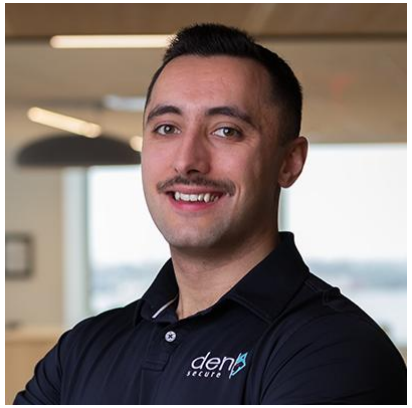
INTRO TO INFRASTRUCTURE AUTOMATION FOR OFFENSIVE SECURITY

HackRedCon • Alex Martirosyan OSEP, CRTO , OSCP, GPEN

WHOAMI

- 6+ years in offensive security
- IT Audit > Penetration Testing
- Interested in intersection of mathematics and security



Alex Martirosyan, OSEP, CRTO , OSCP, GPEN

Lead Penetration Tester, DenSecure

AMartirosyan@wolfandco.com

617.261.8138

https://www.linkedin.com/in/alex-martirosyan/

https://twitter.com/almartiros

https://www.wolfandco.com/services/densecure/

AGENDA

- Offensive Security Background
- Infrastructure Automation
- Existing Wrappers and Tools
- Examples to Manage Phishing/C2
- Lessons Learned
- Closing Thoughts

OFFENSIVE SECURITY BACKGROUND

CURRENT TRENDS

- Offensive security is getting more difficult to do **right**
- Penetration tests are becoming a requirement
- Some organizations have matured, and others are **lucky**

WHAT HAPPENED?

- Endpoint detection response capture all the **telemetry**
  - **https://www.edr-telemetry.com/windows.html**

- Defenders monitor and watch what is published

- Burning tradecraft is not worth the effort

THE OLD DAYS

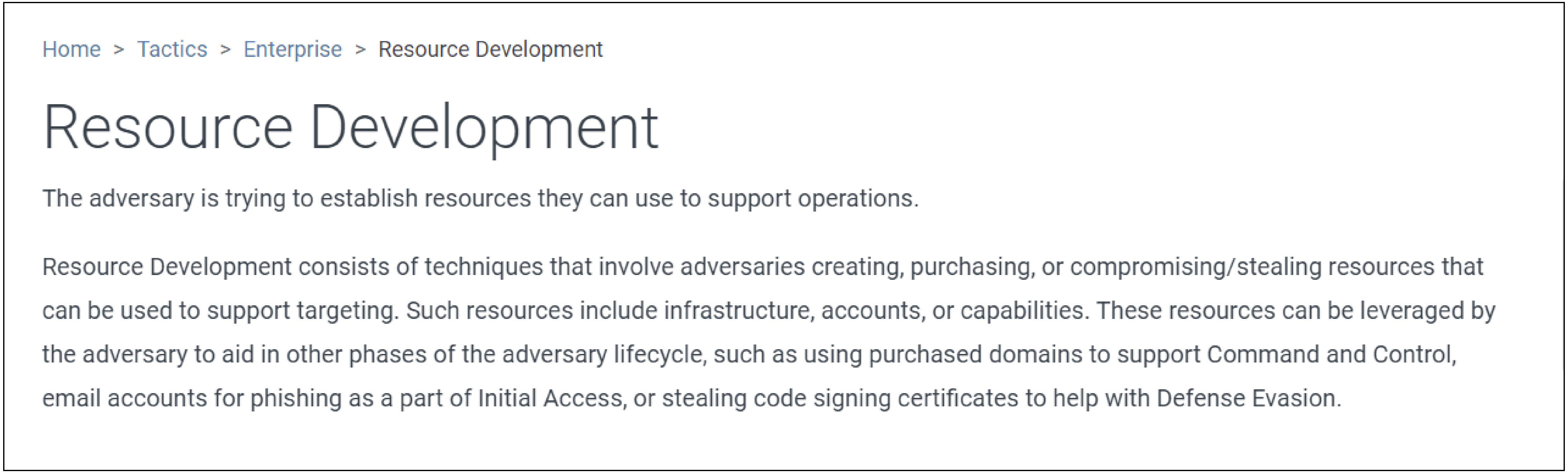
- "Competitive Advantage" brings challenges
- Offensive teams must manage infrastructure
- PoCs require more customization

THREAT ACTORS ADAPT

Beginning in 2022, UNC2565 began incorporating notable changes to the tactics, techniques, and procedures (TTPs) used in its operations. These changes include the use of multiple variations of the FONELAUNCH launcher, the distribution of new follow-on payloads, and changes to the GOOTLOADER downloader and infection chain, including the introduction of GOOTLOADER.POWERSHELL. These changes are illustrative of UNC2565's active development and growth in capabilities.

https://cloud.google.com/blog/topics/threat-intelligence/tracking-evolution-gootloader-operations/

DEVELOP RESOURCES



Home > Tactics > Enterprise > Resource Development

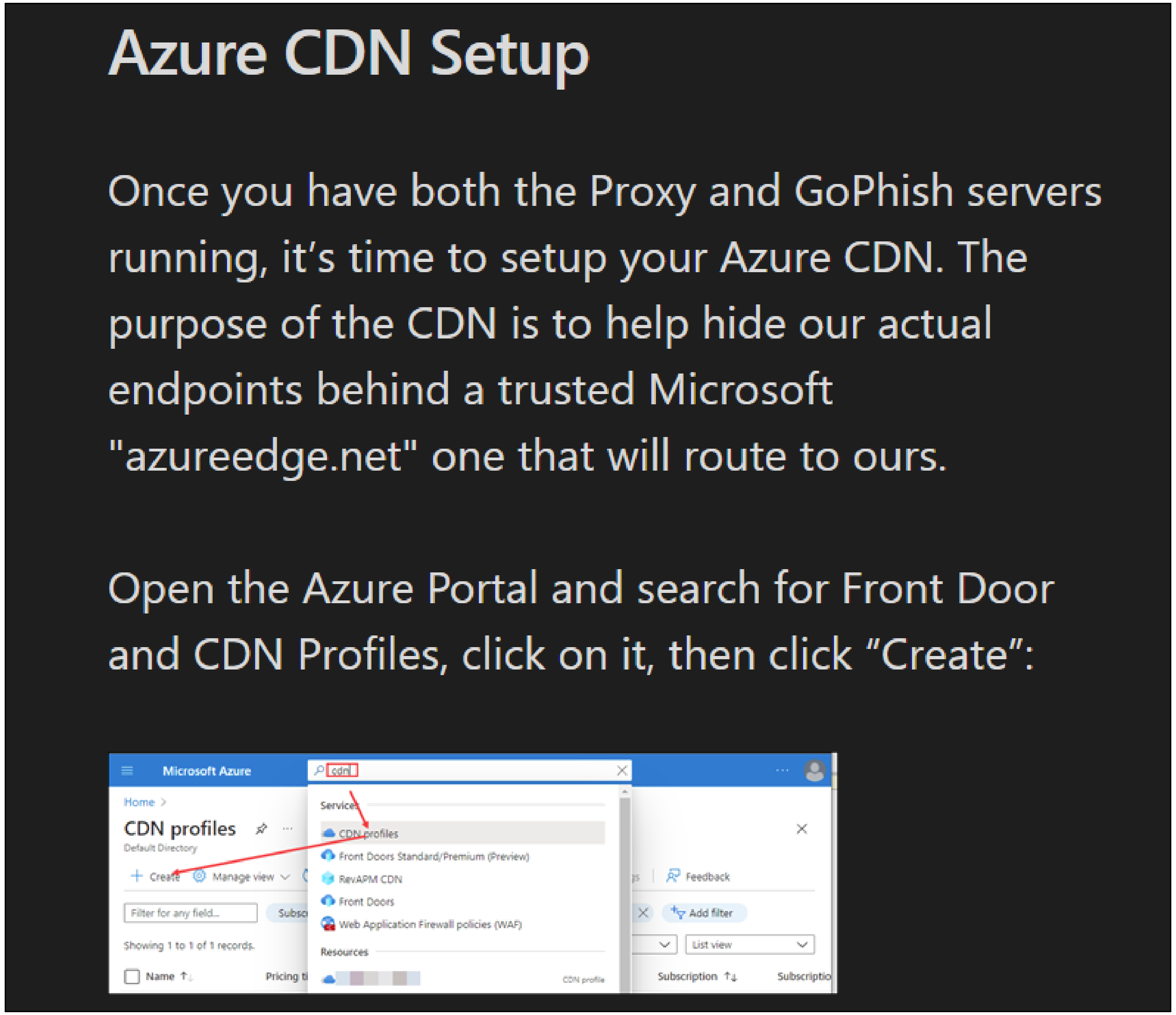
Resource Development

The adversary is trying to establish resources they can use to support operations.

Resource Development consists of techniques that involve adversaries creating, purchasing, or compromising/stealing resources that can be used to support targeting. Such resources include infrastructure, accounts, or capabilities. These resources can be leveraged by the adversary to aid in other phases of the adversary lifecycle, such as using purchased domains to support Command and Control, email accounts for phishing as a part of Initial Access, or stealing code signing certificates to help with Defense Evasion.

IDENTIFYING REPEATABLE TASKS

- What does our team understand well?
- Why should we try to automate a task?
- Is this something cool or adds value?
- Learn to walk before running



Azure CDN Setup

Once you have both the Proxy and GoPhish servers running, it's time to setup your Azure CDN. The purpose of the CDN is to help hide our actual endpoints behind a trusted Microsoft "azureedge.net" one that will route to ours.

