Purple Teaming Cloud Identity: Simulation Labs for Red and Blue teams

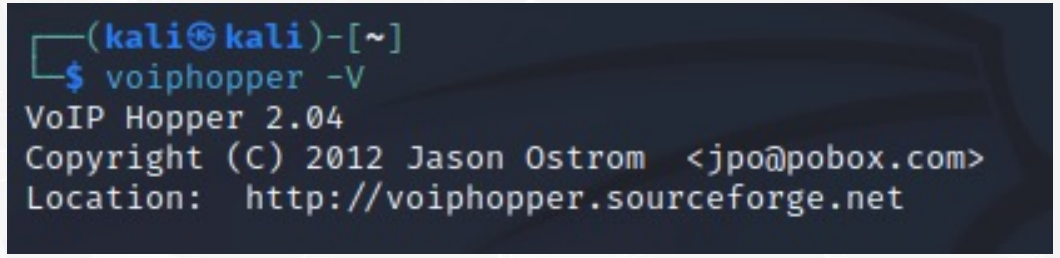
SANS Pen Test Hackfest 2022

Jason Ostrom

November 14, 2022

Hey What's up, I'm Jason

- Builder of things @SANS Institute
- Certified Instructor @SANS, SEC588 → "Cloud Penetration Testing"
- Founder, Principal @Stora Security
- Community tool author
- Family, hockey, football



New tool! Cloud "edge" bug bounty and recon tool
github.com/iknowjason/edge

Agenda for today

- Purple Teaming
- Infrastructure as Code (IaC)
- Labs Overview
- BlueCloud & PurpleCloud
- Demo

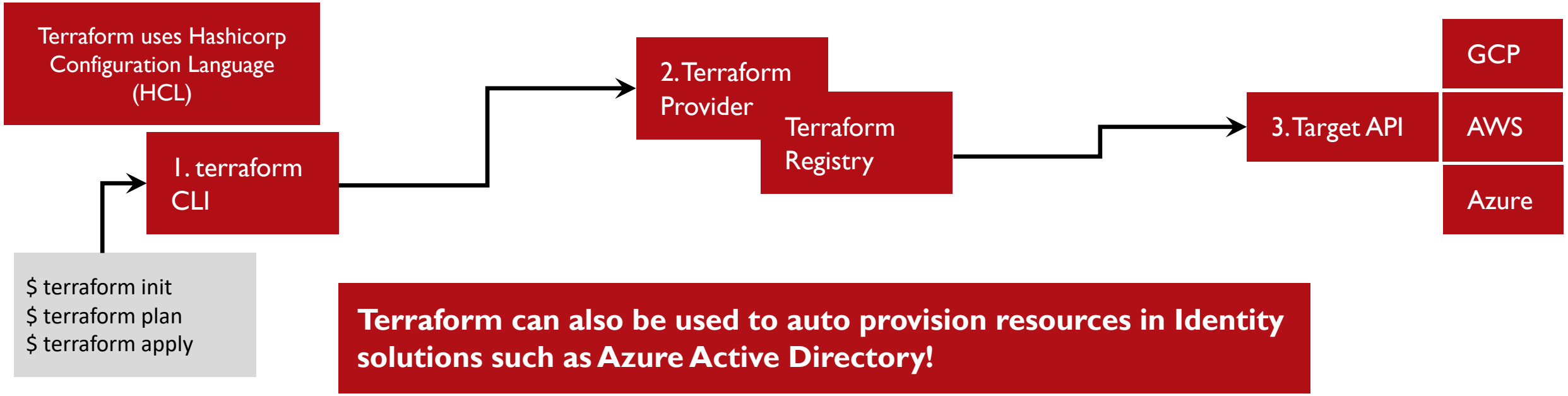
Infrastructure as Code (IaC)

Infrastructure as Code (IaC)

- ***Infrastructure as Code*** is the managing and provisioning of infrastructure automated through code instead of manually.
- Configuration is code stored in a VCS (Github, Gitlab)
- Declarative, tracking desired state
- Benefits: Speed, consistency, repeatability, lower cost
- Each cloud provider has their own IaC service
- Two popular multi-cloud platform tools: **Pulumi** & **Terraform**

