I Need a C2 Infrastructure Immediately… As in, Yesterday!

NIKOS VOURDAS

KYPRIANOS VASILOPOULOS

2025

$WHOAMI

- Senior Offensive Security Consultant based in the USA
- Adversary Simulation/Emulation
- Tiber-EU/iCAST Experienced
- OSCE3, OSCP, OSWP, CRTL, CRTO && OASP Certified
- Author: Supernova, COM-Hunter && SkyFall-Pack
- @nickvourd

$WHOAMI

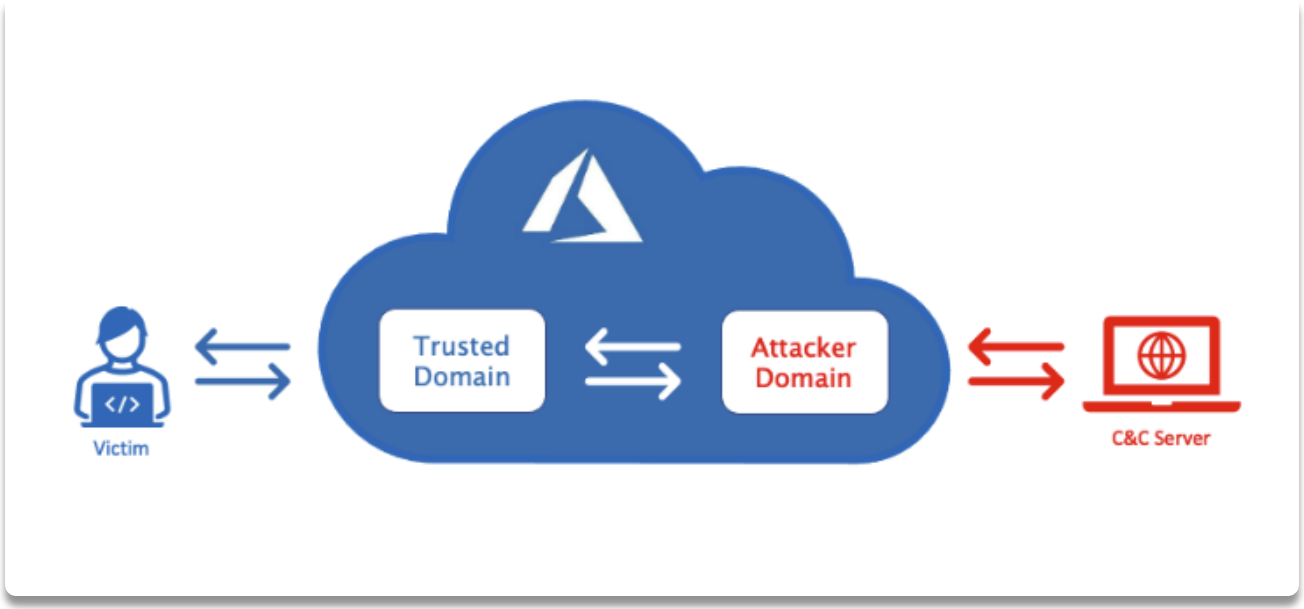
- Chief Information Security Officer (CISO) based in Greece
- Co-Founder of @TheOffensiveX
- Adversary Simulation/Emulation
- OSCP, OSCE, PACES && CRTO
- Zyxel Networks CVE Hall of Fame
- @kavasilo

Table of Contents

- RIP Domain Fronting
- C2 Infrastructure Standards
- Cloudflare Workers
- The Rise of the Redirector
- Team Server Configurations
- Kyle Saved My Team Server
- SkyFall-Pack Automation
- Detections
- References
- Questions

RIP Domain Fronting

- Domain fronting is a networking technique that allows a backend server to hide its identity by using the secure connection of a trusted front-end domain.



RIP Domain Fronting

But One Day Unfortunately ☹

> Azure CDN from Edgio will be retired on January 15, 2025. You must migrate your workload to Azure Front Door before this date to avoid service disruption. This article provides guidance on how to migrate your workloads from Azure CDN from Edgio to Azure Front Door using Azure Traffic Manager. The migration process in this article can also be used to migrate workloads from a legacy CDN to Azure Front Door.

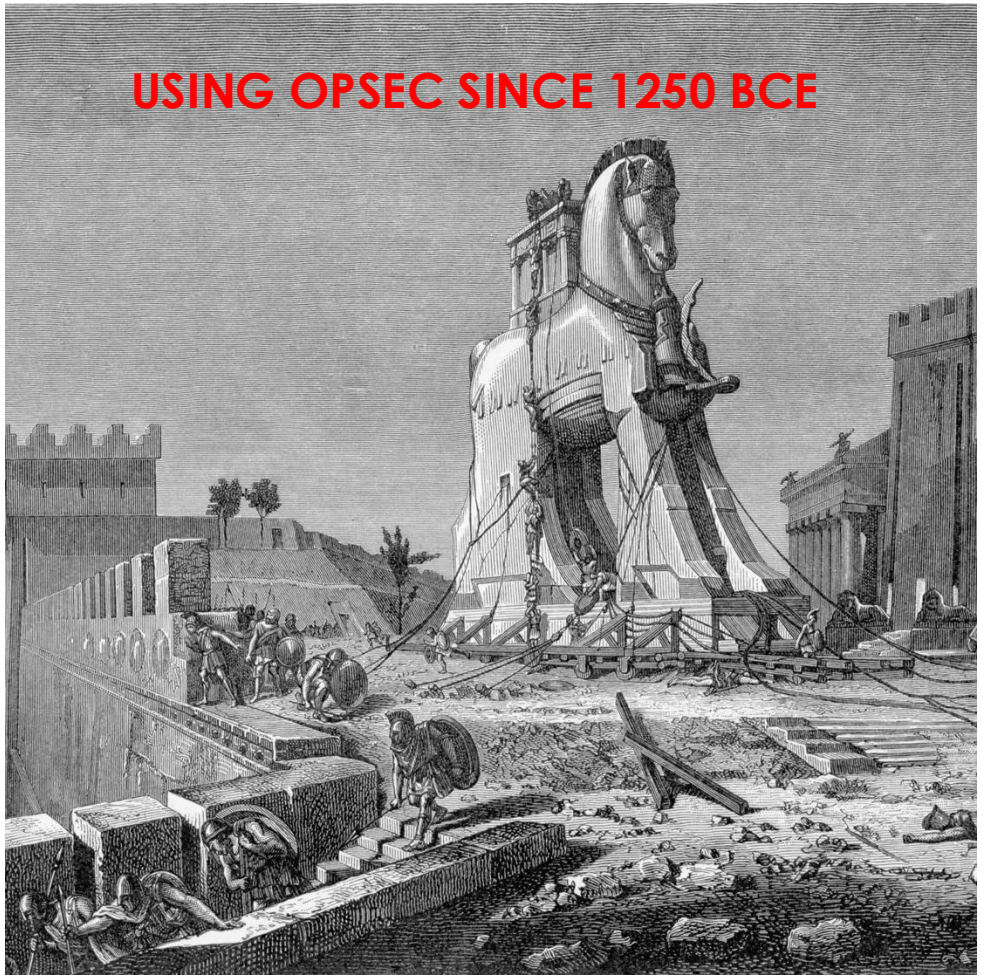
RIP Domain Fronting

C2 Infrastructure Standards

The 4 Standards of a C2 Infrastructure:

- Low-cost or affordable pricing
- OPSEC, targeted to the client's environment
- Protected Team Server
- Set up time