Open the Azure Portal and search for Front Door and CDN Profiles, click on it, then click "Create":

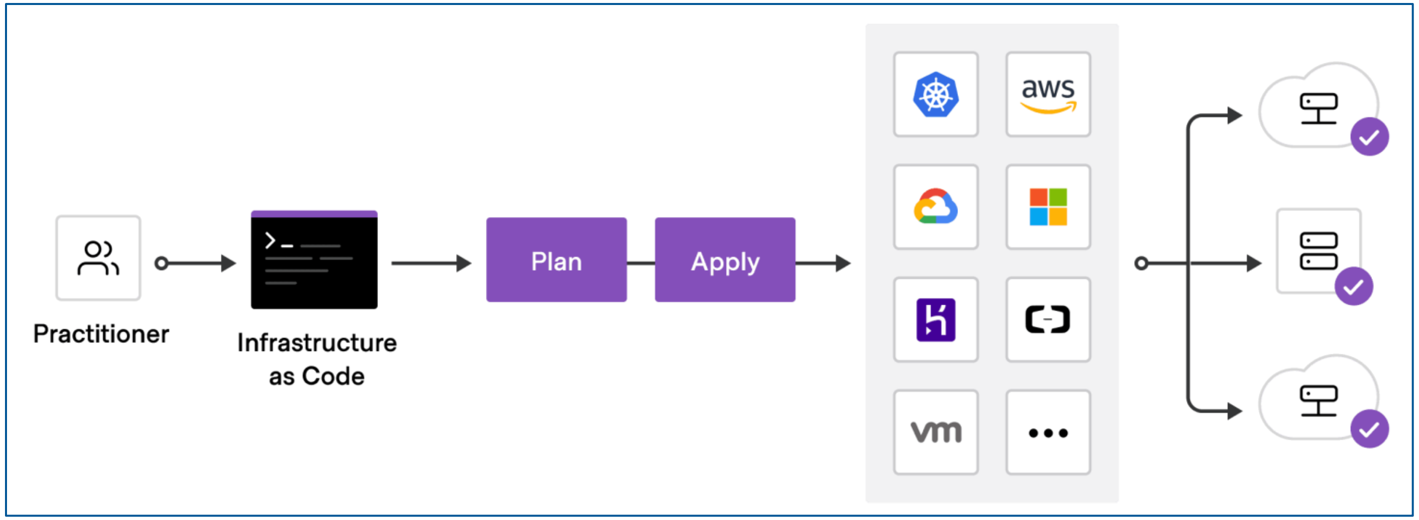
TERRAFORM & ANSIBLE

PRINCIPLES OF IaC

- Version Control
  - Maintaining master templates is key for success
  - Allows any team member to make meaningful contributions
  - GitHub, GitLab, etc.
- Consistent Updates and Deployments
  - “Standard” deployments of infrastructure we plan to deploy
  - Quality control and ability to fix errors quickly
- Ability to Scale
  - Offensive security operations expand -> more to automate

TERRAFORM OVERVIEW

- Learn:
  - https://www.antisyphontraining.com/course/hackerops-with-ralph-may/
  - https://github.com/warhorse/warhorse
  - https://github.com/froyo75/SpREaD



AUTOMATE!

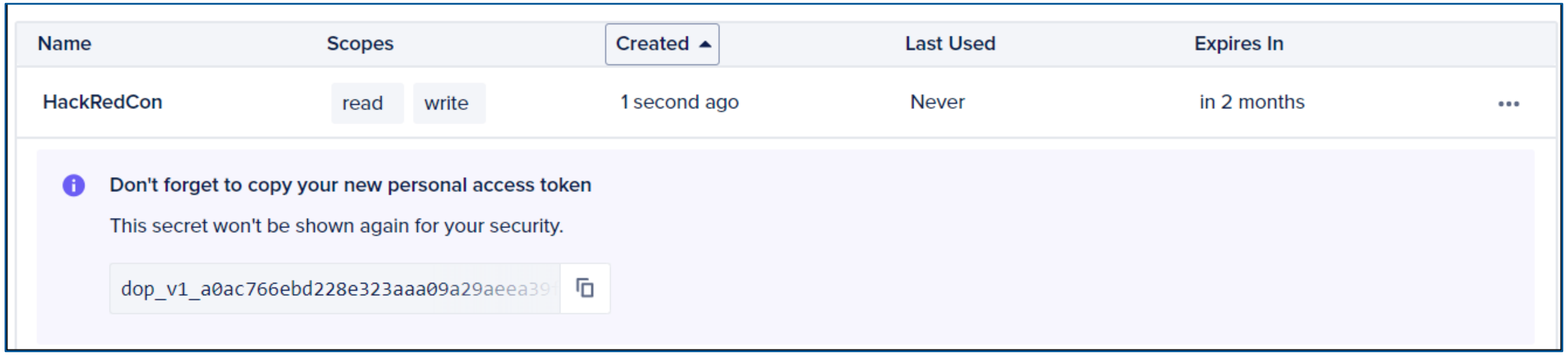
- Identify provider we want to deploy to this can be Azure, DigitalOcean, AWS

- Terraform uses a “working directory” to initialize, plan, and deploy configurations

- We use Terraform to automatically deploy standard configurations and templates
  - CobaltStrike, GoPhish/Evilginx2, Mythic, Redirectors

- When we are done testing we can then destroy the entire deployment

BASIC CONFIG

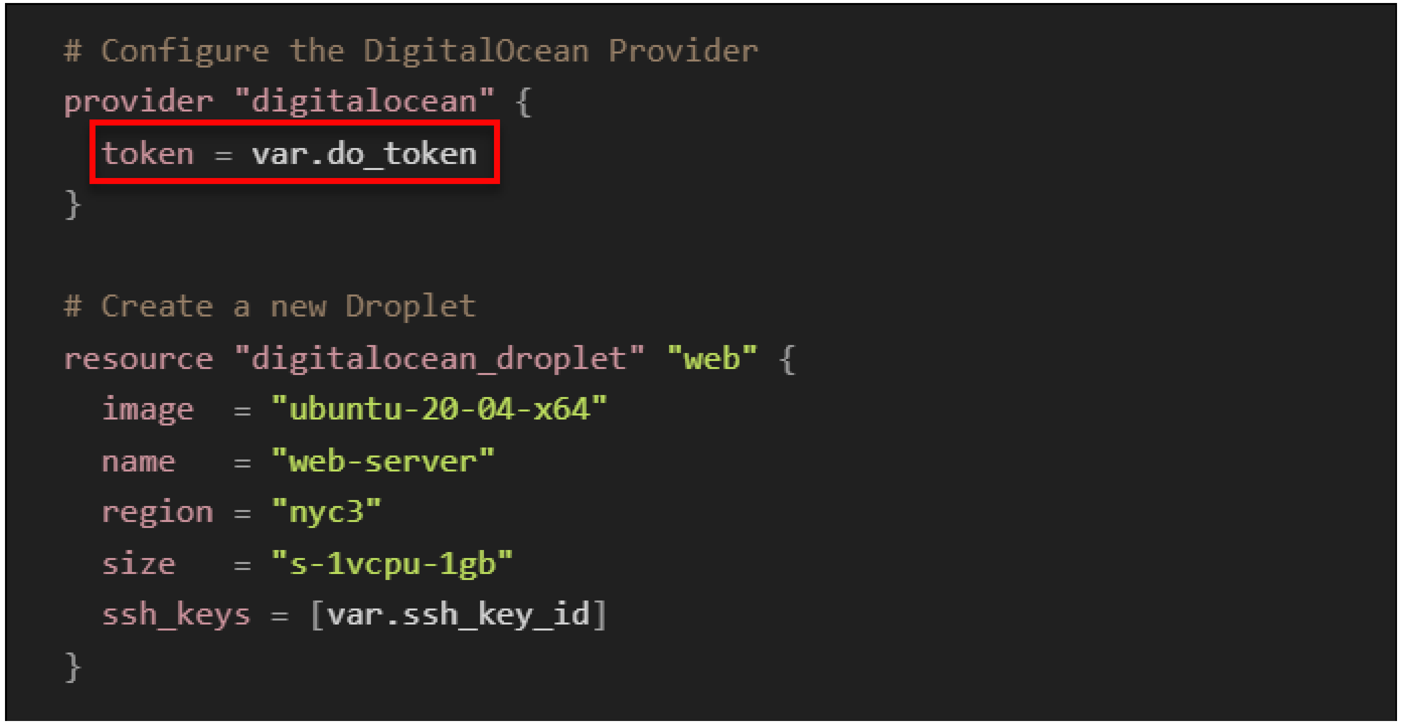


Name	Scopes	Created ▲	Last Used	Expires In	
HackRedCon	read write	1 second ago	Never	in 2 months	•••

Don't forget to copy your new personal access token

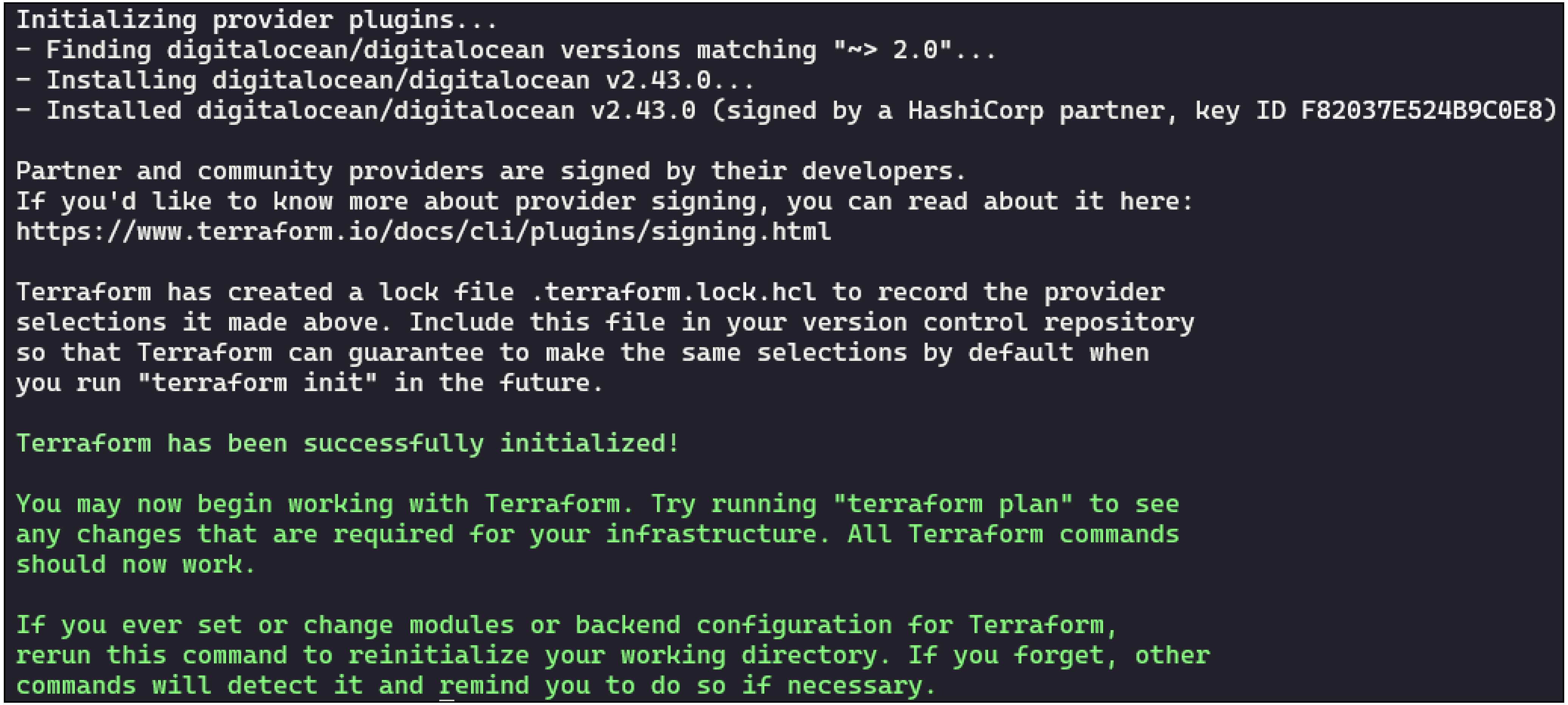
This secret won't be shown again for your security.

```
dop_v1_a0ac766ebd228e323aaa09a29aeea39
```