Terraform Overview

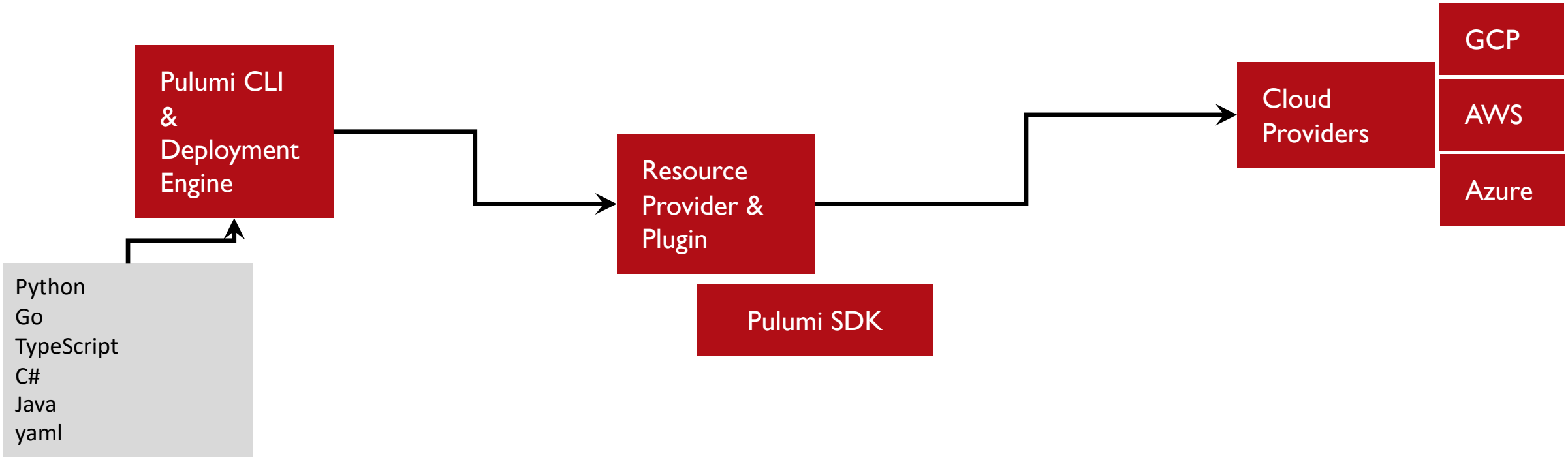
- **Terraform** is a free, universal, and popular IaC tool that can manage infrastructure with declarative files
  - Build, change, and version infrastructure in AWS, GCP, Azure
  - Providers: plugins that talk to API for different cloud providers and can provision Infrastructure, DNS, SaaS services, Kubernetes



Terraform can also be used to auto provision resources in Identity solutions such as Azure Active Directory!