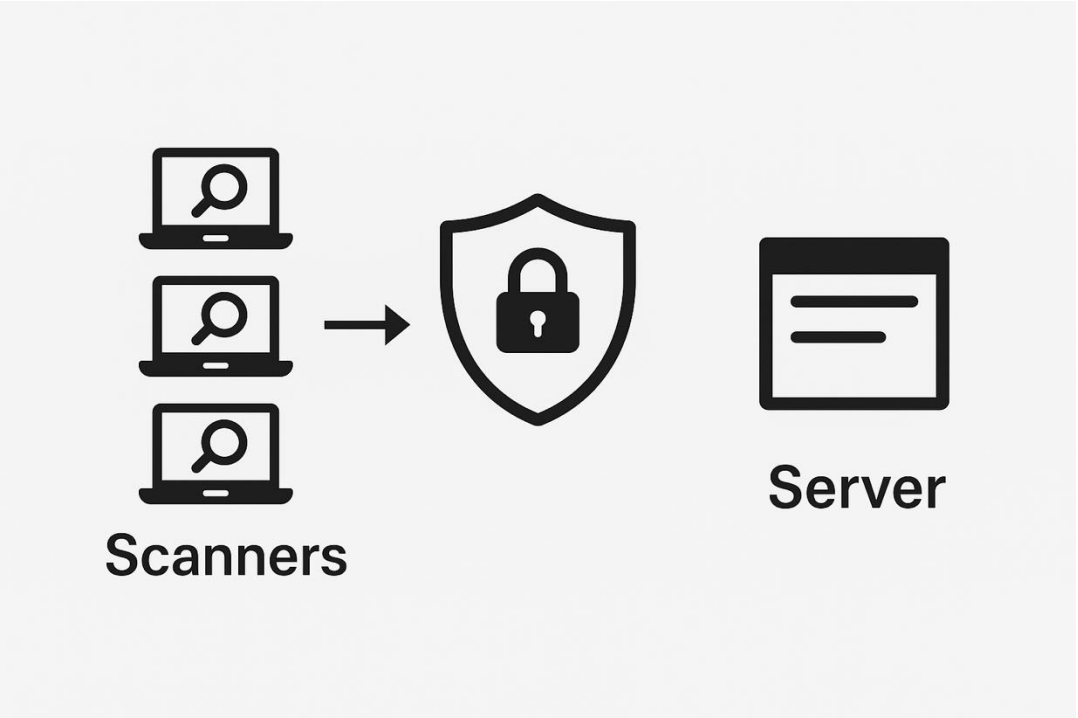
C2 Infrastructure Standards



C2 Infrastructure Standards

C2 Infrastructure Standards

C2 Infrastructure Standards



C2 Infrastructure Standards

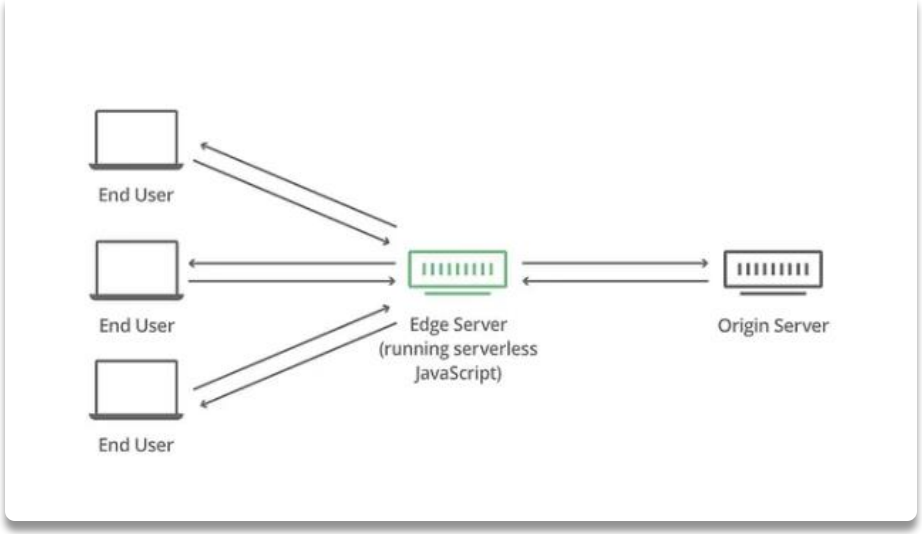
Time == Money

C2 Infrastructure Standards

Time == Money == Energy

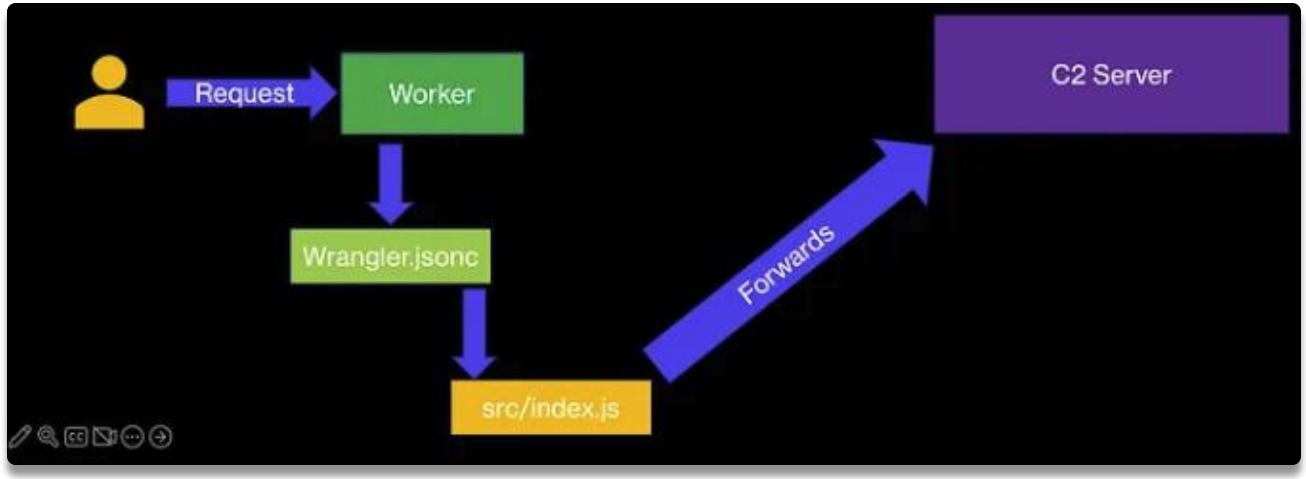
Cloudflare Workers

- A serverless computing platform provided by Cloudflare that allows developers to run JavaScript/TypeScript/Python code on their global network of data centers.
- Build and manage a Cloudflare Worker using the command-line tool named Wrangler or through Cloudflare's platform.



Cloudflare Workers

- wragler.jsonc
- Index.js

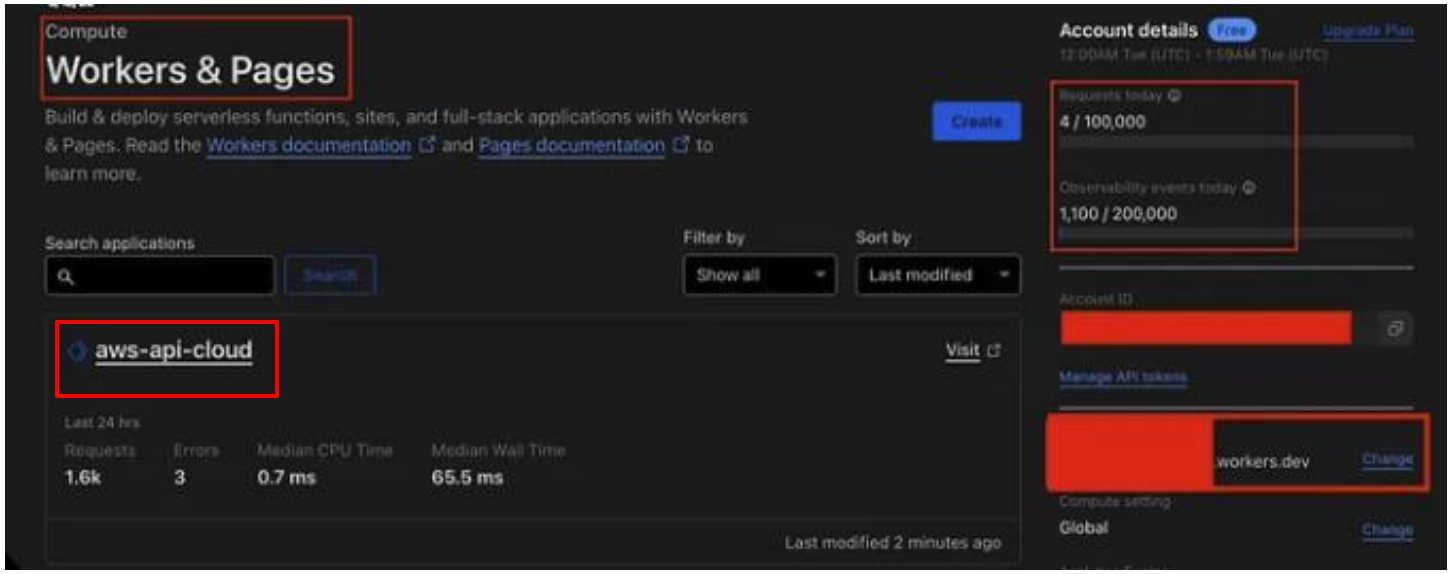


* Per day

Cloudflare Workers

Cloudflare Workers

```
https://<worker_project_name>.<subdomain>.workers.dev/
```



Cloudflare Workers

- wrangler.jsonc

```jsonc
{
  "$schema": "node_modules/wrangler/config-schema.json",
  "name": "aws-api-cloud",
  "main": "src/index.js",
  "compatibility_date": "2025-03-13",
  "workers_dev": true,
  "observability": {
    "enabled": true
  },
  "vars": {
    "TEAMSERVER": "https://nickvourd.eastus2.cloudapp.azure.com/",
    "REDIRECTOR": "https://aws-api-cloud.nickvourd.workers.dev/"
  }
}
```

Cloudflare Workers

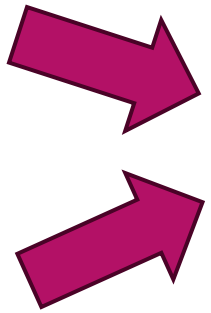
- index.js

```
// Listen for incoming fetch events
addEventListener('fetch', event => {
  // Respond with the result of the handleRequest function
  event.respondWith(handleRequest(event))
})

async function handleRequest(event) {
  // Get the incoming request object
  const incomingRequest = event.request

  // Clone the incoming request to preserve the body for forwarding
  const clonedIncomingRequest = incomingRequest.clone()

  // Extract the path from the incoming request URL by removing the REDIRECTOR
  const requestPath = incomingRequest.url.replace(REDIRECTOR, "")

  // Construct the target URL by appending the path to the TEAMSERVER base URL
  const targetUrl = TEAMSERVER + requestPath

  try {
    // Check if the incoming request is a POST request
    if (incomingRequest.method === 'POST') {
      // Forward the POST request to the target URL with the same headers and b
      const forwardResponse = await fetch(targetUrl, {
        method: 'POST',        headers: incomingRequest.headers,
        body: clonedIncomingRequest.body,
      })

      // Return the forwarded response
      return forwardResponse
    }
    // Check if the incoming request is a GET request
    else if (incomingRequest.method === 'GET') {
      // Forward the GET request to the target URL with the same headers
      const forwardResponse = await fetch(targetUrl, {
        method: 'GET',
        headers: incomingRequest.headers
      })

      // Return the forwarded response
      return forwardResponse
    }
    // Handle other HTTP methods that are not allowed (Forbidden)
    else {
      return new Response('Forbidden', { status: 403 })
    }
  } catch (error) {
    // If there is an error, return a 500 response with an error message
    return new Response('Error forwarding request', { status: 500 })
  }
}
```

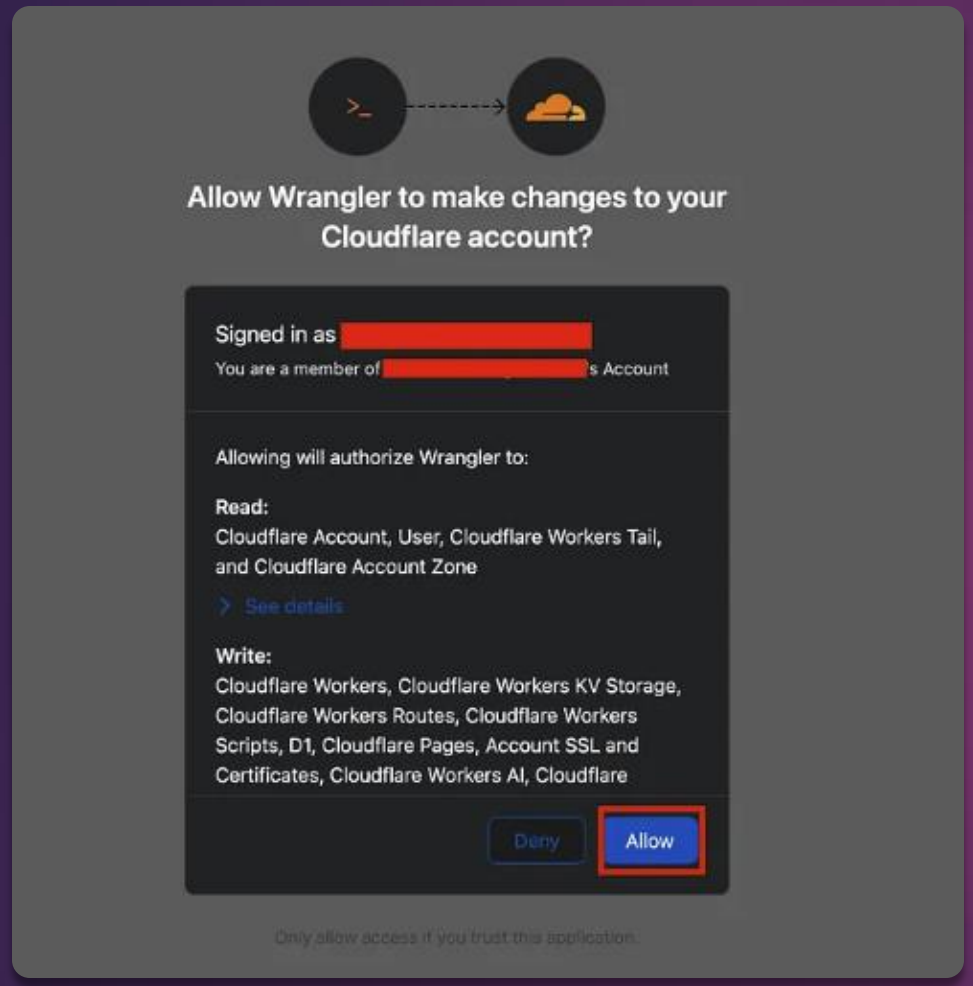
The Rise of the Redirector

```
sudo apt install npm -y
```

```
brew install node wrangler
```