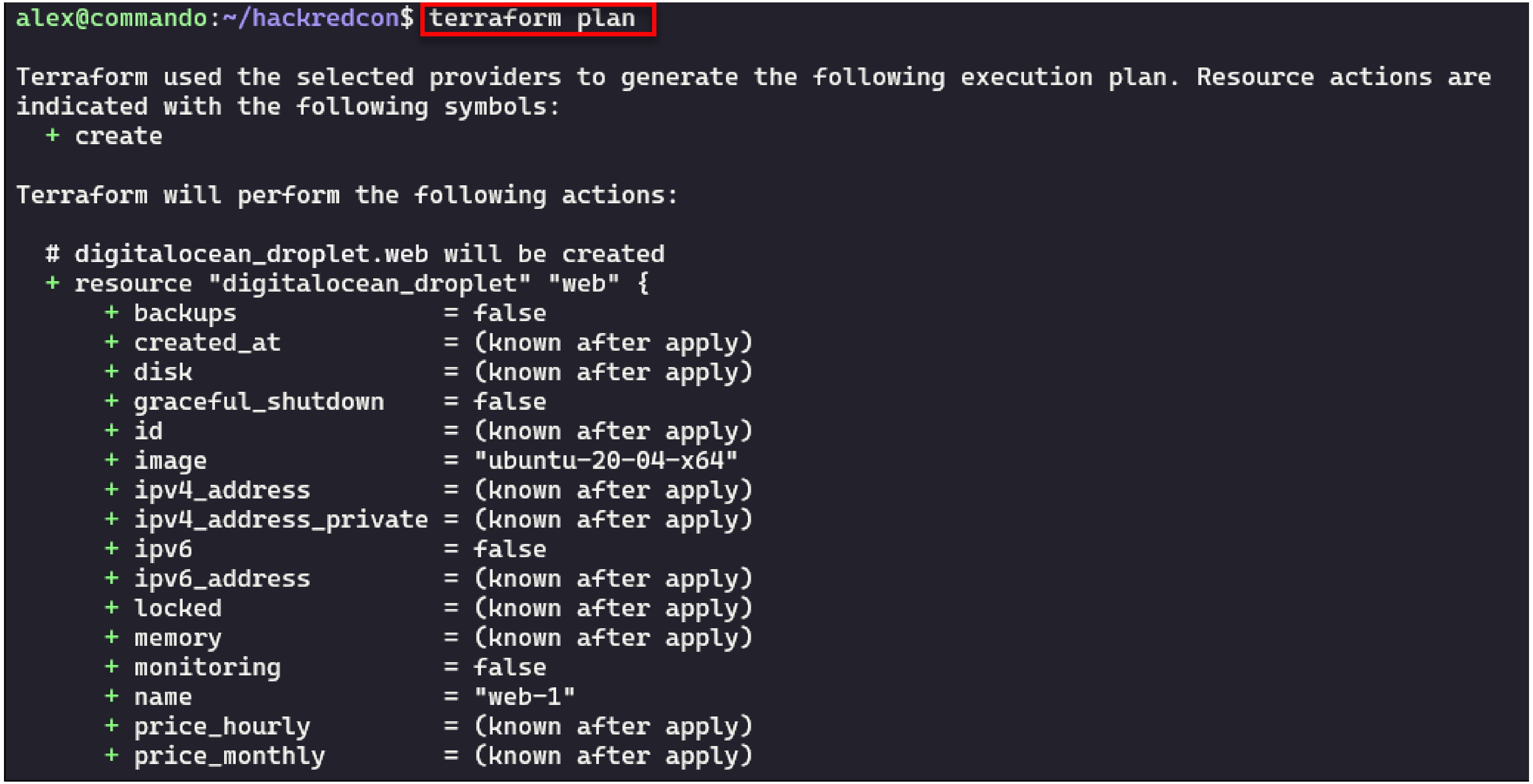
```
# Configure the DigitalOcean Provider
provider "digitalocean" {
  token = var.do_token
}

# Create a new Droplet
resource "digitalocean_droplet" "web" {
  image   = "ubuntu-20-04-x64"
  name    = "web-server"
  region  = "nyc3"
  size    = "s-1vcpu-1gb"
  ssh_keys = [var.ssh_key_id]
}
```

BASIC INIT



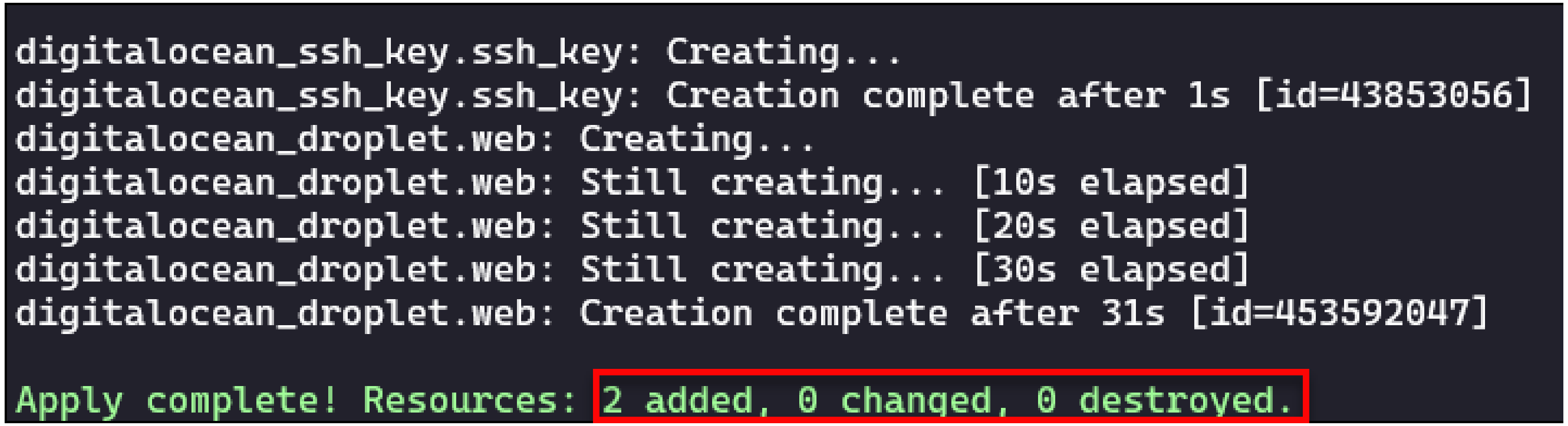
```
Initializing provider plugins...
- Finding digitalocean/digitalocean versions matching "~> 2.0"...
- Installing digitalocean/digitalocean v2.43.0...
- Installed digitalocean/digitalocean v2.43.0 (signed by a HashiCorp partner, key ID F82037E524B9C0E8)

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
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.
```

BASIC PLAN

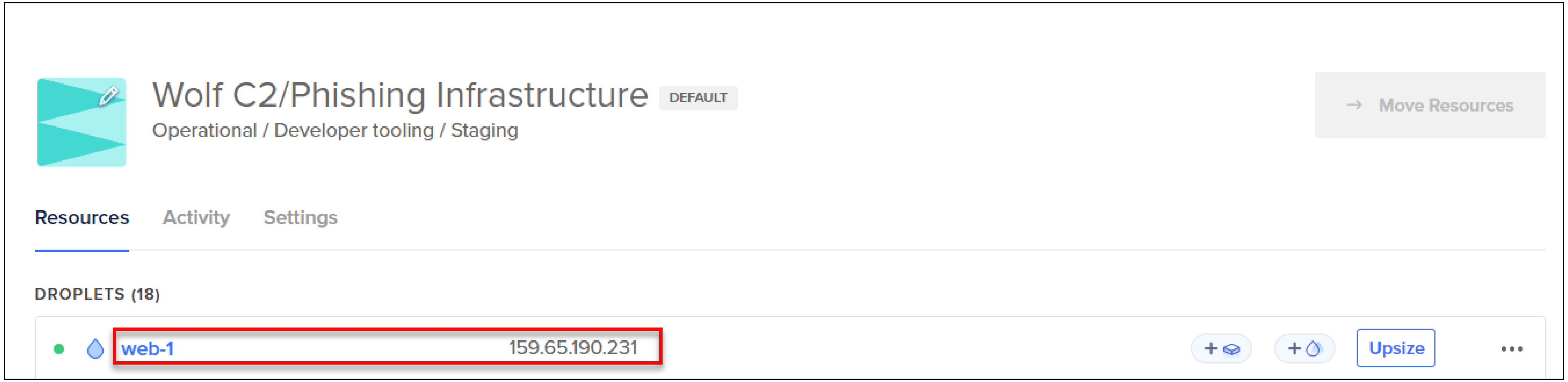


```
alex@commando:~/hackredcon$ terraform plan

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:

  # digitalocean_droplet.web will be created
  + resource "digitalocean_droplet" "web" {
      + backups              = false
      + created_at           = (known after apply)
      + disk                 = (known after apply)
      + graceful_shutdown    = false
      + id                   = (known after apply)
      + image                = "ubuntu-20-04-x64"
      + ipv4_address         = (known after apply)
      + ipv4_address_private = (known after apply)
      + ipv6                 = false
      + ipv6_address         = (known after apply)
      + locked               = (known after apply)
      + memory               = (known after apply)
      + monitoring           = false
      + name                 = "web-1"
      + price_hourly         = (known after apply)
      + price_monthly        = (known after apply)
```