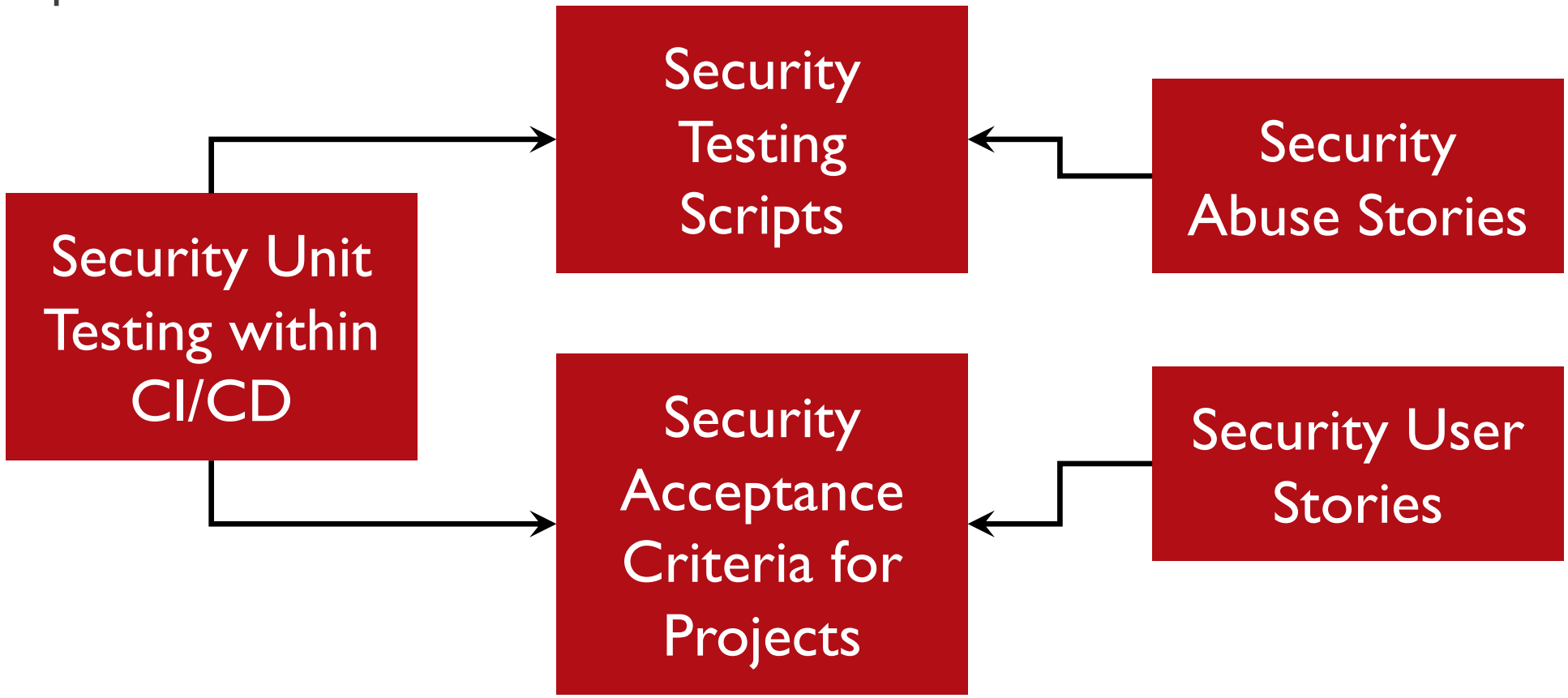
Pulumi Overview

- **Pulumi** is a free and universal IaC tool that leverages existing programming languages to interact with cloud resources through the Pulumi SDK.
  - Use Python, Go, JavaScript to to build, change, and version infrastructure in AWS, GCP, Azure
  - Also uses the desired state model like Terraform



Security as Code (SaC)

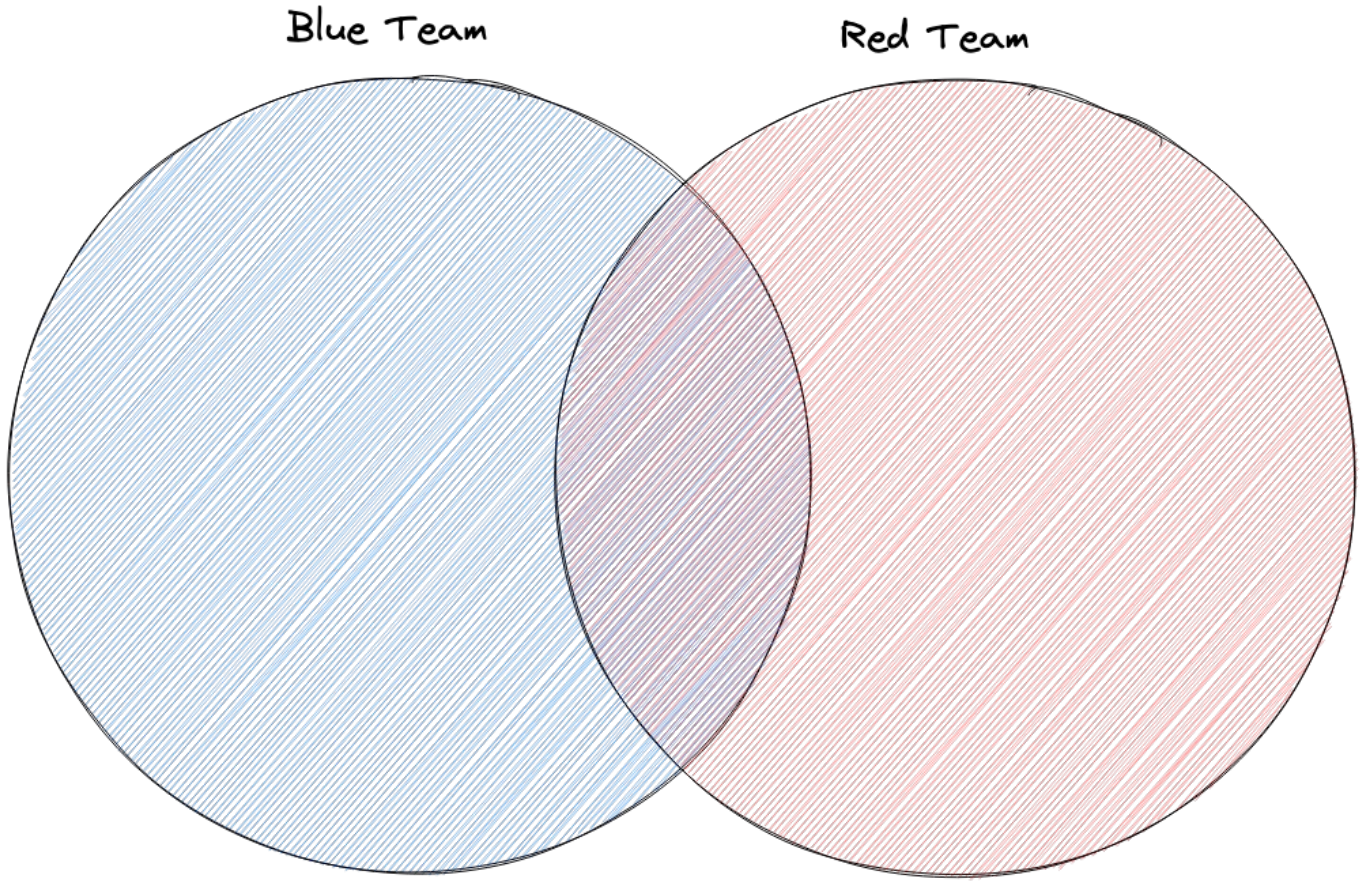
- Using code within a DevOps SDLC environment to help meet security requirements.