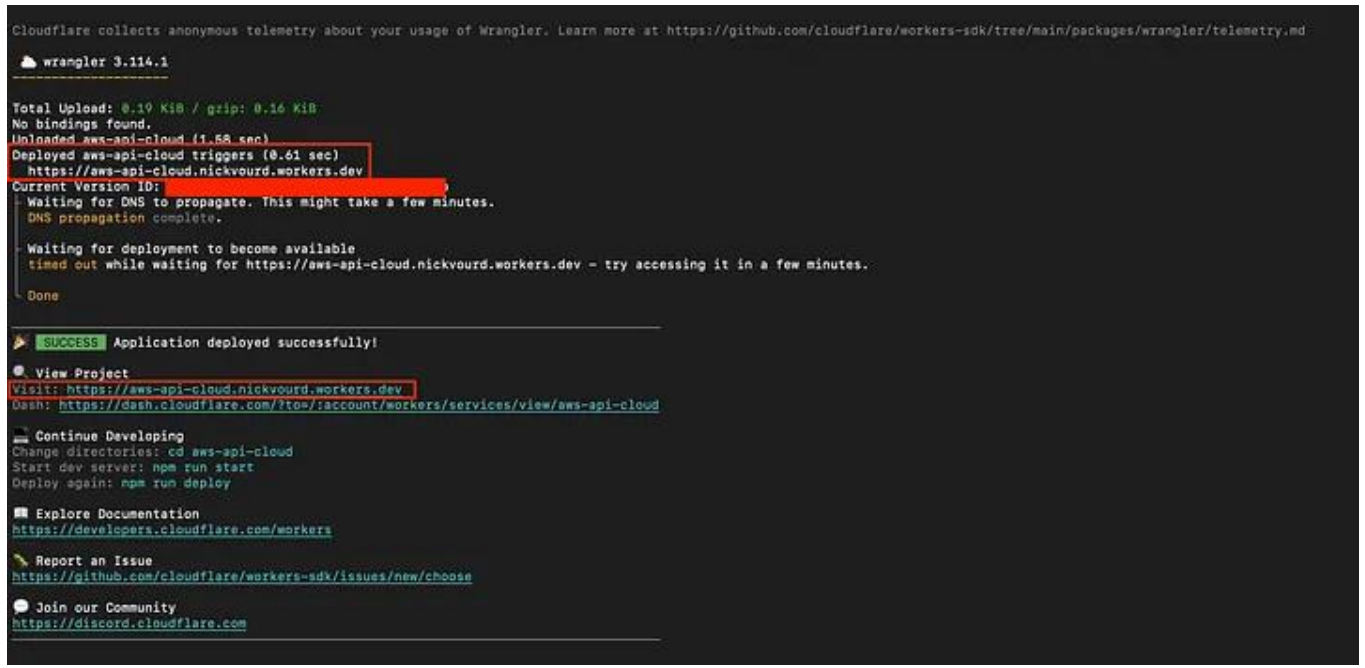
```
npm exec wangler login
```



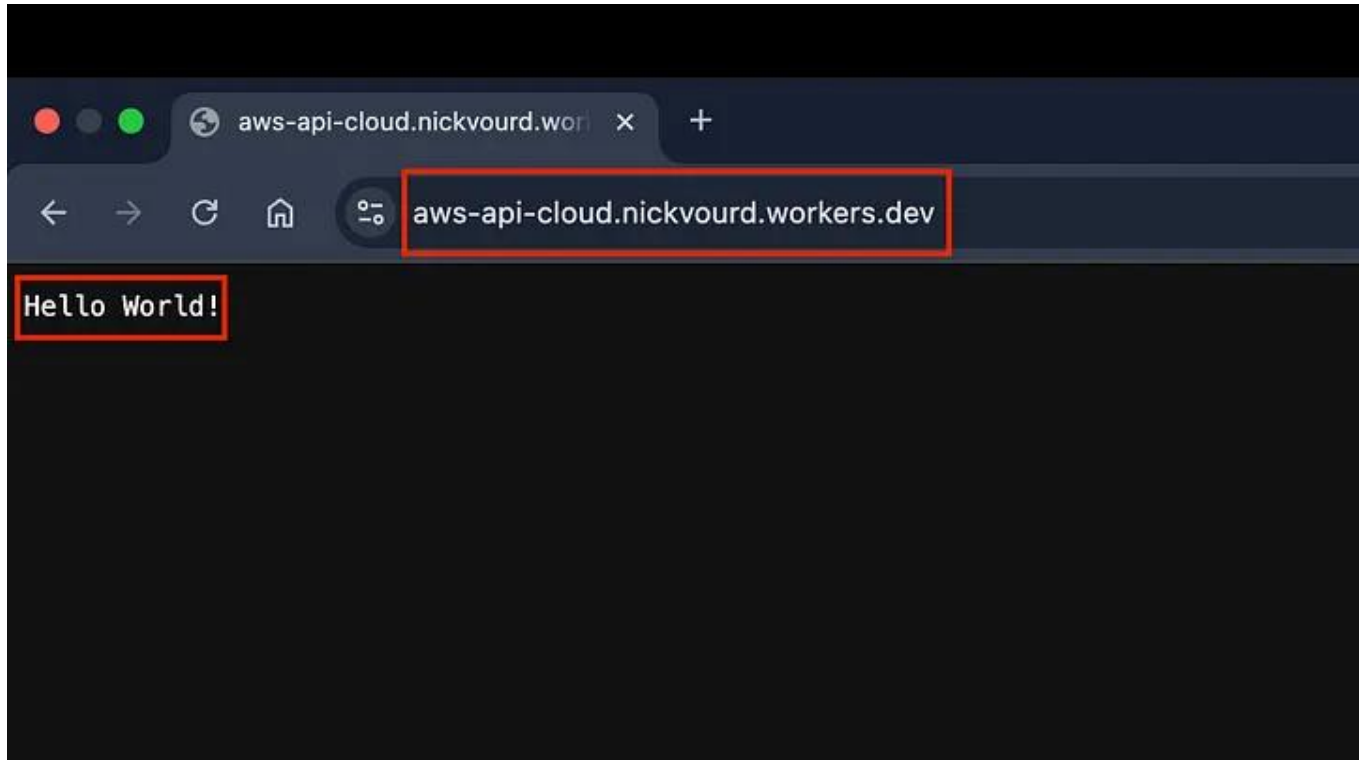
The Rise of the Redirector

The Rise of the Redirector

```
npm create cloudflare -y
```



The Rise of the Redirector

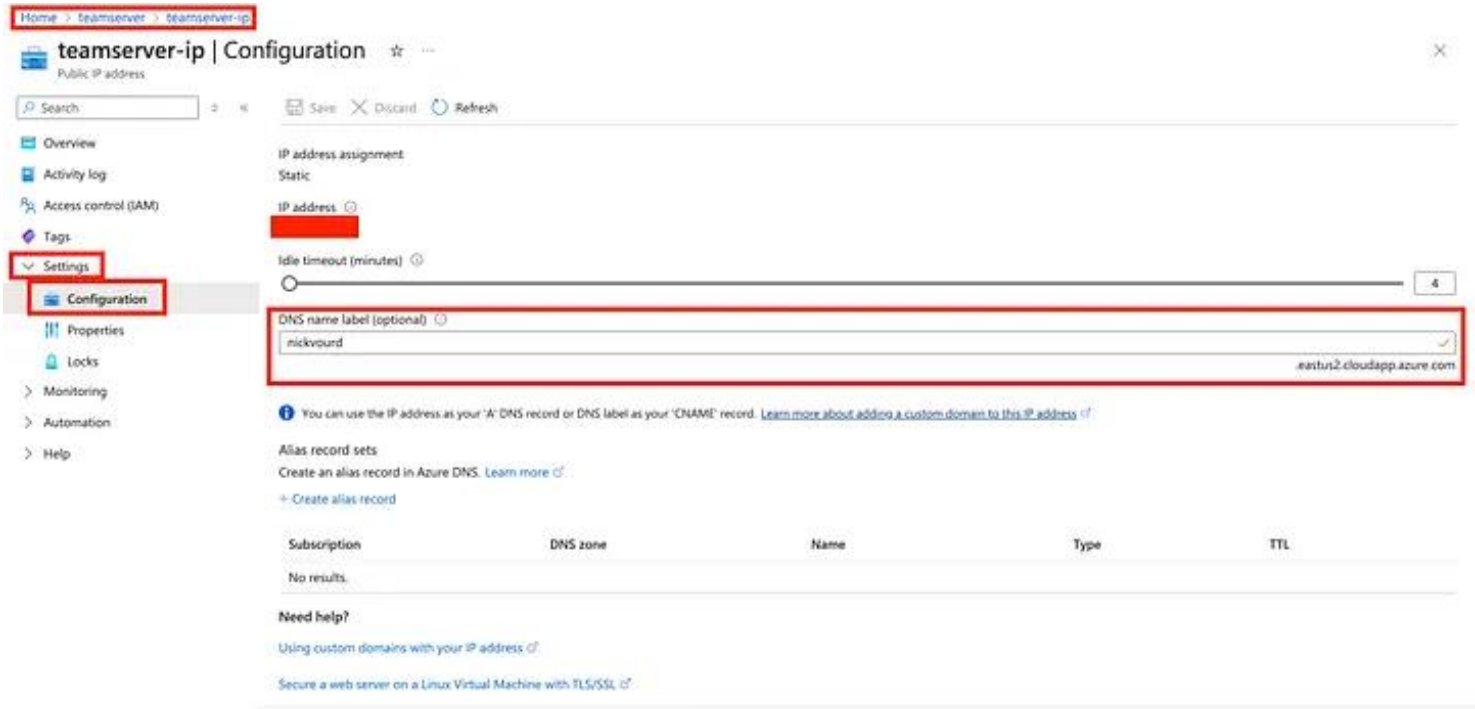


The Rise of the Redirector

The Rise of the Redirector

```
npm exec wrangler deploy
```

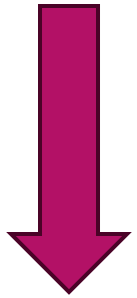
```
[nickvourd@Nikos-MacBook-Pro aws-api-cloud % vim wrangler.jsonc
[nickvourd@Nikos-MacBook-Pro aws-api-cloud % vim src/index.js
[nickvourd@Nikos-MacBook-Pro aws-api-cloud % npm exec wrangler deploy

 ⛅ wrangler 4.0.0
------------------

Total Upload: 1.24 KiB / gzip: 0.51 KiB
Your worker has access to the following bindings:
- Vars:
  - TEAMSERVER: "https://nickvourd.eastus2.cloudapp.az..."
  - REDIRECTOR: "https://aws-api-cloud.nickvourd.worke..."
Uploaded aws-api-cloud (1.39 sec)
Deployed aws-api-cloud triggers (0.24 sec)
  https://aws-api-cloud.nickvourd.workers.dev
Current Version ID:
nickvourd@Nikos-MacBook-Pro aws-api-cloud %
```