BASIC APPLY



```
digitalocean_ssh_key.ssh_key: Creating...
digitalocean_ssh_key.ssh_key: Creation complete after 1s [id=43853056]
digitalocean_droplet.web: Creating...
digitalocean_droplet.web: Still creating... [10s elapsed]
digitalocean_droplet.web: Still creating... [20s elapsed]
digitalocean_droplet.web: Still creating... [30s elapsed]
digitalocean_droplet.web: Creation complete after 31s [id=453592047]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```



Wolf C2/Phishing Infrastructure DEFAULT

Operational / Developer tooling / Staging

DROPLETS (18)

web-1 159.65.190.231

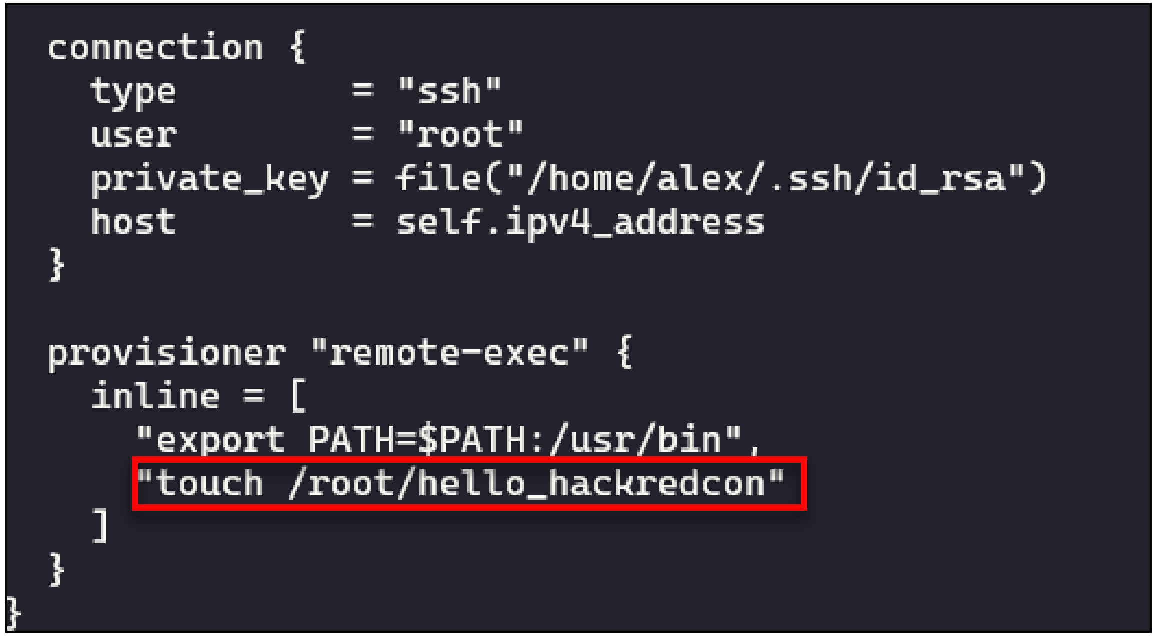
BASIC RECAP

- Made a Terraform working directory with a configuration file
- Initialized and verified our configuration file using terraform
- Deployed the server and successfully accessed it with our key

BUILDING ON BASICS

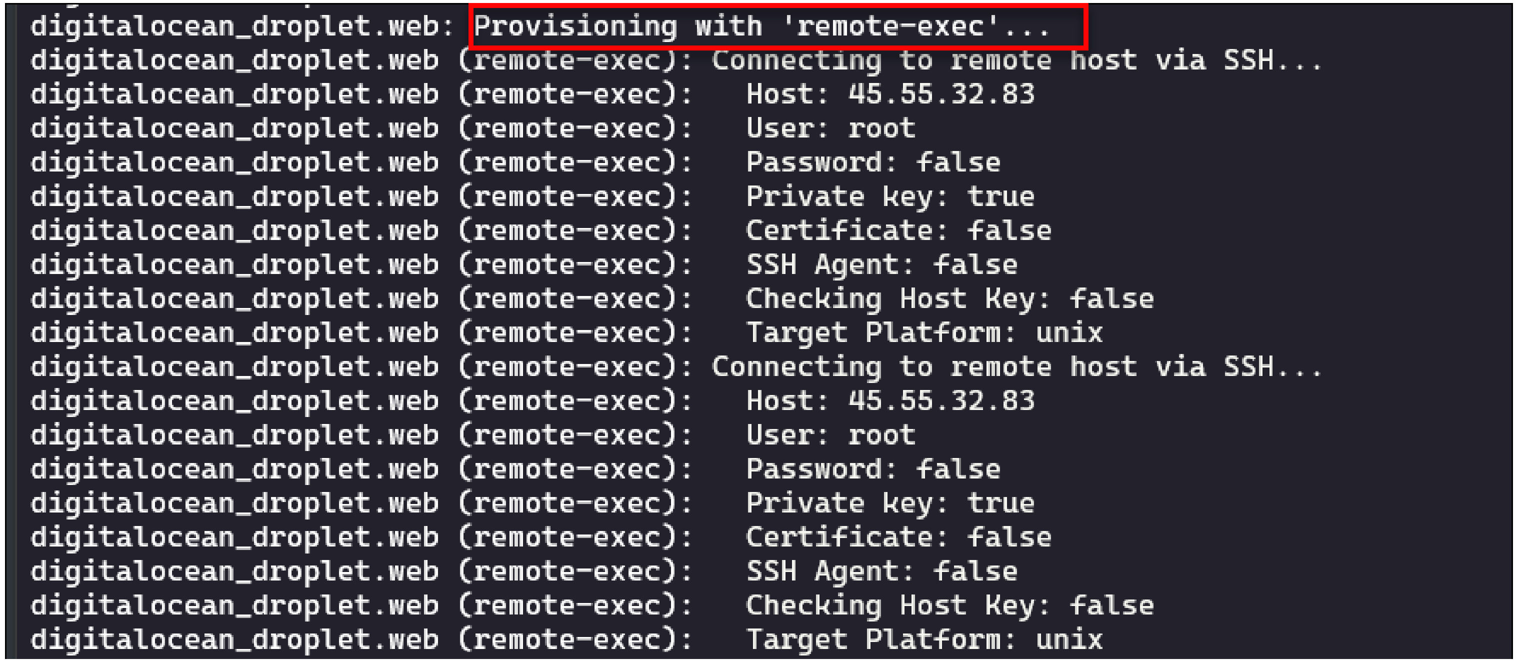
- We can now add provisioners to modify our server
  - Imagine how we can extend this (deploy more than one server, add users, add files, etc.)

- Here is a simple example to add a file to the server we created

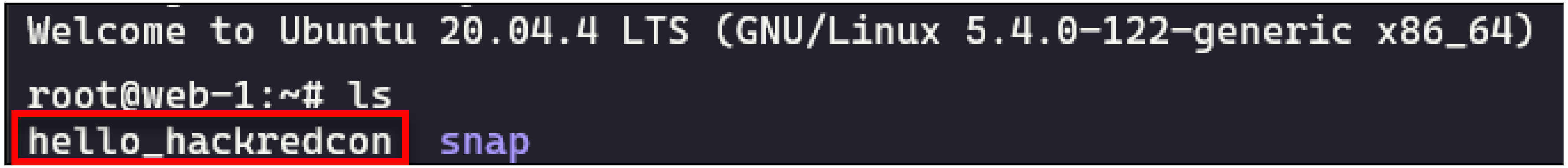


```
  connection {
    type        = "ssh"
    user        = "root"
    private_key = file("/home/alex/.ssh/id_rsa")
    host        = self.ipv4_address
  }

  provisioner "remote-exec" {
    inline = [
      "export PATH=$PATH:/usr/bin",
      "touch /root/hello_hackredcon"
    ]
  }
}
```

TERRAFORM EXECUTION



```
digitalocean_droplet.web: Provisioning with 'remote-exec'...
digitalocean_droplet.web (remote-exec): Connecting to remote host via SSH...
digitalocean_droplet.web (remote-exec):   Host: 45.55.32.83
digitalocean_droplet.web (remote-exec):   User: root
digitalocean_droplet.web (remote-exec):   Password: false
digitalocean_droplet.web (remote-exec):   Private key: true
digitalocean_droplet.web (remote-exec):   Certificate: false
digitalocean_droplet.web (remote-exec):   SSH Agent: false
digitalocean_droplet.web (remote-exec):   Checking Host Key: false
digitalocean_droplet.web (remote-exec):   Target Platform: unix
digitalocean_droplet.web (remote-exec): Connecting to remote host via SSH...
digitalocean_droplet.web (remote-exec):   Host: 45.55.32.83
digitalocean_droplet.web (remote-exec):   User: root
digitalocean_droplet.web (remote-exec):   Password: false
digitalocean_droplet.web (remote-exec):   Private key: true
digitalocean_droplet.web (remote-exec):   Certificate: false
digitalocean_droplet.web (remote-exec):   SSH Agent: false
digitalocean_droplet.web (remote-exec):   Checking Host Key: false
digitalocean_droplet.web (remote-exec):   Target Platform: unix
```

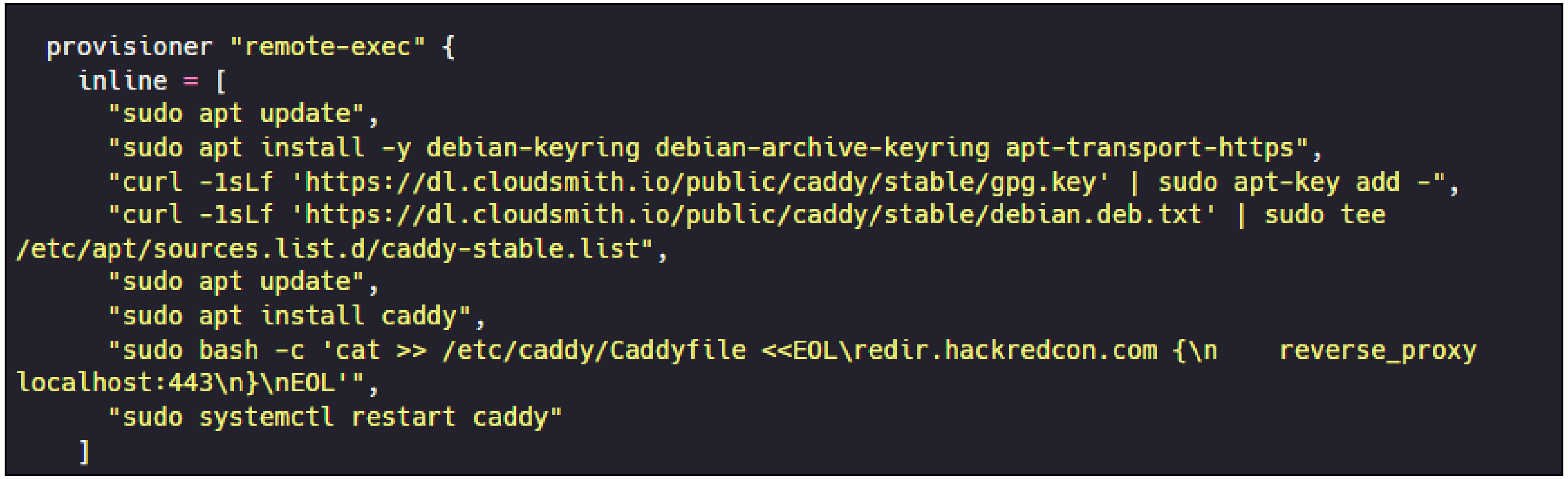