Purple Teaming

Purple Teaming Overview (1)

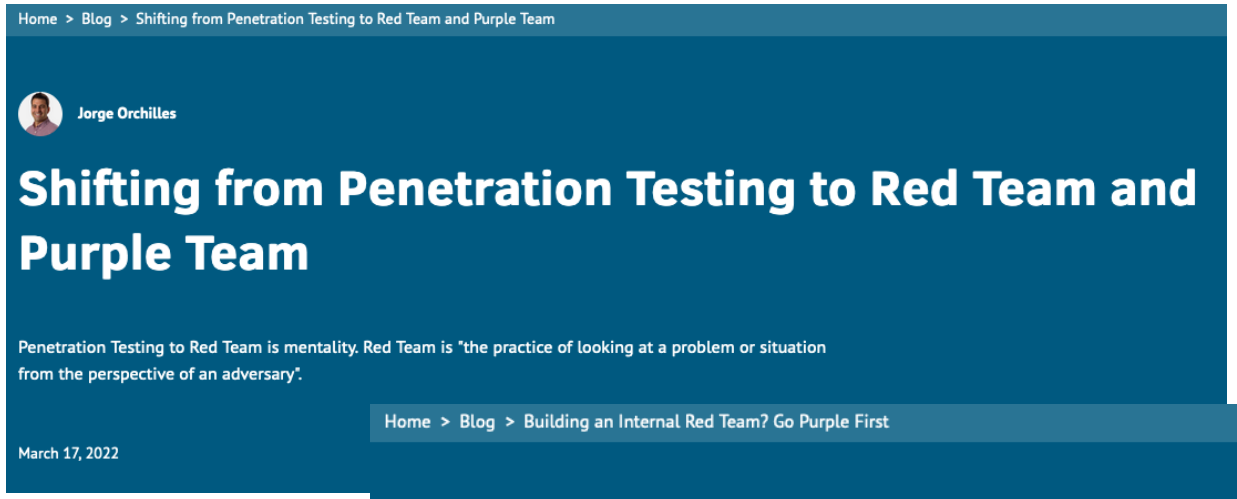
- ***Purple Teaming:*** Red and Blue teams collaborating as a team to make defense better

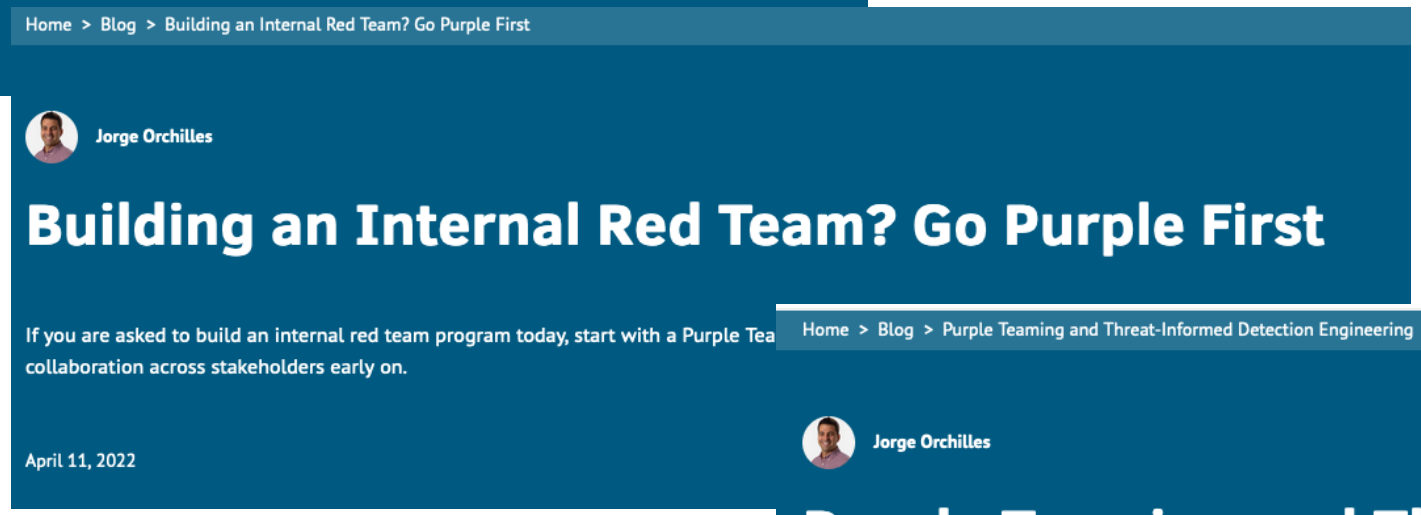


Purple Teaming Overview (2)

- Adversary Emulations run, from collaboration comes improvements:
- Adding better log sources
- Log Enrichment
- Improve Detection Engineering
- Process improvements
- Training for blue team