Team Server Configurations

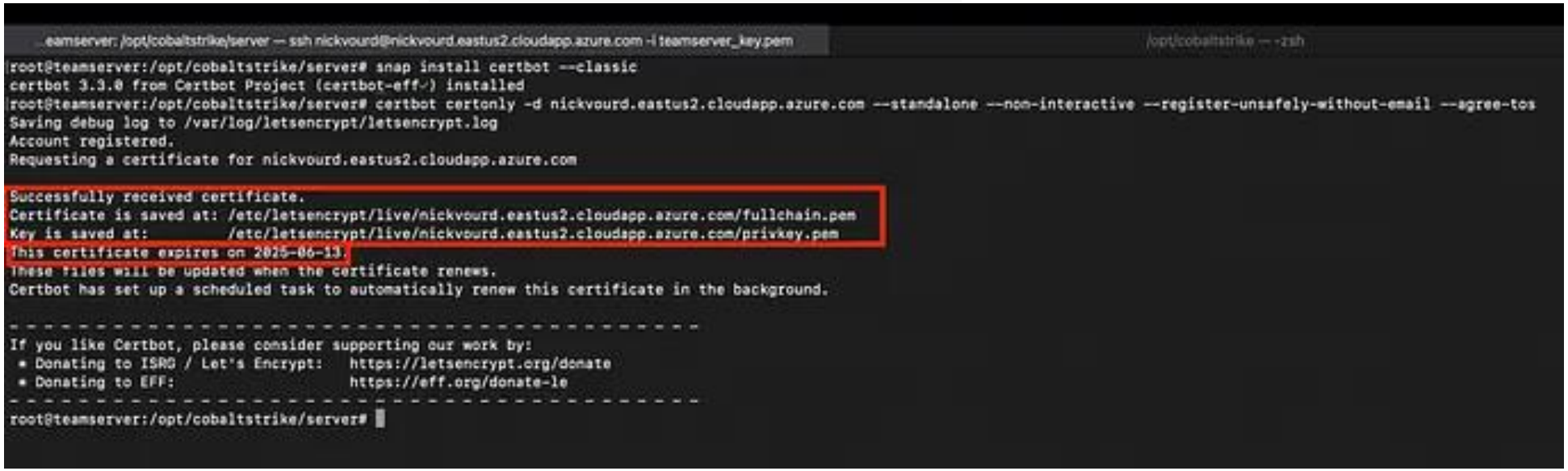
Team Server Configurations

```
snap install certbot --classic
```



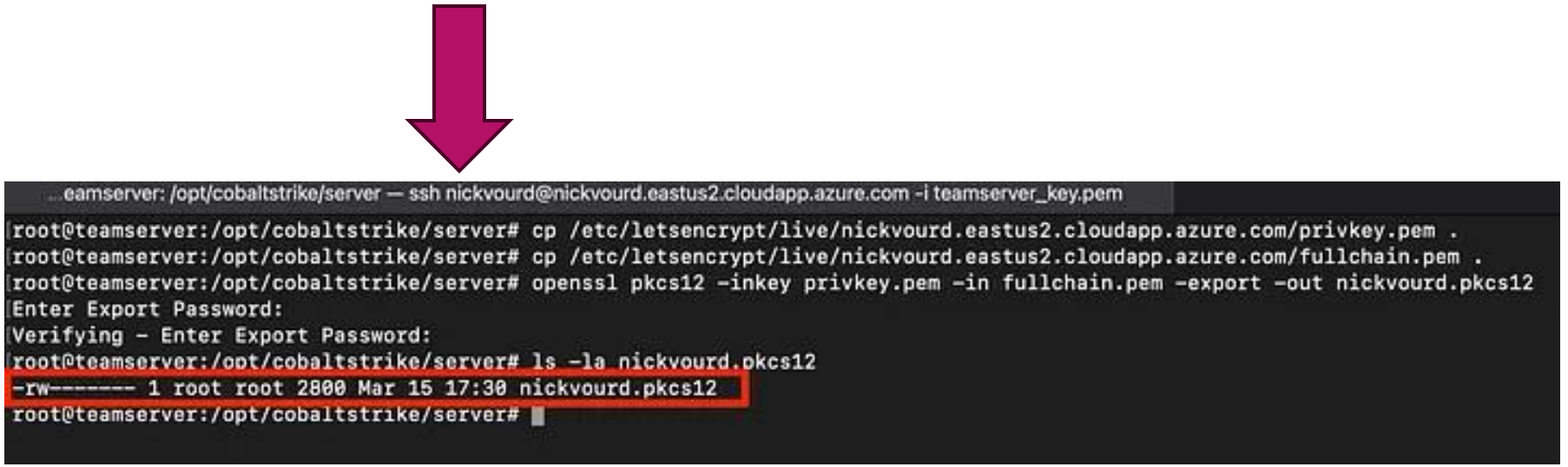
```
certbot certonly -d <fqdn> --standalone --non-interactive --register-unsafely-without-email --agree-tos
```

Team Server Configurations



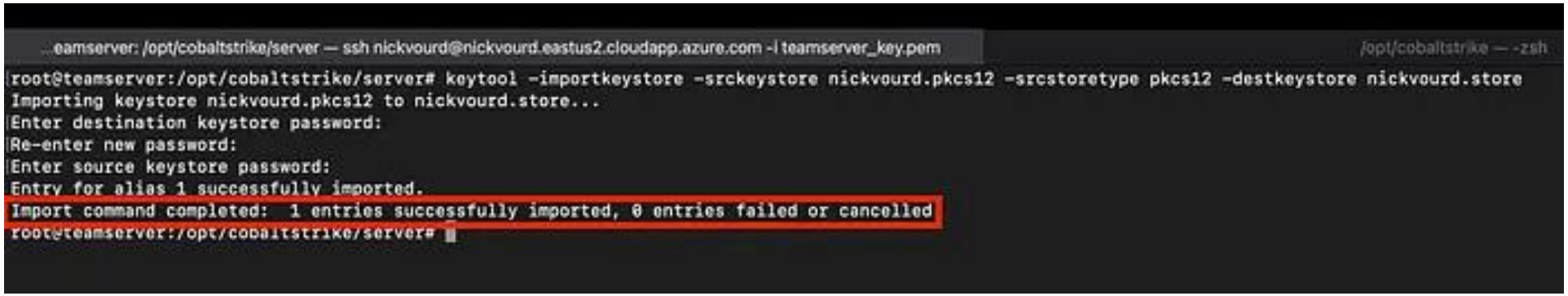
Team Server Configurations

```
openssl pkcs12 -inkey privkey.pem -in fullchain.pem -export -out <filename>.pkcs12
```