```
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-122-generic x86_64)

root@web-1:~# ls
hello_hackredcon  snap
```

CHEAP REDIRECTOR

- Creating a redirector for a C2 server should be trivial now
  - Regardless of how simple, each step we can automate saves **time**!

- Create a configuration file using Terraform
  - Install Caddy
  - Create a CaddyFile



```
  provisioner "remote-exec" {
    inline = [
      "sudo apt update",
      "sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https",
      "curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo apt-key add -",
      "curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee
/etc/apt/sources.list.d/caddy-stable.list",
      "sudo apt update",
      "sudo apt install caddy",
      "sudo bash -c 'cat >> /etc/caddy/Caddyfile <<EOL\redir.hackredcon.com {\n    reverse_proxy
localhost:443\n}\nEOL'",
      "sudo systemctl restart caddy"
    ]
```

ANSIBLE ROLES

- Configuration management tool to orchestrate our deployments
- Configuration files that extend what Terraform can do for us
  - Install Docker containers (GoPhish/CobaltStrike/Evilginx/etc.)
  - Install packages on the deploy server
- Deploy with Terraform and configure with Ansible

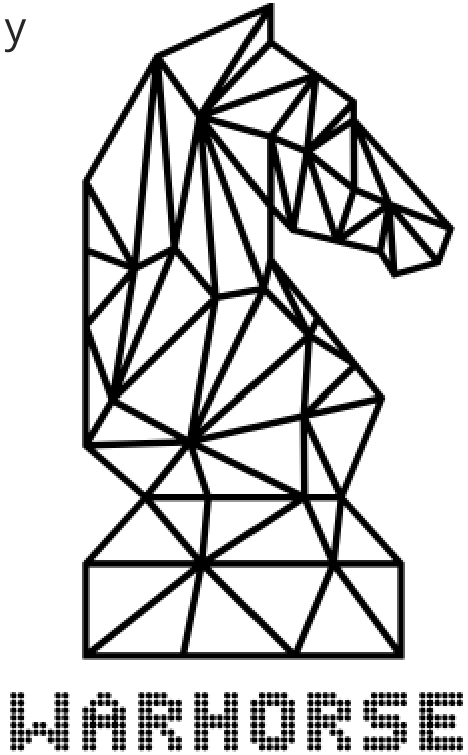
```
ansible-project/
├── ansible.cfg
├── inventory/
│   └── hosts
├── group_vars/
│   └── group1/
│       ├── vars.yml
│       └── vault.yml
├── host_vars/
│   └── hostname1/
│       ├── vars.yml
│       └── vault.yml
├── roles/
│   └── role_name/
│       ├── defaults/
│       │   └── main.yml
│       ├── files/
│       ├── handlers/
│       │   └── main.yml
│       ├── meta/
│       │   └── main.yml
│       ├── tasks/
│       │   └── main.yml
│       ├── templates/
│       ├── tests/
│       │   ├── inventory
│       │   └── test.yml
│       └── vars/
│           └── main.yml
├── playbooks/
│   ├── playbook1.yml
│   └── playbook2.yml
└── README.md
```

WARHORSE

- Warhorse is a wrapper for Terraform/Ansible that can generate files to deploy
  - Built for Offensive Security Infrastructure Automation
  - Developed by Ralph May (Black Hills Information Security)
  - Training in HackerOps Course

- Why reinvent the wheel / create another tool that does the same thing?

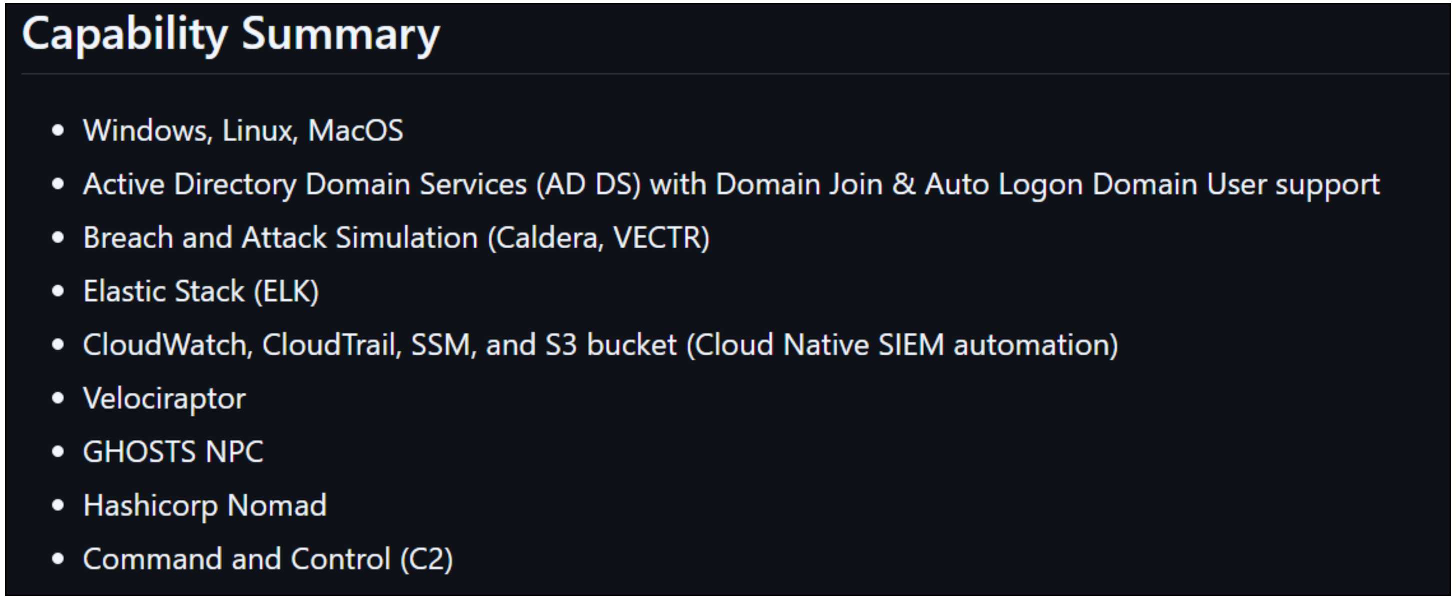
- Many Ansible roles can be viewed here:
  - https://github.com/geerlingguy (Jeff Geerling)



EXAMPLES GENERATORS

- Jason Ostrom creates a wrapper to easily deploy labs in Terraform
  - https://www.purplecloud.network/

- Make vulnerable labs to test attacker techniques/payloads/etc.



Capability Summary

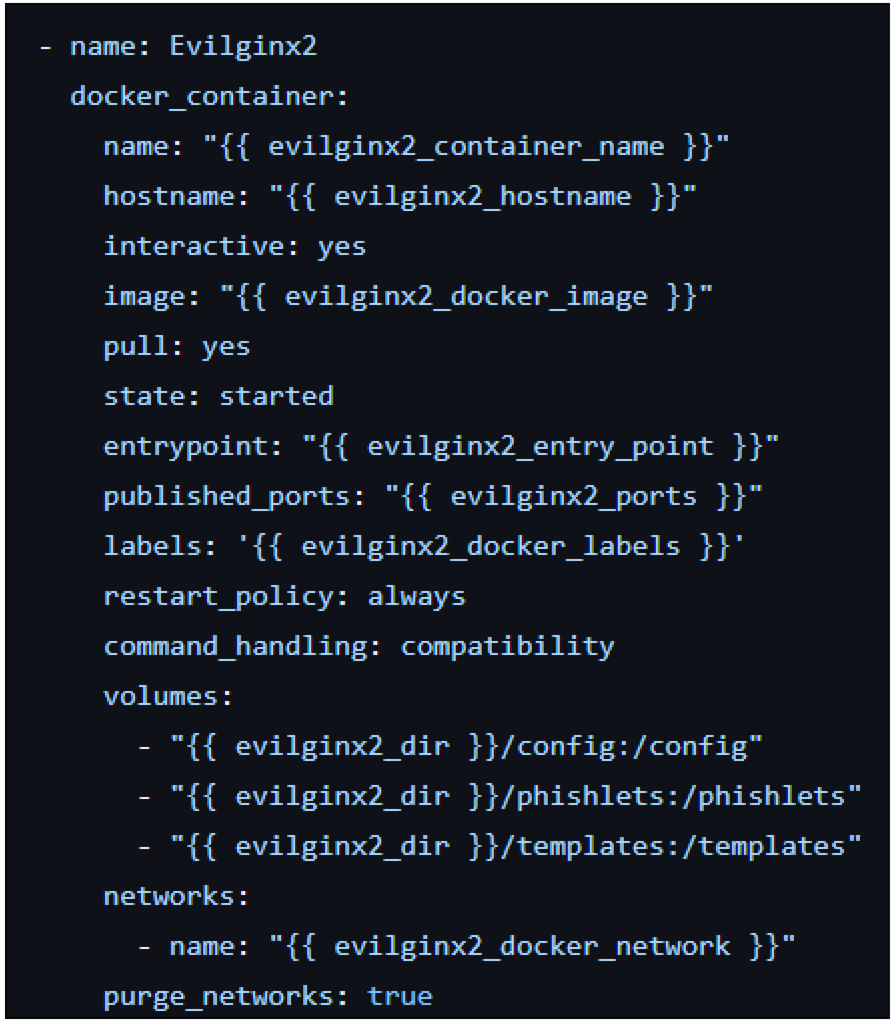
- Windows, Linux, MacOS
- Active Directory Domain Services (AD DS) with Domain Join & Auto Logon Domain User support
- Breach and Attack Simulation (Caldera, VECTR)
- Elastic Stack (ELK)
- CloudWatch, CloudTrail, SSM, and S3 bucket (Cloud Native SIEM automation)
- Velociraptor
- GHOSTS NPC
- Hashicorp Nomad
- Command and Control (C2)

EVILGINX ROLE

- Warhorse uses Ansible Roles and Docker Images

- Jinja code can be used to template configuration files and phishlets/C2 profiles

- Operators must monitor code changes and modify as needed
  - Evilginx/Mythic/CobaltStrike all change!