Purple Teaming Overview (3)





https://www.sans.org/blog/shifting-from-penetration-testing-to-red-team-and-purple-team/

https://www.sans.org/blog/building-internal-red-team-go-purple-first/

https://www.sans.org/blog/purple-teaming-threat-informed-detection-engineering/

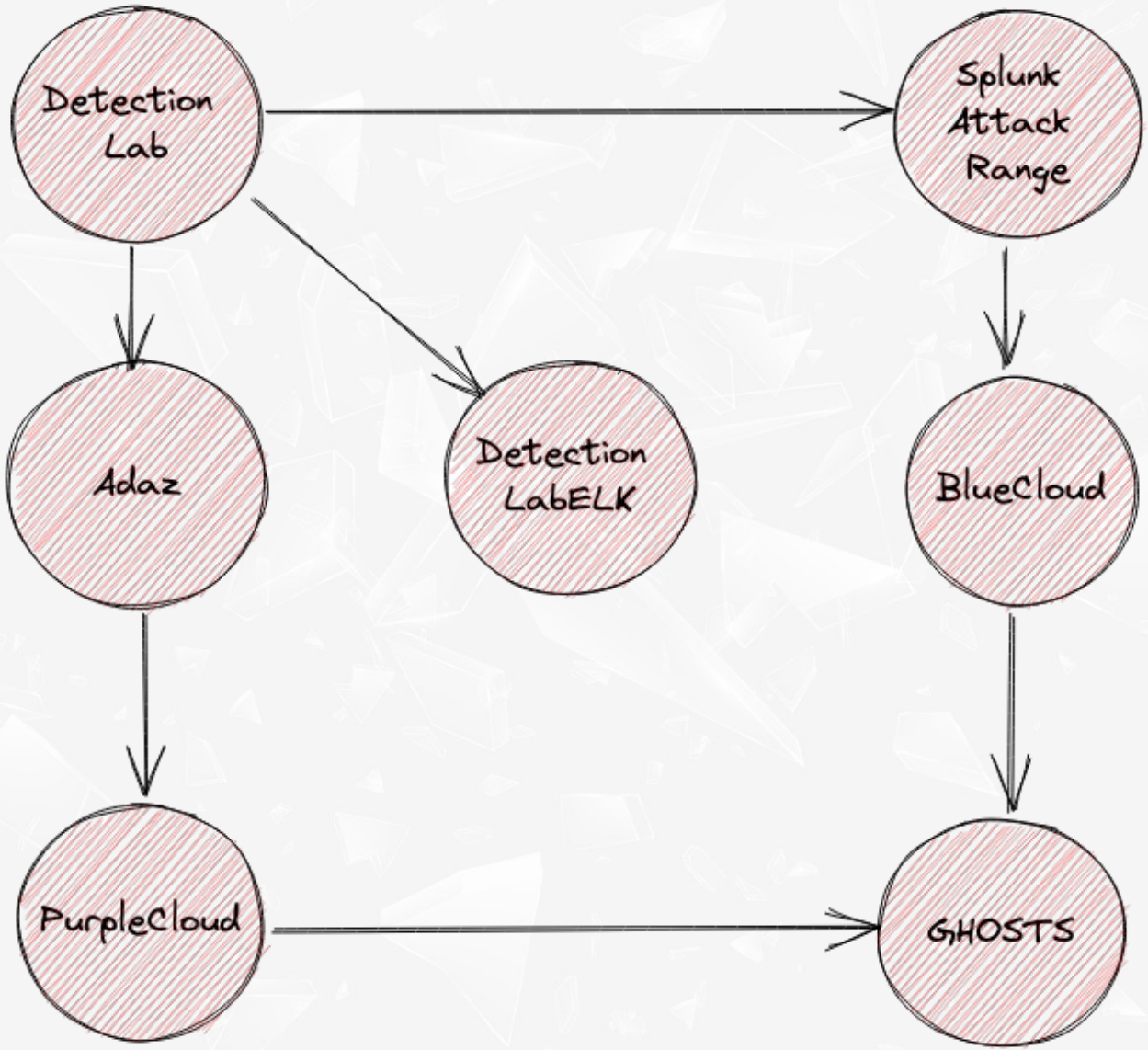
Labs Overview

Why Simulation Labs?

- Learning a new technology
- Detection Engineering labs for properly instrumenting logs to detect attacks
- ”***Detection as Code***” as an ideal state
- Training Red and Blue teams, experimenting with a specific tool or technique against a simulated EDR endpoint or AD environment
- **Purple Teaming** exercises
- R&D security research
- Bug bounty
- Malware lab
- Fun!

Purple Teaming: To test and improve people, process, and technology. In Cyber Ranges, the focus is on training people and improving technology. The Detection Engineering process is running emulations in the Simulation Lab to improve logging and technology. These changes eventually go into production.