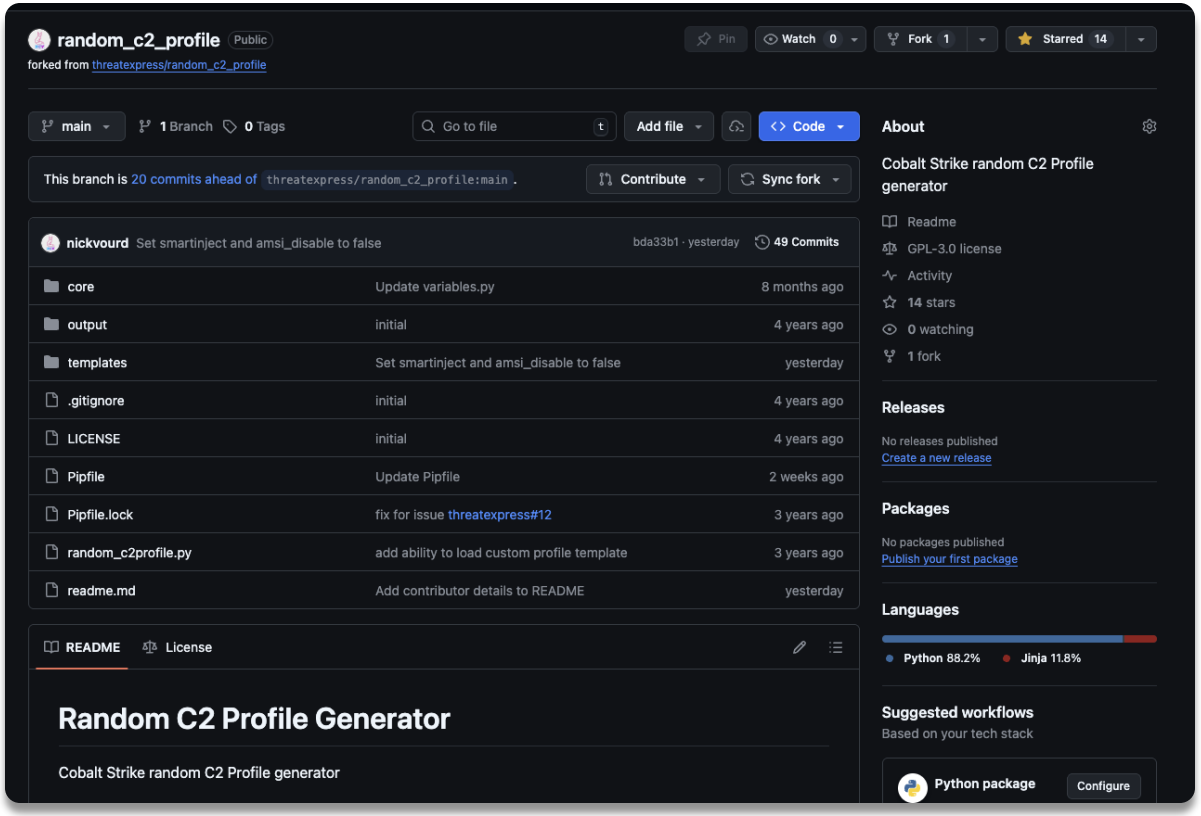


Team Server Configurations

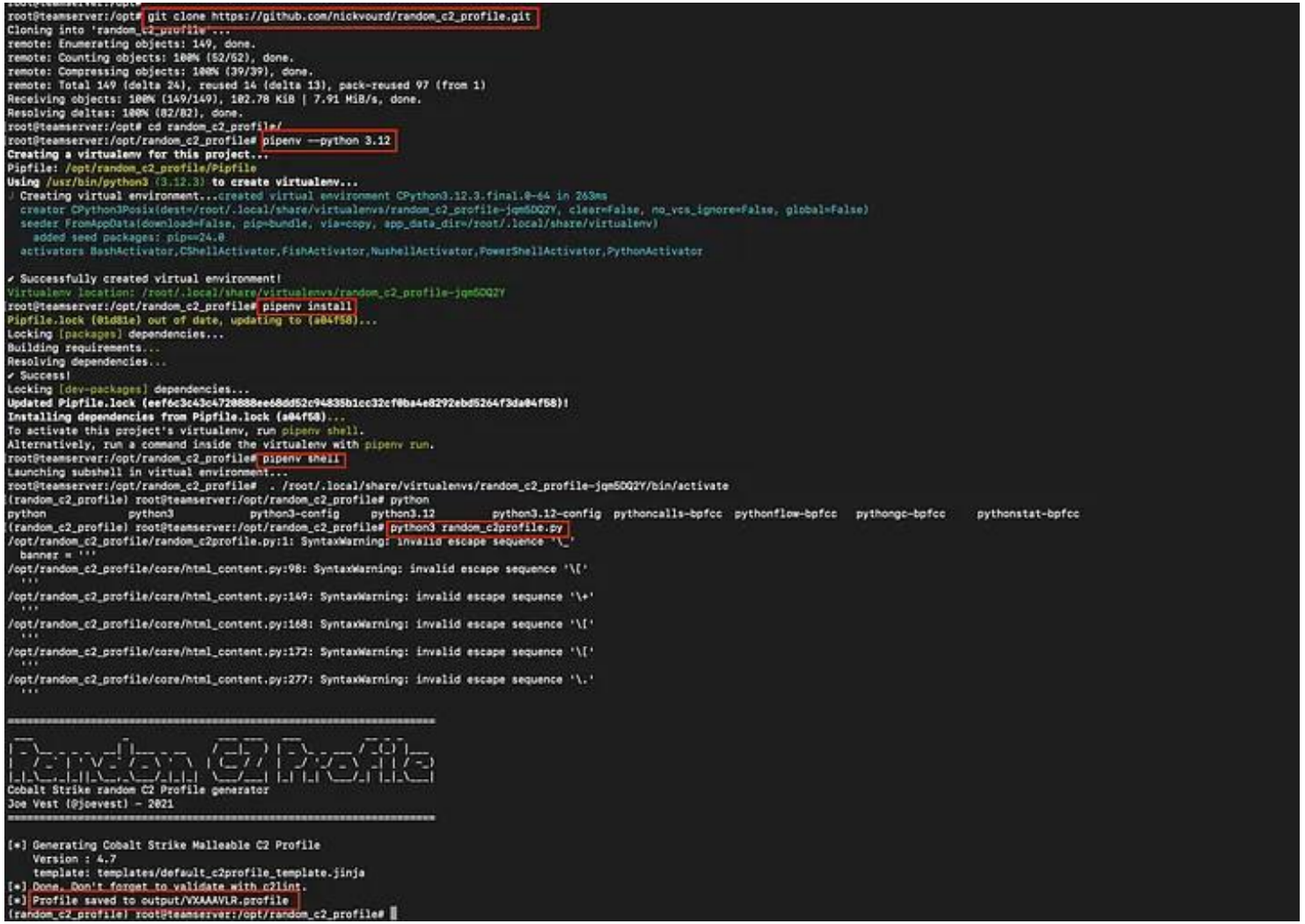
```
keytool -importkeystore -srckeystore <filename_of_pkcs12>.pkcs12 -srcstoretype pkcs12 -destkeystore <filename_keystore>.store
```



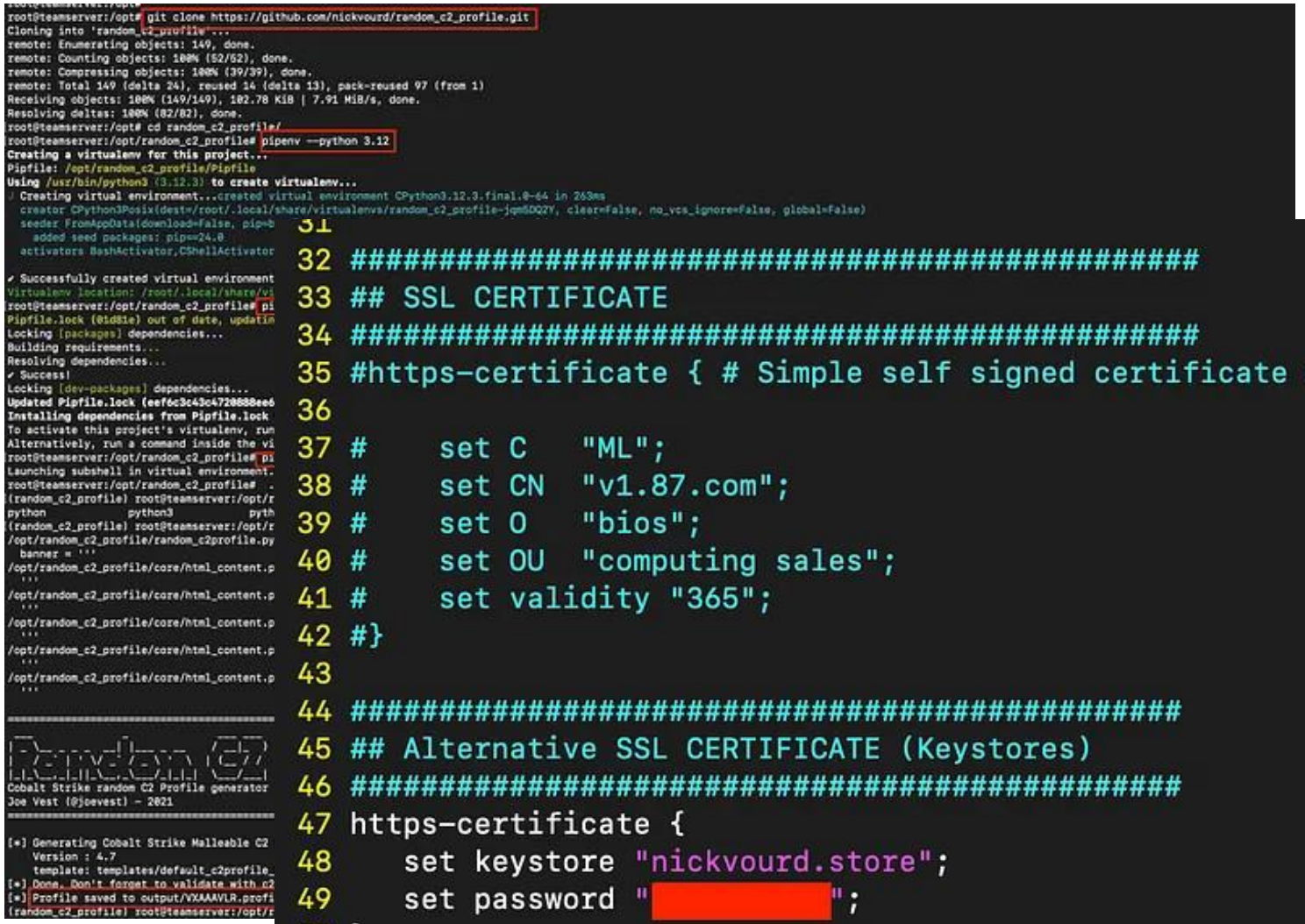
Team Server Configurations



- Project: random_c2_profile
- Original Author: Joe vest (@joevest)
- Original Repo: https://github.com/threatexpress/random_c2_profile
- Modified Repo: https://github.com/nickvourd/random_c2_profile