```
- name: Evilginx2
  docker_container:
    name: "{{ evilginx2_container_name }}"
    hostname: "{{ evilginx2_hostname }}"
    interactive: yes
    image: "{{ evilginx2_docker_image }}"
    pull: yes
    state: started
    entrypoint: "{{ evilginx2_entry_point }}"
    published_ports: "{{ evilginx2_ports }}"
    labels: '{{ evilginx2_docker_labels }}'
    restart_policy: always
    command_handling: compatibility
    volumes:
      - "{{ evilginx2_dir }}/config:/config"
      - "{{ evilginx2_dir }}/phishlets:/phishlets"
      - "{{ evilginx2_dir }}/templates:/templates"
    networks:
      - name: "{{ evilginx2_docker_network }}"
    purge_networks: true
```

https://github.com/warhorse/ansible-role-evilginx2-docker/blob/master/templates/phishlets/o365.yaml.j2

EVILGINX ROLE

- Docker image maintained in
- Jinja code can be used to template configuration files and phishlets
- Operators must monitor code changes and modify them as needed
- Docker “ghcr.io” is used to manage these images and can be modified (CI/CD)
- Remove IoC’s, expose ports, choose versions you need!

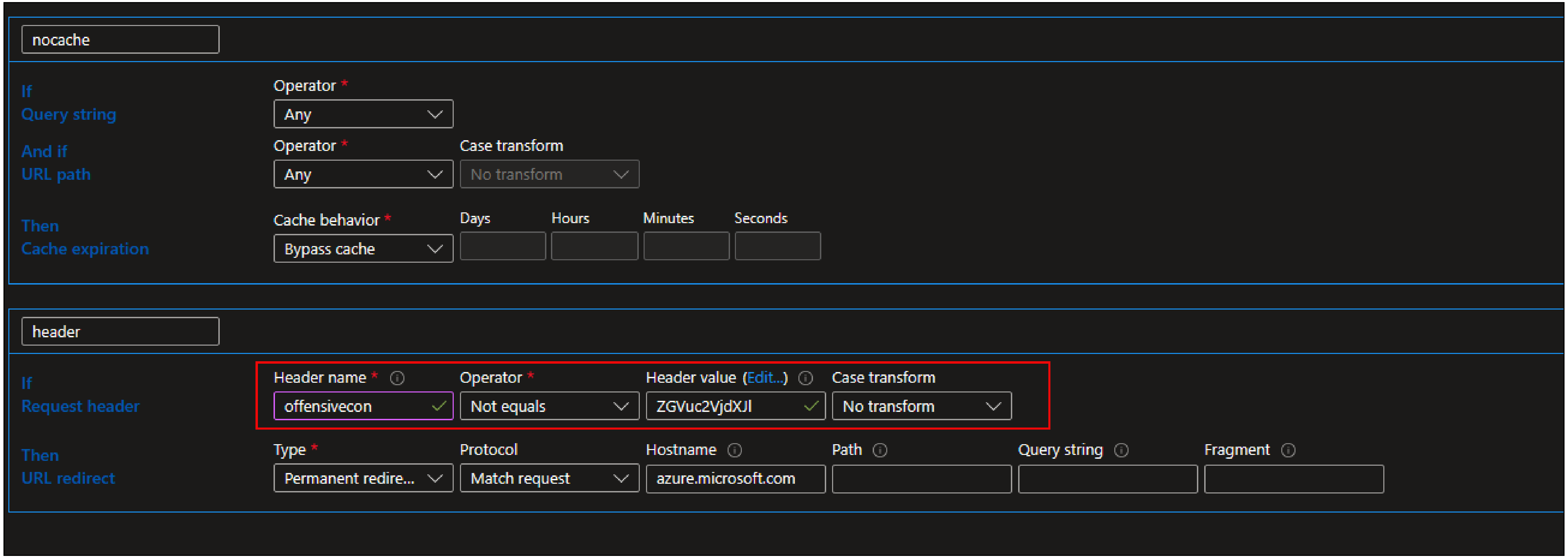
https://github.com/almart/docker-evilginx2/pkgs/container/docker-evilginx2

CDN Abuse

CLOUD CDN

- Already trusted by cloud providers Azure/AWS/etc.
  - azureedge.net
  - cloudfront.net
- Will probably bypass standard web filtering rules
- Use to facilitate social engineering attacks (Evilginx/Teams/Direct Send)
- Use as a redirector for Command and Control!

CDN REDIRECTOR



nocache

If
Query string

Operator * Any

And if
URL path

Operator * Any

Case transform: No transform

Then
Cache expiration

Cache behavior * Bypass cache

Days | Hours | Minutes | Seconds

header

If
Request header

Header name * offensivecon

Operator * Not equals

Header value (Edit...) ZGVuc2VjdXJl

Case transform: No transform

Then
URL redirect

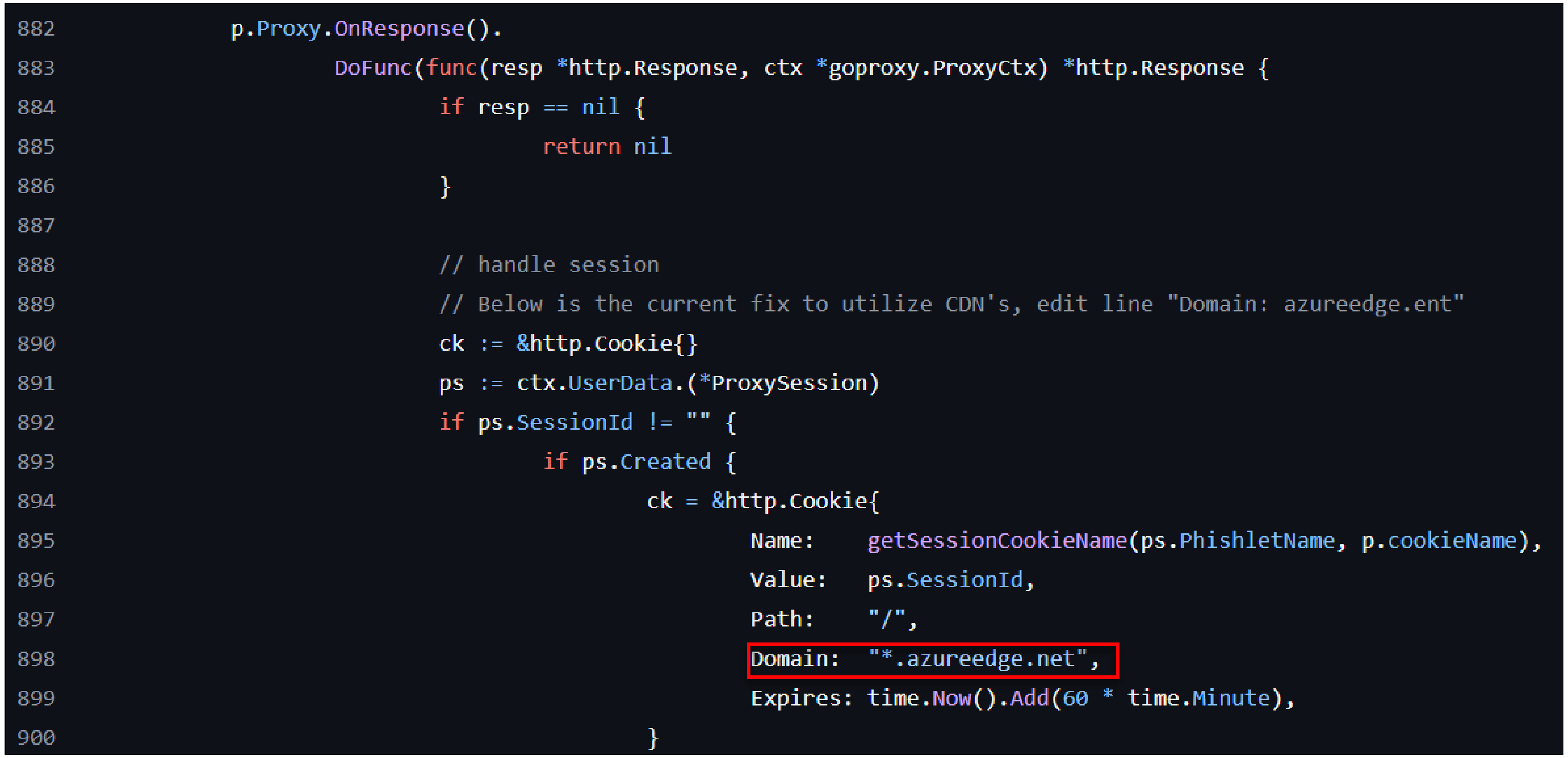
Type * Permanent redire...

Protocol: Match request

Hostname: azure.microsoft.com

Path | Query string | Fragment

CDN FIX EVILGINX2

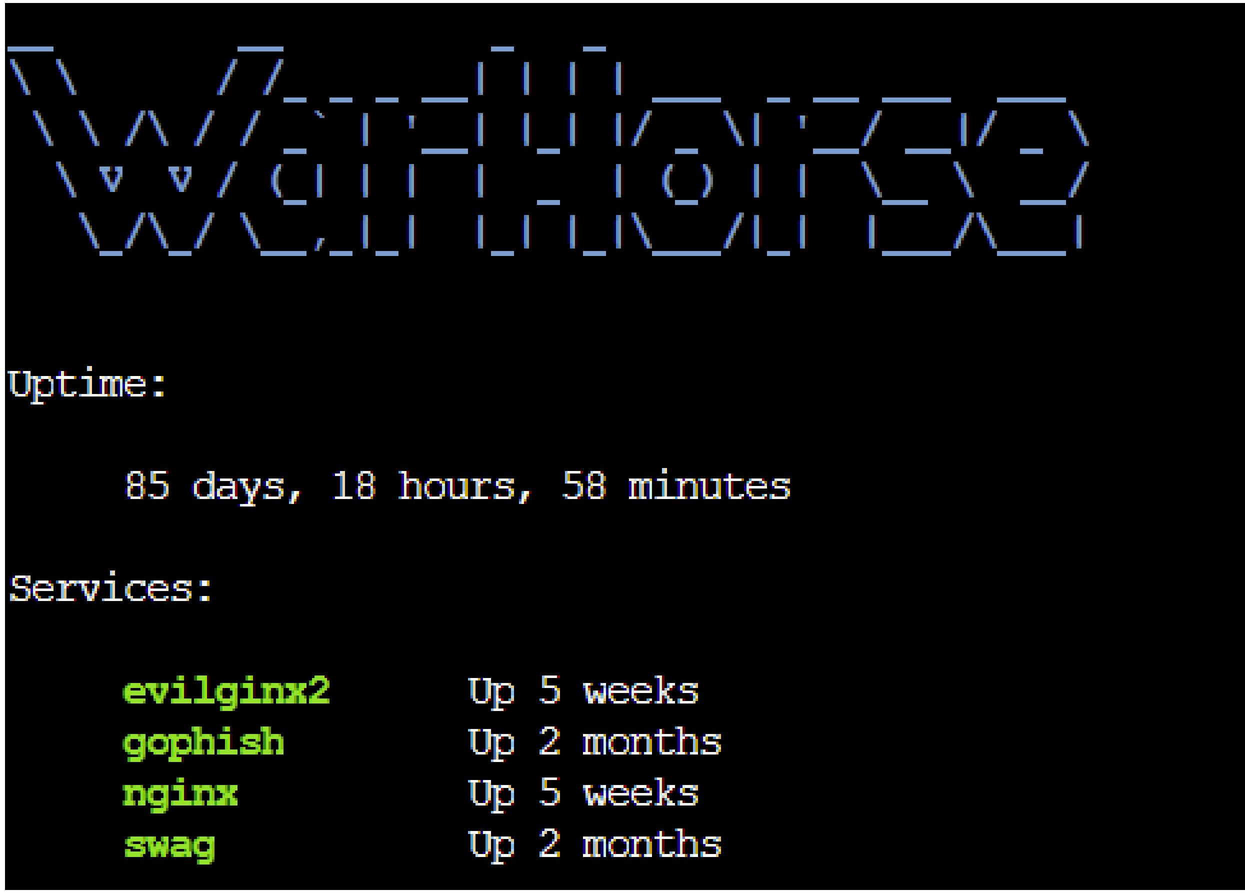