Overview of Labs (1)



Overview of Labs (2)

- DetectionLab
  - https://www.detectionlab.network/
  - Splunk, Velociraptor, OSQuery with Fleet, Sysmon, Logging best practices with WEF/WEC
- Splunk Attack Range
  - https://github.com/splunk/attack_range
  - Splunk, Sysmon, Remote attack simulations using Ansible + Python + ART / Prelude
- Adaz
  - https://github.com/christophetd/Adaz
  - Azure, ELK, Multi-Host AD with Domain Join, WEF, Audit policies

GHOSTS: User Simulation Framework (1)

- A user simulation framework for complex, realistic NPC orchestration
- **NPC:** Non-player characters: Realism of users and their behavior running applications on an enterprise network
- Test skills and train network defenders with real NPC players operating on the network creating static and background noise

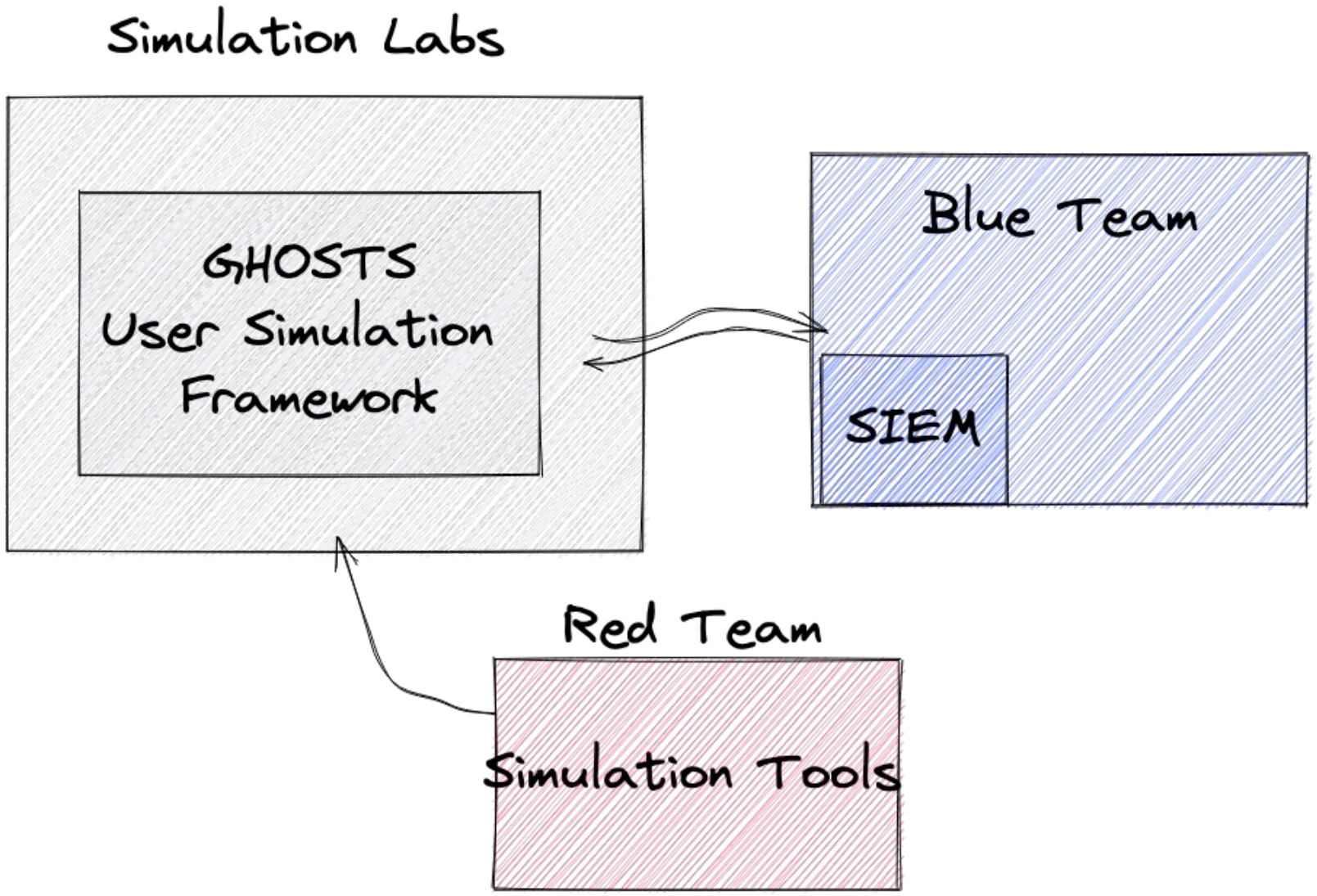
Image Source: https://github.com/cmu-sei/GHOSTS

GHOSTS: User Simulation Framework (2)