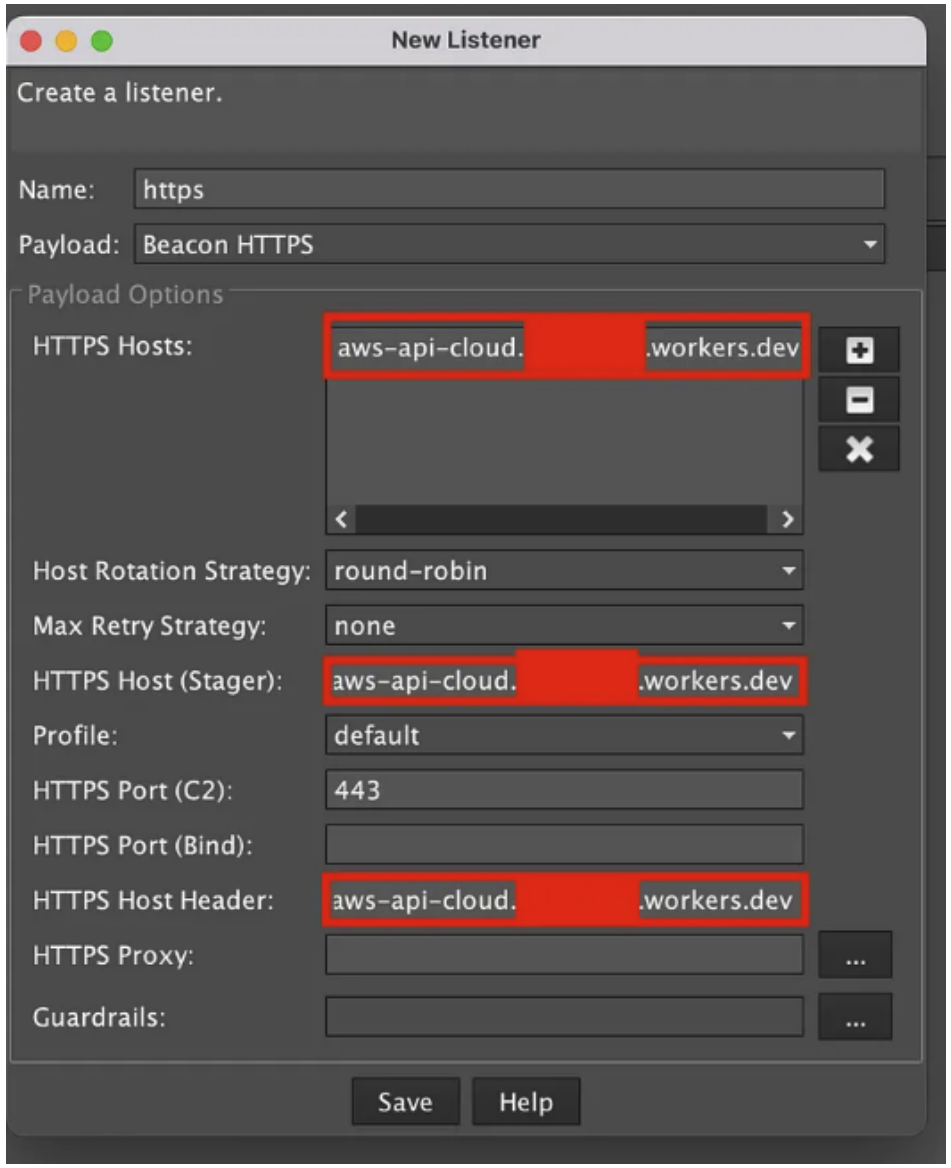


Team Server Configurations

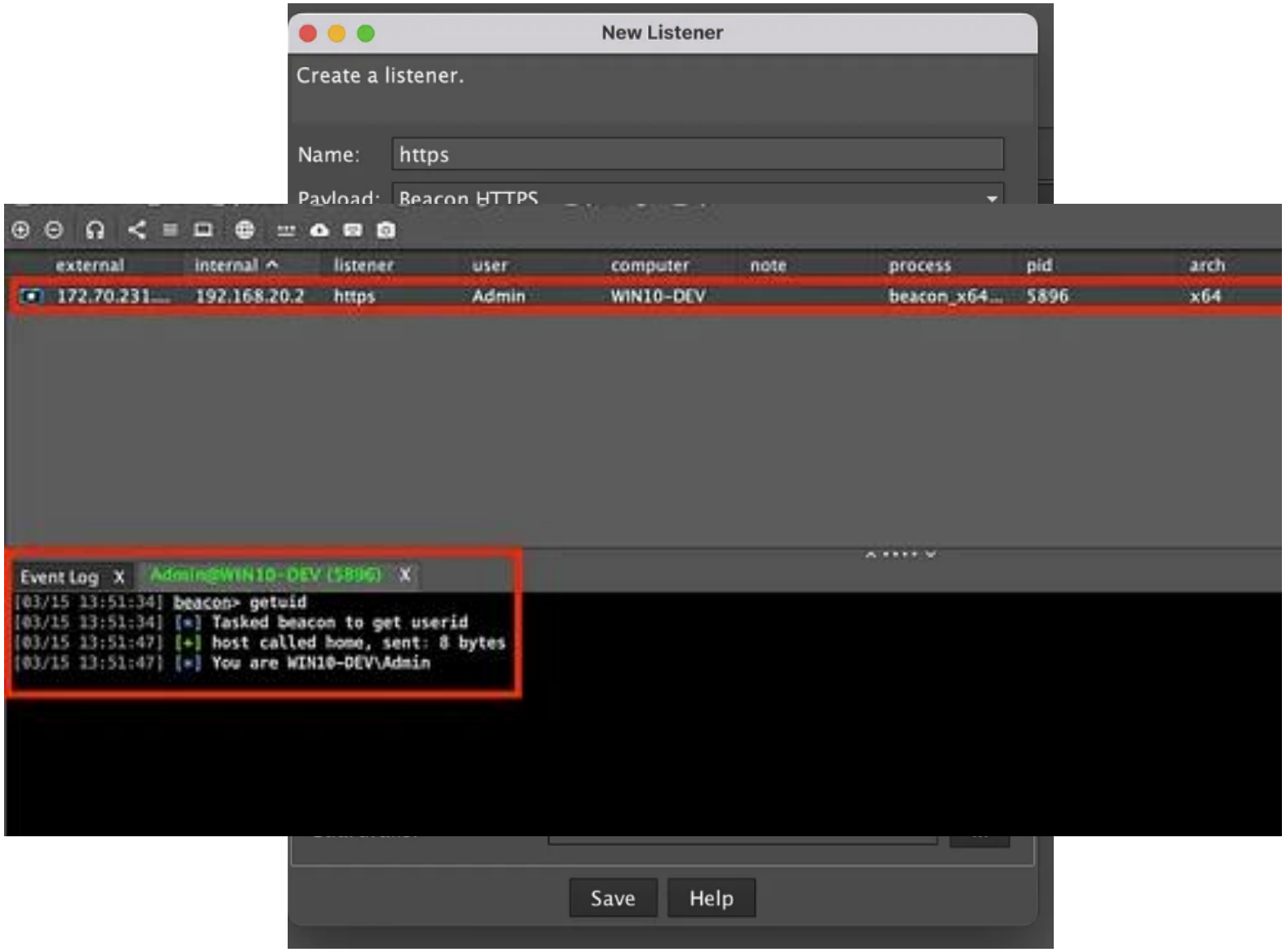



```
31
32 ##################################################
33 ## SSL CERTIFICATE
34 ##################################################
35 #https-certificate { # Simple self signed certificate data
36
37 #     set C    "ML";
38 #     set CN   "v1.87.com";
39 #     set O    "bios";
40 #     set OU   "computing sales";
41 #     set validity "365";
42 #}
43
44 ##################################################
45 ## Alternative SSL CERTIFICATE (Keystores)
46 ##################################################
47 https-certificate {
48     set keystore "nickvourd.store";
49     set password "[redacted]";
50 }
51
```

Team Server Configurations



Team Server Configurations

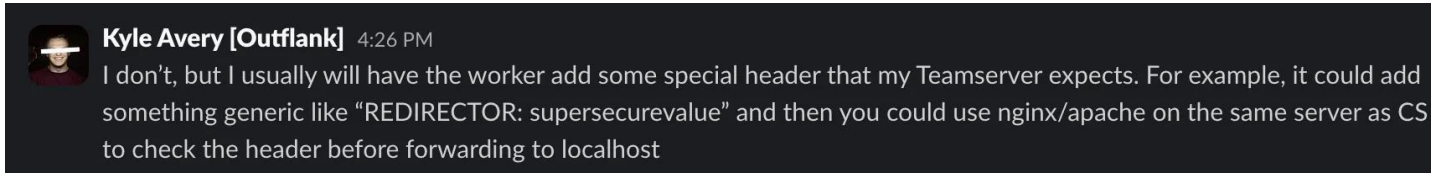


Team Server Configurations

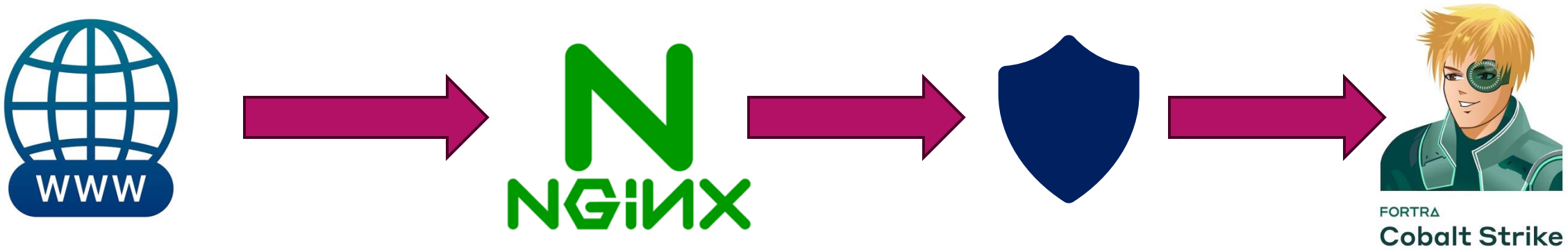
Kyle Saved My Team Server

Kyle Saved My Team Server

Shout out to Kyle Avery (@kyleavery_) for all the guidance



Kyle Saved My Team Server



Kyle Saved My Team Server

TCP/443

TCP/8443

Kyle Saved My Team Server

- wrangler.jsonc

```jsonc
{
  "$schema": "node_modules/wrangler/config-schema.json",
  "name": "aws-api-cloud",
  "main": "src/index.js",
  "compatibility_date": "2025-03-13",
  "workers_dev": true,
  "observability": {
    "enabled": true
  },
  "vars": {
    "HEADERVALUE": "SuperSecretValue",
    "TEAMSERVER": "https://nickvourd.eastus2.cloudapp.azure.com/",
    "REDIRECTOR": "https://aws-api-cloud.nickvourd.workers.dev/"
  }
}
```

Kyle Saved My Team Server

- wrangler.jsonc

```
{
  "$schema": "node_modules/wrangler/config-schema.json",
  "name": "aws-api-cloud",
  "main": "src/index.js",
  "compatibility_date": "2025-03-13",
  "workers_dev": true,
  "observability": {
    "enabled": true
  },
  "vars": {
    "HEADERVALUE": "SuperSecretValue",
    "TEAMSERVER": "https://nickvourd.eastus2.cloudapp.azure.com/",
    "REDIRECTOR": "https://aws-api-cloud.nickvourd.workers.dev/"
  }
}
```

Kyle Saved My Team Server

- Index.js

```
const PRESHARED_AUTH_HEADER_KEY = "X-Custom-Header"

addEventListener('fetch', event => {
  event.respondWith(handleRequest(event))
})

async function handleRequest(event) {
  const request = event.request

  // Check if the method is GET or POST
  if (request.method !== 'GET' && request.method !== 'POST') {
    return new Response(JSON.stringify(
      {
        "Error": "Method not allowed."
      }, null, 2),
      {
        status: 405,
        headers: {
          "content-type": "application/json;charset=UTF-8",
          "Allow": "GET, POST"
        }
      }
    )
  }

  const clonedRequest = request.clone()
```

```
if (psk === HEADERVALUE) {
  try {
    if (request.method === 'POST') {
      const response = await fetch(destUrl, {
        method: 'POST',
        headers: request.headers,
        body: clonedRequest.body,
      })

      return response
    }
    else if (request.method === 'GET') {
      const response = await fetch(destUrl, {
        method: 'GET',
        headers: request.headers
      })
      return response
    }
  } catch (error) {
    return new Response('Error forwarding request', { status: 500 })
  }
} else {
  return new Response(JSON.stringify(
      {
        "Error" : "Authentication Failure."
      }, null, 2),
      {
        status: 401,
        headers: {
          "content-type": "application/json;charset=UTF-8"
        }
```