```
        p.Proxy.OnResponse().
            DoFunc(func(resp *http.Response, ctx *goproxy.ProxyCtx) *http.Response {
                if resp == nil {
                    return nil
                }

                // handle session
                // Below is the current fix to utilize CDN's, edit line "Domain: azureedge.ent"
                ck := &http.Cookie{}
                ps := ctx.UserData.(*ProxySession)
                if ps.SessionId != "" {
                    if ps.Created {
                        ck = &http.Cookie{
                            Name:    getSessionCookieName(ps.PhishletName, p.cookieName),
                            Value:   ps.SessionId,
                            Path:    "/",
                            Domain:  "*.azureedge.net",
                            Expires: time.Now().Add(60 * time.Minute),
                        }
```

/core/http_proxy.go

WARHORSE CONTEXT

- WarHorse confirms deployments
- Templated configurations
- Easy to test and rebuild

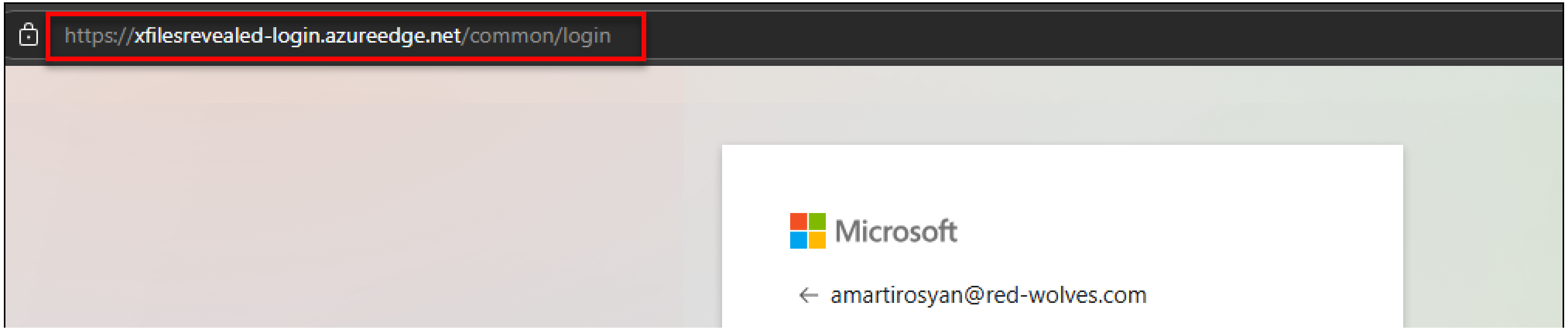


```
__        __          _   _
\ \      / /_ _ _ __ | | | | ___  _ __ ___  ___
 \ \ /\ / / _` | '__|| |_| |/ _ \| '__/ __|/ _ \
  \ V  V / (_| | |   |  _  | (_) | |  \__ \  __/
   \_/\_/ \__,_|_|   |_| |_|\___/|_|  |___/\___|


Uptime:

     85 days, 18 hours, 58 minutes

Services:

     evilginx2       Up 5 weeks
     gophish         Up 2 months
     nginx           Up 5 weeks
     swag            Up 2 months
```

USING CDN's for EVILGINX

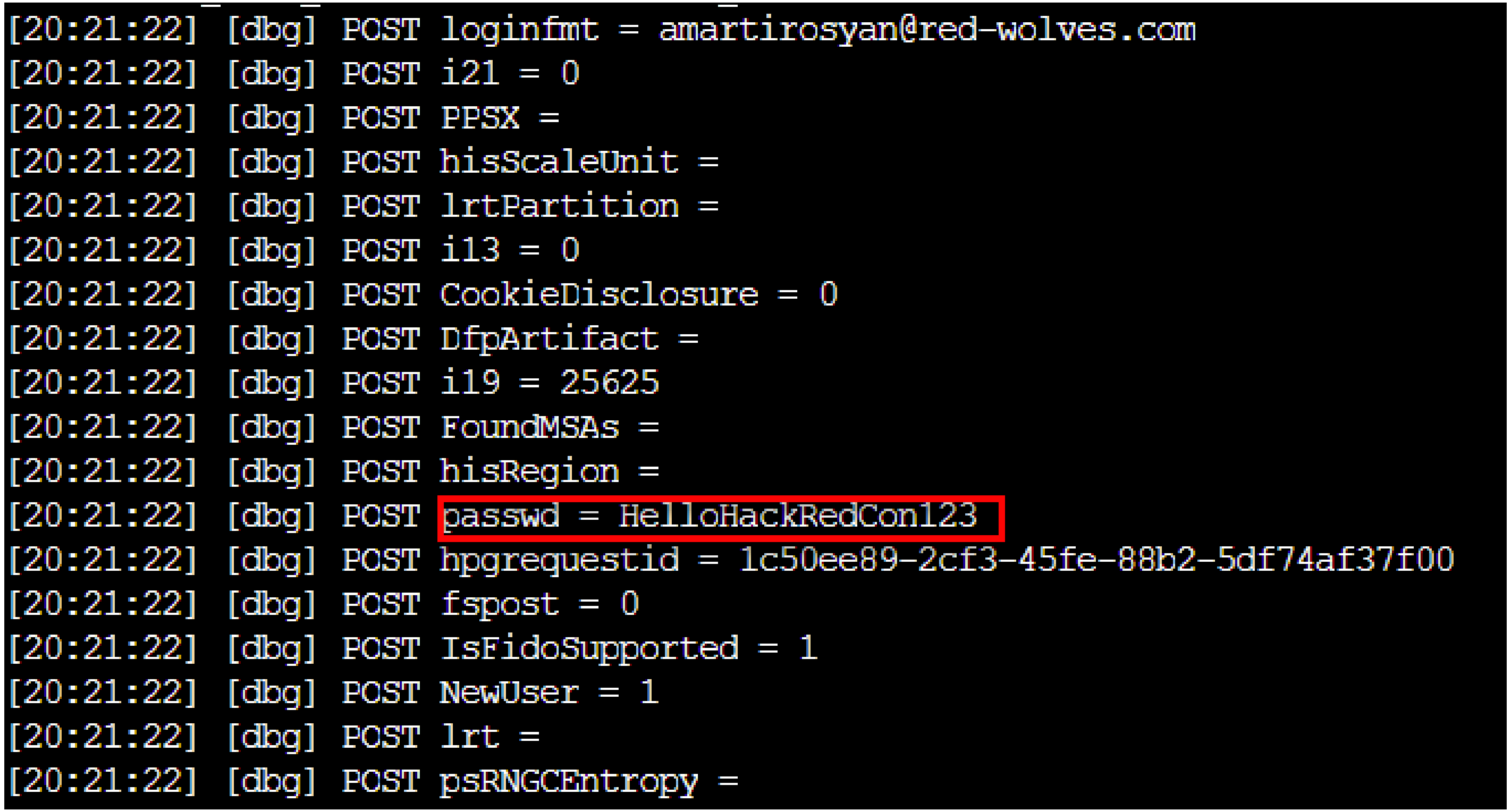
- Nginx reverse proxy handles connections based on host headers

- Now we have a trusted certificate and a way to evade defenses



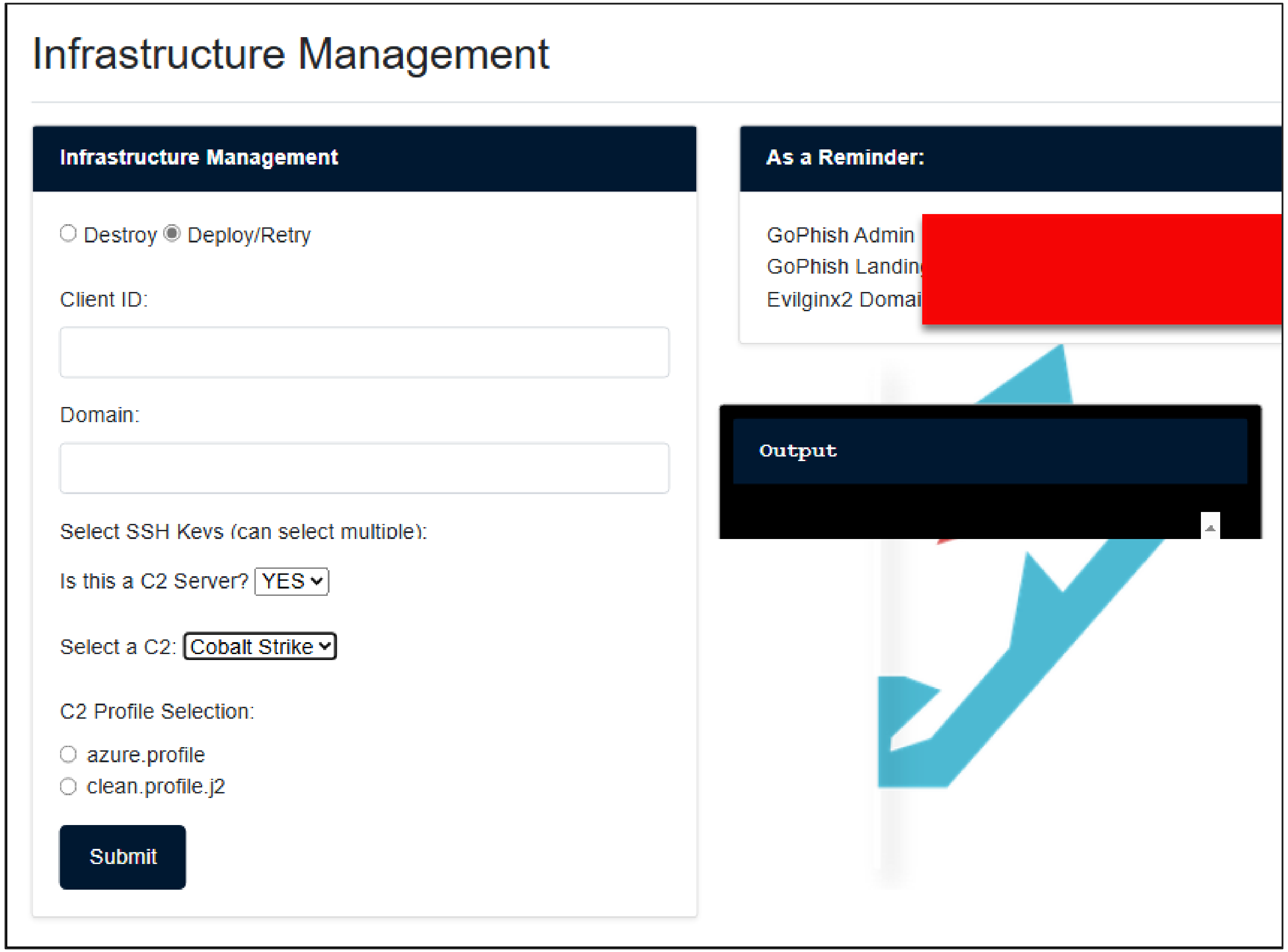
https://xfilesrevealed-login.azureedge.net/common/login

CAPTURE COOKIES/PASSWORDS