- Grafana Dashboards and API
- GHOSTS Windows client binary runs user behavior based on JSON files
- Created by Carnegie Mellon University Software Engineering Institute: https://github.com/cmu-sei/GHOSTS
- GHOSTS is a great tool for enhancing any Simulation Lab with realistic, user behavior

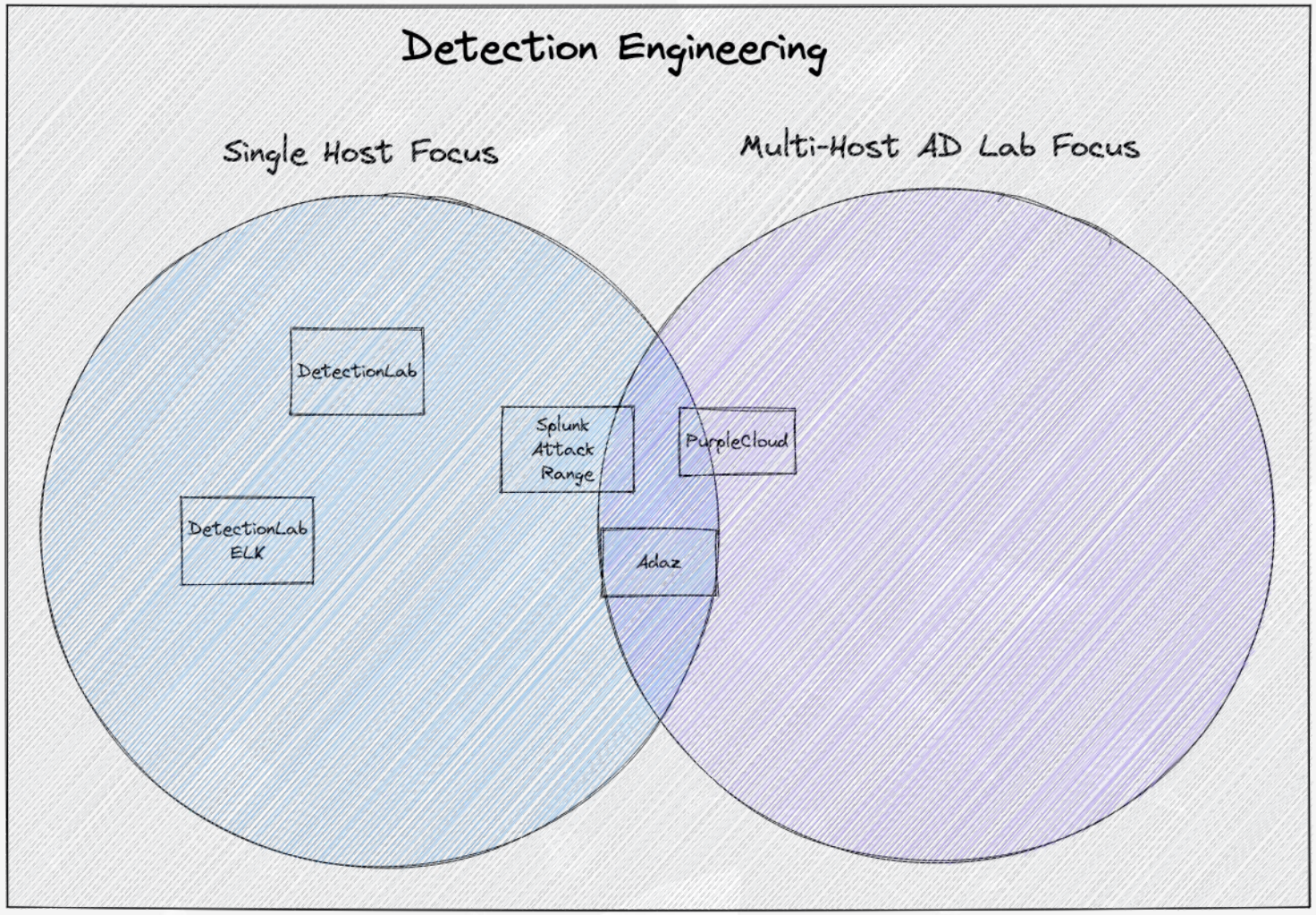
Image Source: https://github.com/cmu-sei/GHOSTS

GHOSTS Use Case with Simulation Labs



BlueCloud + PurpleCloud

Lab Types by Focus Area



Detection Engineering transforms an idea of how to detect a specific condition or activity into a concrete description of how to detect it.
Credit: Florian Roth

Detection Engineering is used to run emulations that improve logging and technology. The Labs that include multi-host AD will better inform complete attack coverage, exploiting trust relationships between domain joined systems and authenticated sessions.