Kyle Saved My Team Server

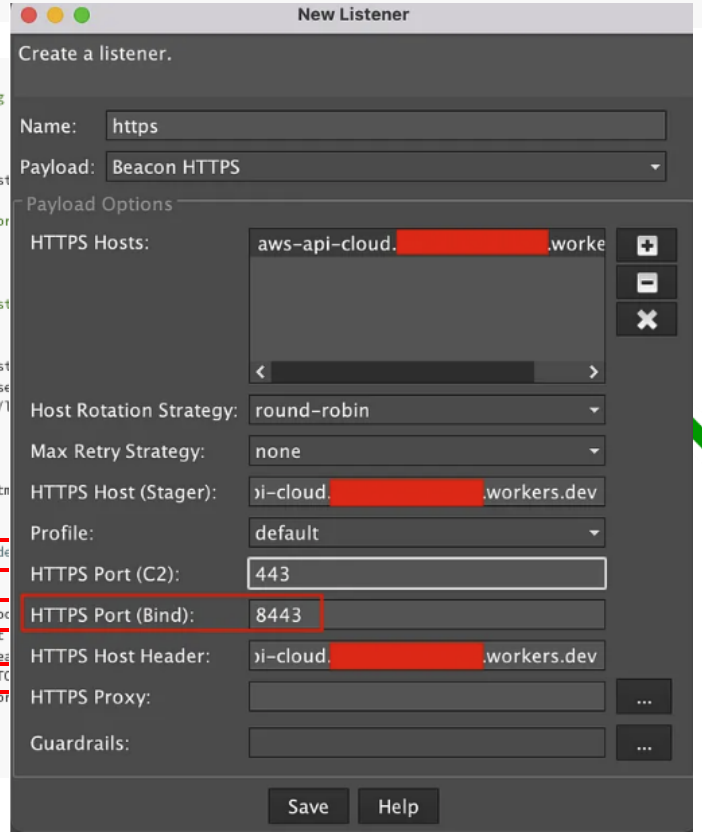
```
certbot certonly -d <fqdn> --nginx --non-interactive --register-unsafely-without-email --agree-tos
```

```
# HTTP server block returning 403 Forbidden
server {
    listen 80;
    listen [::]:80;
    server_name nickvourd.eastus2.cloudapp.azure.com;

    # Return 403 Forbidden for all HTTP requests
    return 403;
}

# HTTPS server block with custom header authorization and proxy setup
server {
    listen 443 ssl;
    server_name nickvourd.eastus2.cloudapp.azure.com;
    ssl_certificate /etc/letsencrypt/live/nickvourd.eastus2.cloudapp.azure.co
    ssl_certificate_key /etc/letsencrypt/live/nickvourd.eastus2.cloudapp.azur
    ssl_protocols TLSv1.3;

    root /var/www/html;
    index index.html index.htm;

    location / {
        if ($http_custom_header != "SuperSecretValue") {
            return 403;
        }
        proxy_pass https://localhost:8443;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header CUSTOM-HEADER $http_custom_header;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Kyle Saved My Team Server

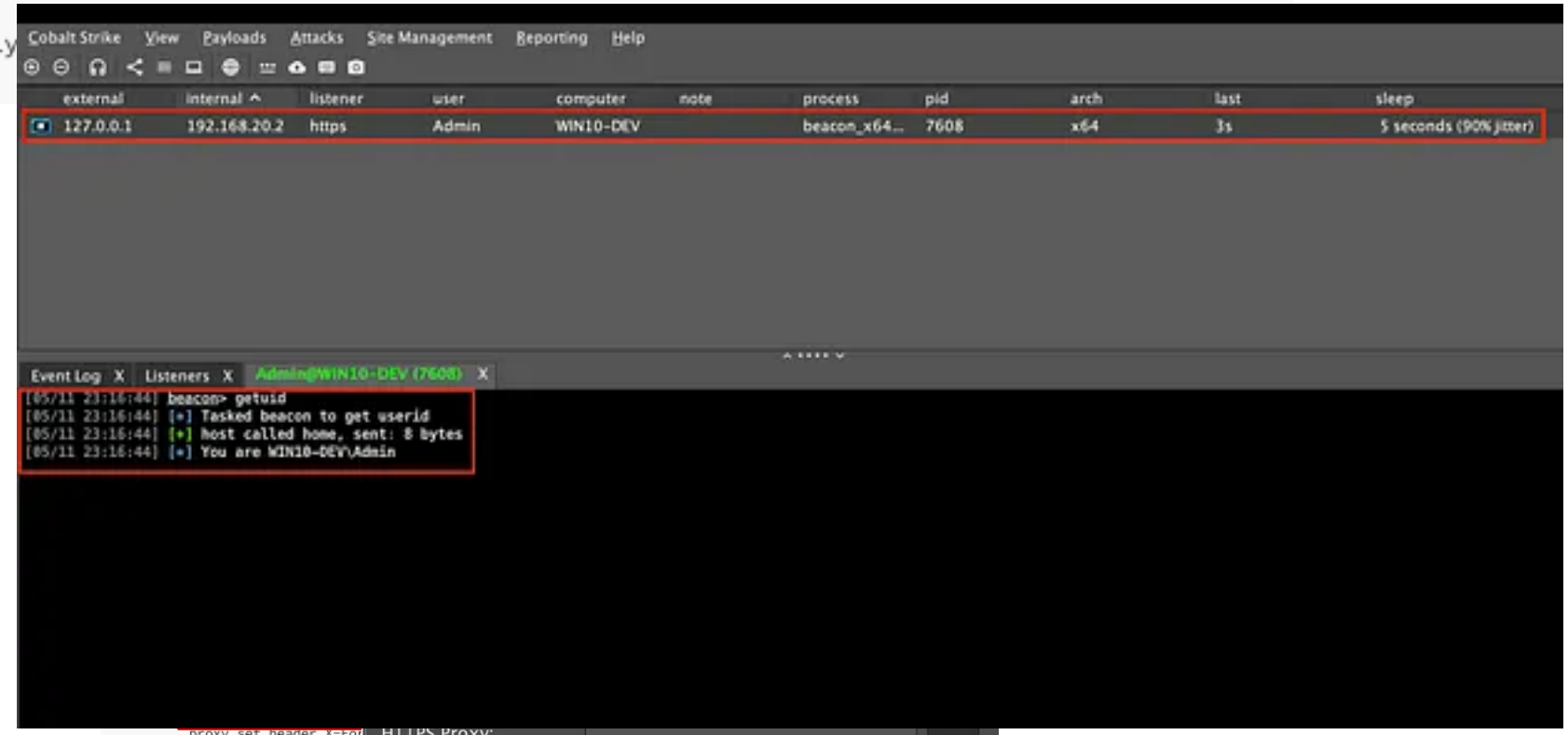
```
certbot certonly -d <fqdn> --nginx --non-interactive --register-unsafely-without-email --agree-tos
```

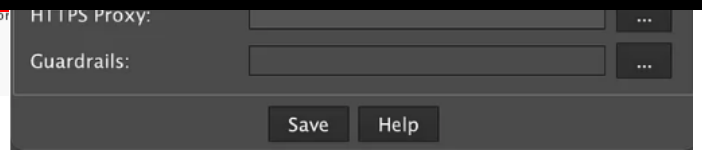
```
# HTTP server block returning
server {
    listen 80;
    listen [::]:80;
    server_name nickvourd.east

    # Return 403 Forbidden for
    return 403;
}

# HTTPS server block with cust
server {
    listen 443 ssl;
    server_name nickvourd.east
    ssl_certificate /etc/letse
    ssl_certificate_key /etc/l
    ssl_protocols TLSv1.3;