```
[20:21:22] [dbg] POST loginfmt = amartirosyan@red-wolves.com
[20:21:22] [dbg] POST i21 = 0
[20:21:22] [dbg] POST PPSX =
[20:21:22] [dbg] POST hisScaleUnit =
[20:21:22] [dbg] POST lrtPartition =
[20:21:22] [dbg] POST i13 = 0
[20:21:22] [dbg] POST CookieDisclosure = 0
[20:21:22] [dbg] POST DfpArtifact =
[20:21:22] [dbg] POST i19 = 25625
[20:21:22] [dbg] POST FoundMSAs =
[20:21:22] [dbg] POST hisRegion =
[20:21:22] [dbg] POST passwd = HelloHackRedCon123
[20:21:22] [dbg] POST hpgrequestid = 1c50ee89-2cf3-45fe-88b2-5df74af37f00
[20:21:22] [dbg] POST fspost = 0
[20:21:22] [dbg] POST IsFidoSupported = 1
[20:21:22] [dbg] POST NewUser = 1
[20:21:22] [dbg] POST lrt =
[20:21:22] [dbg] POST psRNGCEntropy =
```

INFRA SERVER



Infrastructure Management

Infrastructure Management

○ Destroy ● Deploy/Retry

Client ID:

Domain:

Select SSH Keys (can select multiple):

Is this a C2 Server? YES

Select a C2: Cobalt Strike

C2 Profile Selection:

○ azure.profile
○ clean.profile.j2

As a Reminder:

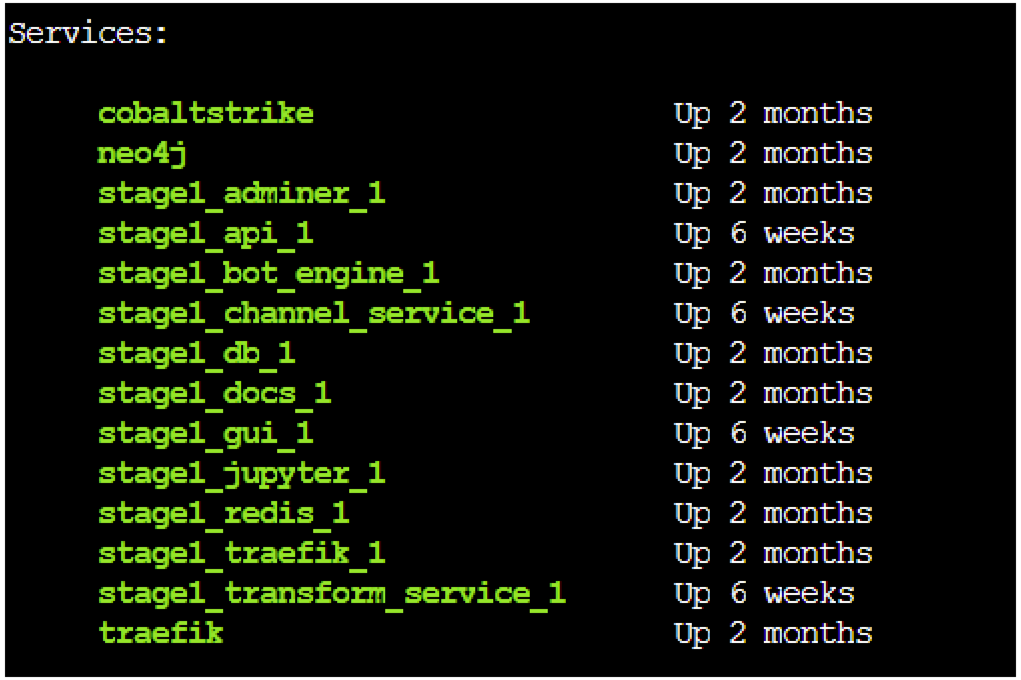
GoPhish Admin
GoPhish Landin
Evilginx2 Domai
[redacted]

C2 EXAMPLE

- HTTPS listeners by CDN (Azure/AWS)
  - Deploy Mythic/CobaltStrike/Others?

- Simple fixes again for Docker/new updates

- Link with redirectors, what else can be done?



```
Services:

     cobaltstrike                           Up 2 months
     neo4j                                  Up 2 months
     stage1_adminer_1                       Up 2 months
     stage1_api_1                           Up 6 weeks
     stage1_bot_engine_1                    Up 2 months
     stage1_channel_service_1               Up 6 weeks
     stage1_db_1                            Up 2 months
     stage1_docs_1                          Up 2 months
     stage1_gui_1                           Up 6 weeks
     stage1_jupyter_1                       Up 2 months
     stage1_redis_1                         Up 2 months
     stage1_traefik_1                       Up 2 months
     stage1_transform_service_1             Up 6 weeks
     traefik                                Up 2 months
```

WHAT WORKS FOR YOU

- Many existing open source tools to help automate infrastructure
- Customizing is still required
- Start simple and look for tasks that are well understood by all operators!

BACKDOORS & BREACHES

https://spearphish-general-store.myshopify.com/collections/backdoors-breaches-incident-response-card-game

QUESTIONS

Alex Martirosyan,
CRTO , OSCP, GPEN
Lead Penetration Tester, DenSecure
AMartirosyan@wolfandco.com
617.261.8138
https://www.linkedin.com/in/alex-martirosyan/
https://twitter.com/almartiros
https://www.wolfandco.com/services/densecure/

ABOUT WOLF & COMPANY, P.C.

1911 WOLF & CO. ESTABLISHED

300+ PROFESSIONALS

3 OFFICES IN:

- ✓ Boston, MA
- ✓ Springfield, MA
- ✓ Livingston, NJ

SERVICES OFFERED IN:

- ✓ Audit
- ✓ Tax
- ✓ Risk Management

ABOUT WOLF & COMPANY, P.C.

111

YEARS IN BUSINESS

- ✓ Established in 1911
- ✓ Built on quality and integrity
- ✓ Succession strategy to remain independent allows us to be with you throughout your business lifecycle

300+

EXPERIENCED, HIGHLY TRAINED PROFESSIONALS

- ✓ Lower-than-industry-average staff turnover means a consistent team structure year after year
- ✓ Niche team dedicated to your industry

RESOURCES TO LEARN MORE

- ✓ Cultures & Values
- ✓ Inclusion & Diversity
- ✓ Our History
- ✓ Social Responsibility
- ✓ Thought Leadership
- ✓ Wolf Global

Wolf & Company ranked

#2 BEST LARGE FIRM TO WORK FOR

nationwide

accountingTODAY

ABOUT WOLF & COMPANY, P.C.

SERVICES WE OFFER

We combine industry expertise with service specialization to provide your organization with insight, opportunities, and solutions allowing you to address your unique business needs.

ADVISORY

- Business Continuity Planning
- Cybersecurity
- Enterprise Risk Management
- Environment, Social & Governance
- Internal Audit
- IT Audit
- Model Risk Management
- Outsourced Accounting Solutions
- Penetration Testing
- Regulatory Compliance
- Strategic Planning

ASSURANCE

- Employee Benefit Plan Audits
- Financial Statements Audits
- HITRUST
- PCI DSS
- SOC Reporting

TAX

- Business Tax
- Federal
- International
- State & Local
- Private Client Group

vSUITE

- Virtual Consulting Services
  - Business Continuity Planning (BCP)
  - Virtual Chief Information Security Officer (vCISO)
  - Virtual Chief Privacy Officer (vCPO)
  - Virtual Chief Risk Officer (vCRO)
  - Virtual Vendor Management

WOLFPAC

- Integrated risk management SaaS suite

WOLF & COMPANY, P.C.

den secure by wolf & company, p.c.

WOLF ACCOLADES

Wolf is pleased to have received recognition from a variety of sources for our efforts at providing responsive client service and development of our professionals. Examples of this recognition include:

INSIDE Public Accounting

TOP 100
Accounting Firms

accountingTODAY

TOP 100
Accounting Firms

#2 BEST LARGE FIRM to
Work For Nationwide

TOP FIRMS:
New England

BOSTON BUSINESS JOURNAL

- ✓ Area's Best Places to Work
- ✓ Area's Most Admired Companies
- ✓ Area's Fastest Growing Private Companies
- ✓ Area's Largest I.T. Consulting Firms

Forbes

America's Best
Tax and Accounting
Firms of 2023, 2021

ABOUT DENSECURE

Wolf & Company's IT Assurance & Advisory team of cybersecurity experts, DenSecure™, brings together extensive technical knowledge and industry experience with internationally-recognized frameworks to develop strong cybersecurity programs.

DenSecure's core services include:

- Red Team Assessment
- Application Penetration Testing
- Network Penetration Testing
- Social Engineering
- Threat Emulation
- Assumed Breach Testing

Back to Home →