ba92f5b4-2d11-453d-a403-e96b0029c9fe
Role: b24988ac-6180-42a0-ab88-20f7382dd24c
resourceGroup ghbackup
managementGroup 749c 9415-047f-4db6-86b7-c277edce0156
GitHub Repo dddd/dddd
Triggers
Action ref refs/tags/ddad
Federated Credential awddddd
Enterprise App adawdawdawdwdwd
Associated with
Service Principal adawdawdawdwdwd
Role: acdd72a7-3385-48ef-bd42-f606fba81ae7
Role: acdd72a7-3385-48ef-bd42-f606fba81ae7
Role: fe86443c-f201-4fc4-9d2a-ac61149fbda0
resourceGroup domainservices
subscription 26aae7c 4-3aa7-493a-bd8a-e15d0f518fea
GitHub Repo devorgre poman/kemsala
Triggers
Action pull_request *
Federated Credential add
Enterprise App cloud-infra-as-code
Associated with
Service Principal cloud-infra-as-code
Role: 8e3af657-a8ff-443c-a75c-2fe8c4bcb635
GitHub Repo karimelmel/cloud-infra-as-code
Triggers
Action pull_request *
Triggers
Federated Credential cloud-infra-as-code-pr
Federated Credential cloud-infra-as-code-main
Action ref refs/heads/main