    root /var/www/html;
    index index.html index.htm

    location / {
        if ($http_custom_heade
            return 403;
        }
        proxy_pass https://loc
        proxy_set_header Host
        proxy_set_header X-Rea
        proxy_set_header CUSTO
        proxy_set_header X-For
    }
}
```



Kyle Saved My Team Server

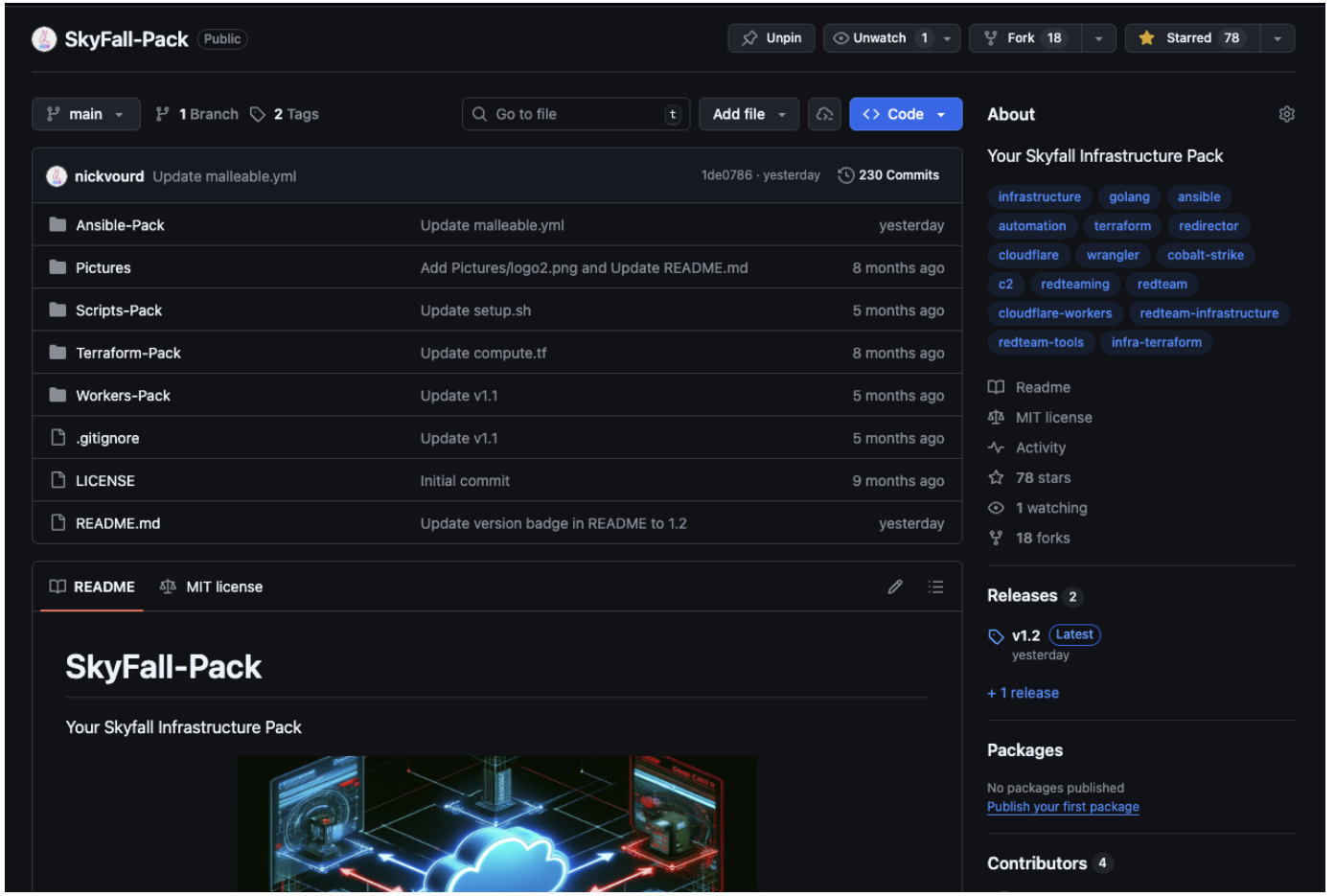
```
certbot certonly
```



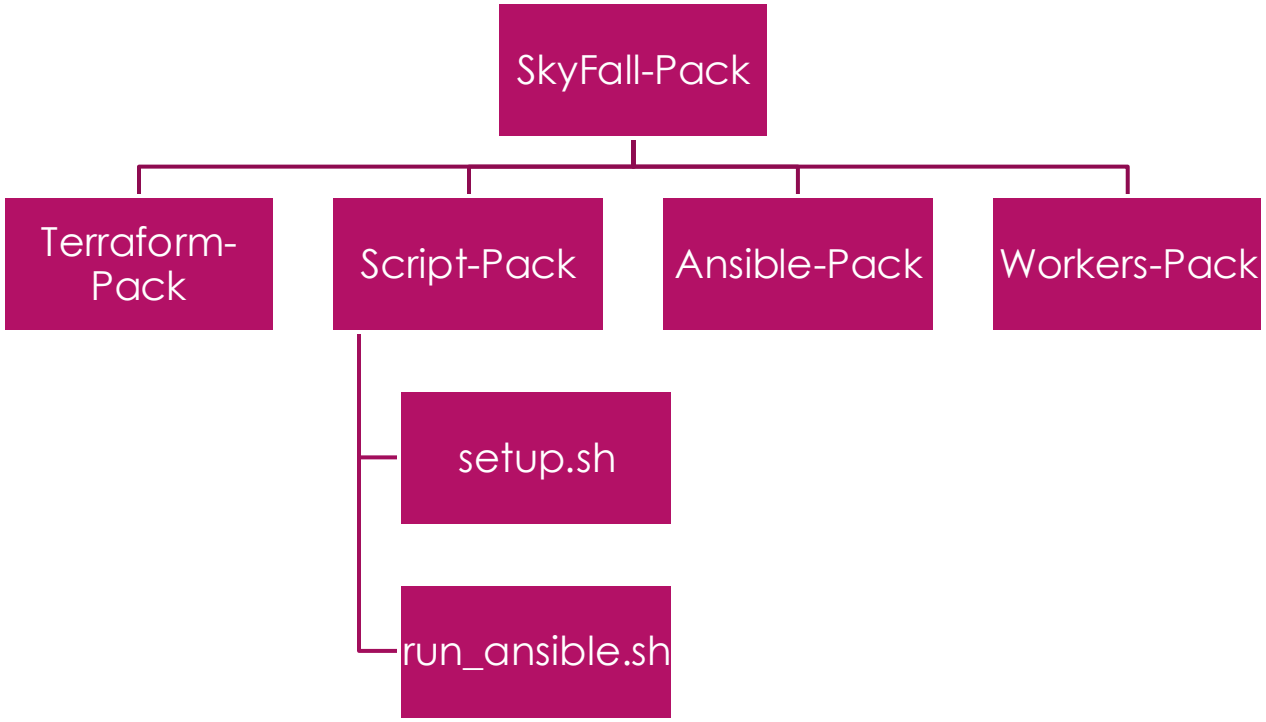


SkyFall-Pack Automation

- Version: 1.2
- Written: Golang, Terraform, Ansible && Bash Scripting
- Authors: @nickvourd, @kavasilo
- Integrated with Cobalt Stike/Outflank Stage 1 C2



SkyFall-Pack Automation



SkyFall-Pack Automation

```
nickvourd@Nikoss-Mac-mini Scripts-Pack % ./setup.sh -l eastus -u nickvourd -n example -s teamserver -d skyfall -v standard_b1ms
[+] terraform.tfvars in SkyFall-Pack/Terraform-Pack has been updated with:

VM-Location: eastus
Username: nickvourd
Resource Prefix: example
SSH Key Name: teamserver
DNS Name: skyfall
VM Size: standard_b1ms

[*] Initializing Terraform...


Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~> 3.0"...
- Finding hashicorp/random versions matching "~> 3.0"...
- Finding hashicorp/tls versions matching "~> 4.0"...
- Finding latest version of hashicorp/local...
- Finding azure/azapi versions matching "~> 1.5"...
- Installing hashicorp/tls v4.1.0...
- Installed hashicorp/tls v4.1.0 (signed by HashiCorp)
- Installing hashicorp/local v2.5.3...
- Installed hashicorp/local v2.5.3 (signed by HashiCorp)
- Installing azure/azapi v1.15.0...
- Installed azure/azapi v1.15.0 (signed by a HashiCorp partner, key ID 6F0B91BDE98478CF)
- Installing hashicorp/azurerm v3.117.1...
- Installed hashicorp/azurerm v3.117.1 (signed by HashiCorp)
- Installing hashicorp/random v3.7.2...
- Installed hashicorp/random v3.7.2 (signed by HashiCorp)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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
```

SkyFall-Pack Automation

```
nickvourd@Nikoss-Mac-mini Scripts-Pack % ./setup.sh -l eastus -u nickvourd -n example -s teamserver -d skyfall -v standard_b1ms
[+] terraform.tfvars in SkyFall-Pack/Terraform-Pack has been updated with:

VM-Location: eastus
Username: nickvourd
Resource Prefix: example
SSH Key Name: teamserver
DNS Name: skyfall
VM Size: standard_b1ms

[*] Initializing Terraform...


Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~> 3.0"...
- Finding hashicorp/random versions matching "~> 3.0"...
- Finding hashicorp/tls versions matching "~> 4.0"...
- Finding latest version of hashicorp/local...
- Finding azure/azapi versions matching "~> 1.5"...
- Installing hashicorp/tls v4.1.0...
- Installed hashicorp/tls v4.1.0 (signed by HashiCorp)
- Installing hashicorp/local v2.5.3...
- Installed hashicorp/local v2.5.3 (signed by HashiCorp)
- Installing azure/azapi v1.15.0...
- Installed azure/azapi v1.15.0 (signed by a HashiCorp partner, key ID 6F0B91BDE98478CF)
- Installing hashicorp/azurerm v3.117.1...
- Installed hashicorp/azurerm v3.117.1 (signed by HashiCorp)
- Installing hashicorp/random v3.7.2...
- Installed hashicorp/random v3.7.2 (signed by HashiCorp)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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
```

```
Apply complete! Resources: 11 added, 0 changed, 0 destroyed.

Outputs:

connection_string = "ssh -i teamserver.pem nickvourd@skyfall-h95dbuo9.eastus.cloudapp.azure.com"
fqdn = "skyfall-h95dbuo9.eastus.cloudapp.azure.com"
public_ip = "52.191.116.184"
resource_group_name = "rg-example-h95dbuo9"
ssh_privkey = <sensitive>
username = "nickvourd"

[*] Getting connection information...

[*] Connection String:
"ssh -i teamserver.pem nickvourd@skyfall-h95dbuo9.eastus.cloudapp.azure.com"

[*] FQDN:
"skyfall-h95dbuo9.eastus.cloudapp.azure.com"

[*] Public IP:
"52.191.116.184"

[*] Username:
"nickvourd"

nickvourd@Nikoss-Mac-mini Scripts-Pack %
```

SkyFall-Pack Automation

```
nickvourd@Nikoss-Mac-mini Scripts-Pack % ./run_ansible.sh -f nickvourd -p 'Asa31904#!' -c 'X-CSRF-Token' -s 'SuperSecret1234!'

[+] Running Ansible playbook with:
VM IP: 52.191.116.184
Username: nickvourd
SSH Key: /Users/nickvourd/Documents/GitHub/SkyFall-Pack/Terraform-Pack/teamserver.pem
VM FQDN: skyfall-h95dbuo9.eastus.cloudapp.azure.com
Keystore Filename: nickvourd
Keystore Password: Asa31904#!
Teamserver Port: 8443
Custom Header: X-CSRF-Token
Custom Header Lower: x_csrf_token
Custom Secret: SuperSecret1234!
Protocol: https

ansible-playbook [core 2.19.3]
  config file = /Users/nickvourd/Documents/GitHub/SkyFall-Pack/Ansible-Pack/ansible.cfg
  configured module search path = ['/Users/nickvourd/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /opt/homebrew/Cellar/ansible/12.1.0/libexec/lib/python3.13/site-packages/ansible
  ansible collection location = /Users/nickvourd/.ansible/collections:/usr/share/ansible/collections
  executable location = /opt/homebrew/bin/ansible-playbook
  python version = 3.13.9 (main, Oct 14 2025, 13:52:31) [Clang 17.0.0 (clang-1700.3.19.1)] (/opt/homebrew/Cellar/ansible/12.1.0/libexec/bin/python)
  jinja version = 3.1.6
  pyyaml version = 6.0.3 (with libyaml v0.2.5)
Using /Users/nickvourd/Documents/GitHub/SkyFall-Pack/Ansible-Pack/ansible.cfg as config file
Skipping callback 'minimal', as we already have a stdout callback.
Skipping callback 'oneline', as we already have a stdout callback.

PLAYBOOK: setup.yml ************************************************************************************************************************************************************************************
1 plays in setup.yml

PLAY [Configure Azure VM] ******************************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
task path: /Users/nickvourd/Documents/GitHub/SkyFall-Pack/Ansible-Pack/setup.yml:2
```

Detections

- **Check URI Patterns**: Repeated use of simple paths such as /get, /cmd, /update and others.
- **Detect Beaconing**: Look for consistent intervals and low-volume HTTPS traffic.
- **Baseline Behavior**: Alert on first-time or uncommon access to workers.dev per host.
- **Analyze Logs**: Identify connections to workers.dev initiated by non-browser processes.

Refences

- https://ajpc500.github.io/c2/Using-CloudFlare-Workers-as-Redirectors/
- https://byt3bl33d3r.substack.com/p/revisiting-cloudflare-workers-for
- https://labs.jumpsec.com/putting-the-c2-in-c2loudflare/


Questions