Attack Path Visualizer

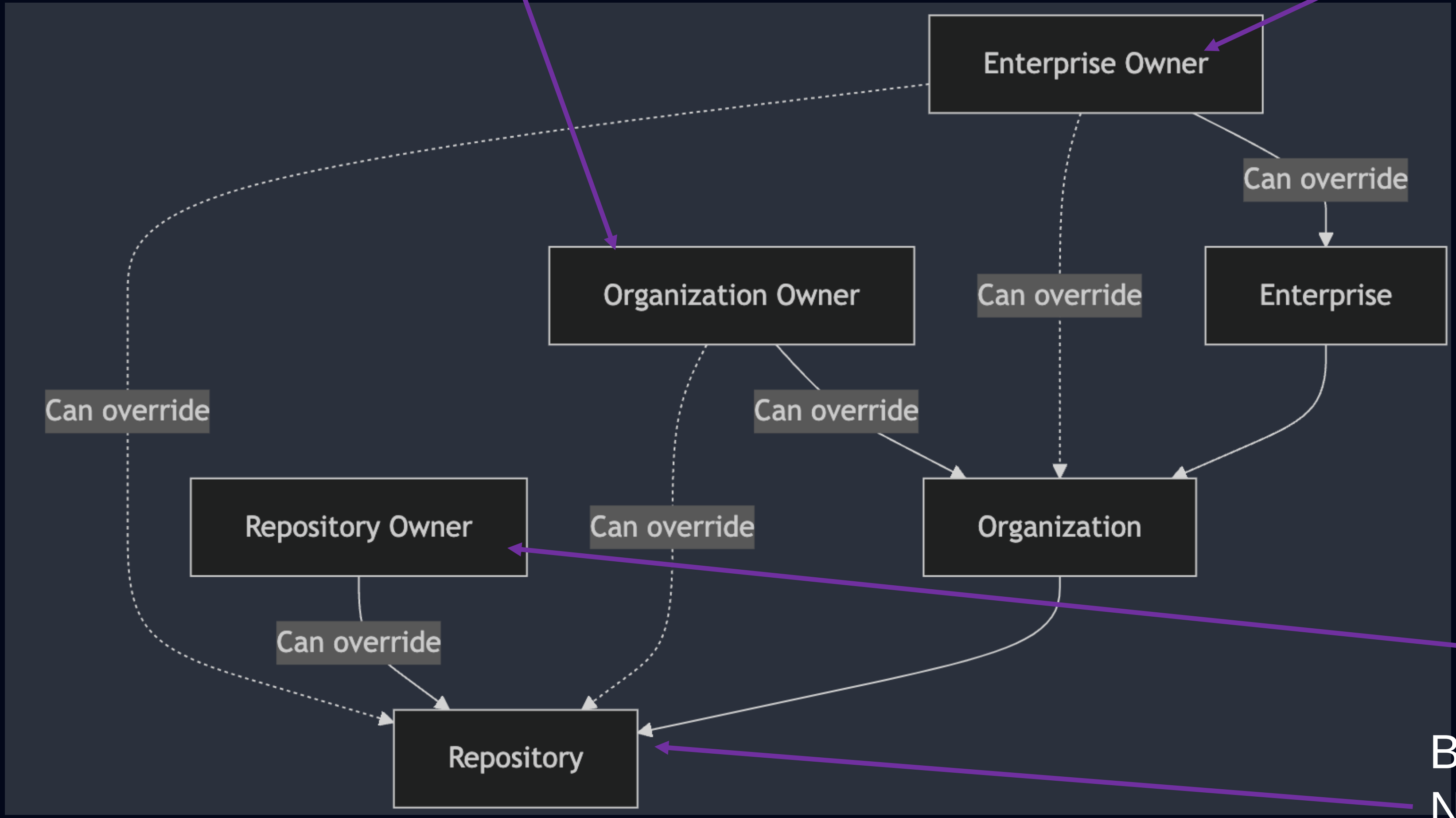


```
Detailed Attack Path Narratives:
Attack Path 1:
Potential attack path discovered:
Step 1: In the GitHub repository 'GitHub Repo backup-dummy/backup-demo', the action 'Action ref refs/heads/main' allows invocation of the service principal.
Step 2: The GitHub action 'Action ref refs/heads/main' uses a federated credential 'Federated Credential gh-backup-dummy-repo' to authenticate as the Enterprise Application 'Enterprise App gh-backup-oi
dc' in Entra ID.
Step 3: The Enterprise Application 'Enterprise App gh-backup-oidc' is associated with the Service Principal 'Service Principal gh-backup-oidc', which can act on its behalf in Azure.
Step 4: The Service Principal 'Service Principal gh-backup-oidc' has been granted the 'Role: ba92f5b4-2d11-453d-a403-e96b0029c9fe' role on the Azure resource 'resourceGroup ghbackup', allowing it to pe
rform actions based on the permissions of this role.
Final target: resourceGroup ghbackup

Attack Path 2:
Potential attack path discovered:
Step 1: In the GitHub repository 'GitHub Repo backup-dummy/backup-demo', the action 'Action ref refs/heads/main' allows invocation of the service principal.
Step 2: The GitHub action 'Action ref refs/heads/main' uses a federated credential 'Federated Credential gh-backup-dummy-repo' to authenticate as the Enterprise Application 'Enterprise App gh-backup-oi
dc' in Entra ID.
Step 3: The Enterprise Application 'Enterprise App gh-backup-oidc' is associated with the Service Principal 'Service Principal gh-backup-oidc', which can act on its behalf in Azure.
Step 4: The Service Principal 'Service Principal gh-backup-oidc' has been granted the 'Role: b24988ac-6180-42a0-ab88-20f7382dd24c' role on the Azure resource 'managementGroup 749c9415-047f-4db6-86b7-c2
77edce0156', allowing it to perform actions based on the permissions of this role.
Final target: managementGroup 749c9415-047f-4db6-86b7-c277edce0156

Attack Path 3:
Potential attack path discovered:
Step 1: In the GitHub repository 'GitHub Repo dddd/dddd', the action 'Action ref refs/tags/ddad' allows invocation of the service principal.
Step 2: The GitHub action 'Action ref refs/tags/ddad' uses a federated credential 'Federated Credential awddddd' to authenticate as the Enterprise Application 'Enterprise App adawdawdawdwdwd' in Entra
ID.
Step 3: The Enterprise Application 'Enterprise App adawdawdawdwdwd' is associated with the Service Principal 'Service Principal adawdawdawdwdwd', which can act on its behalf in Azure.
Step 4: The Service Principal 'Service Principal adawdawdawdwdwd' has been granted the 'Role: acdd72a7-3385-48ef-bd42-f606fba81ae7' role on the Azure resource 'managementGroup 749c9415-047f-4db6-86b7-c
277edce0156', allowing it to perform actions based on the permissions of this role.
Final target: managementGroup 749c9415-047f-4db6-86b7-c277edce0156

Attack Path 4:
Potential attack path discovered:
Step 1: In the GitHub repository 'GitHub Repo dddd/dddd', the action 'Action ref refs/tags/ddad' allows invocation of the service principal.
Step 2: The GitHub action 'Action ref refs/tags/ddad' uses a federated credential 'Federated Credential awddddd' to authenticate as the Enterprise Application 'Enterprise App adawdawdawdwdwd' in Entra
ID.
Step 3: The Enterprise Application 'Enterprise App adawdawdawdwdwd' is associated with the Service Principal 'Service Principal adawdawdawdwdwd', which can act on its behalf in Azure.
Step 4: The Service Principal 'Service Principal adawdawdawdwdwd' has been granted the 'Role: acdd72a7-3385-48ef-bd42-f606fba81ae7' role on the Azure resource 'resourceGroup domainservices', allowing i
t to perform actions based on the permissions of this role.
Final target: resourceGroup domainservices
```

Ideal state

1-2 dedicated admin accounts
Regular backups

2 dedicated admin accounts



Regular backups

Nobody

Branch protection with peer review
No misconfigured OIDC
No secrets

Learn more?

o3c.no

karim@o3c.no