Labs Evolution

Jason Ostrom

Oct 25, 2020 · 9 min read · Listen

Building Azure Cyber Ranges for Learning and Fun

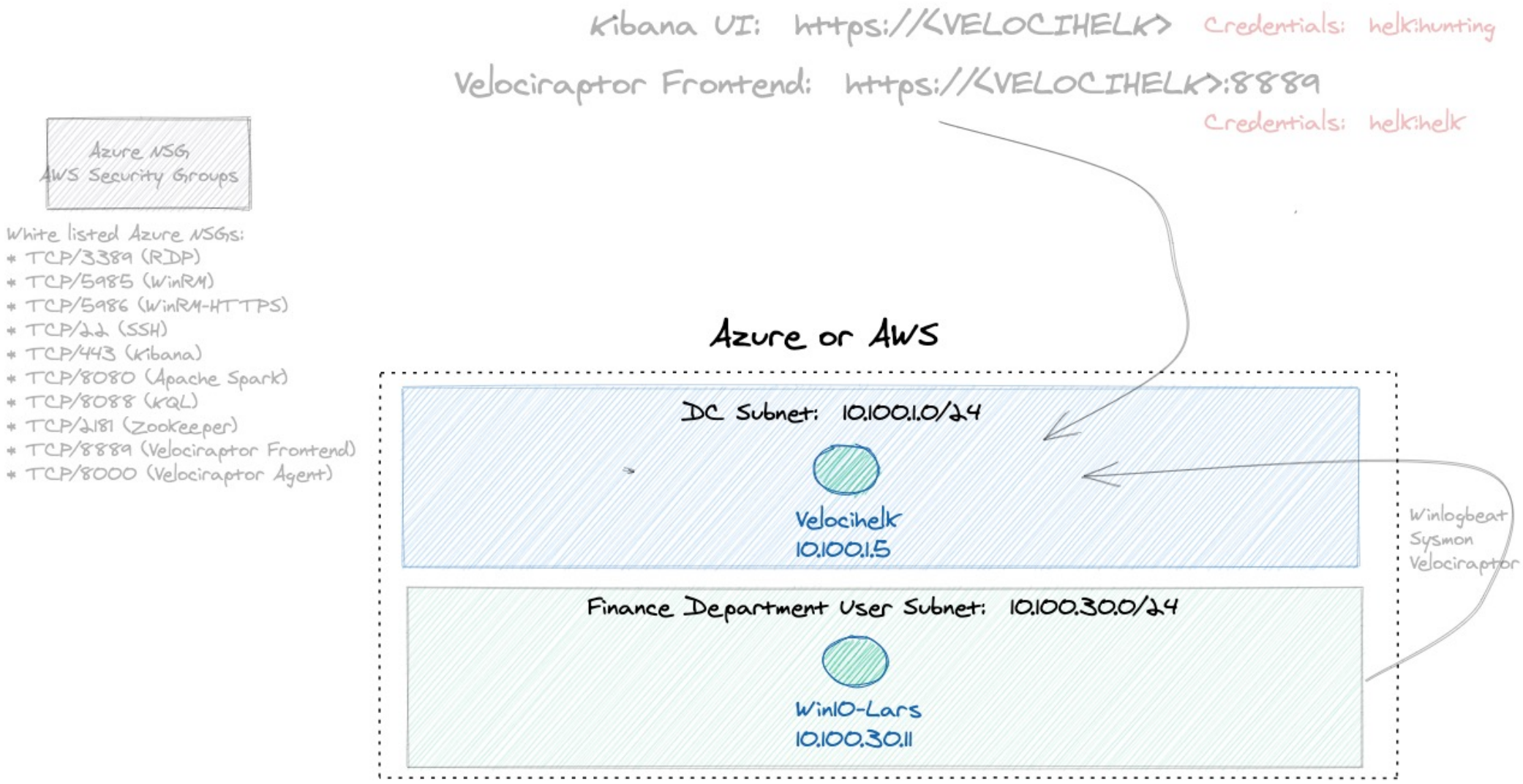
Advance your Cyber Security skills and get your Azure Security Engineer Associate certification

Overview

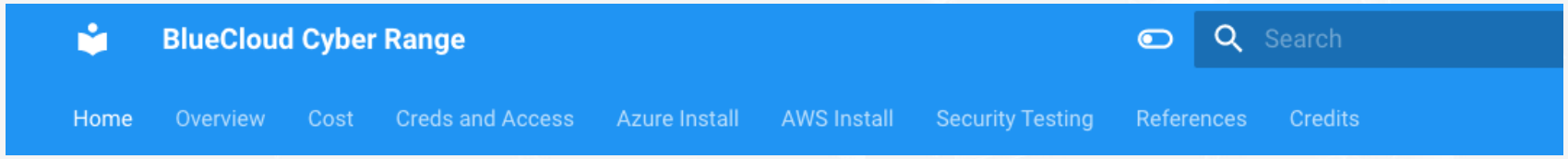
Research shows there is a Cybersecurity skills shortage that is growing worse (Oltsik, 2020). Sadly, we've grown accustomed to hearing news of companies falling victim to data breaches.

- Azure HELK
- Azure Velociraptor
- BlueCloud
- PurpleCloud

BlueCloud

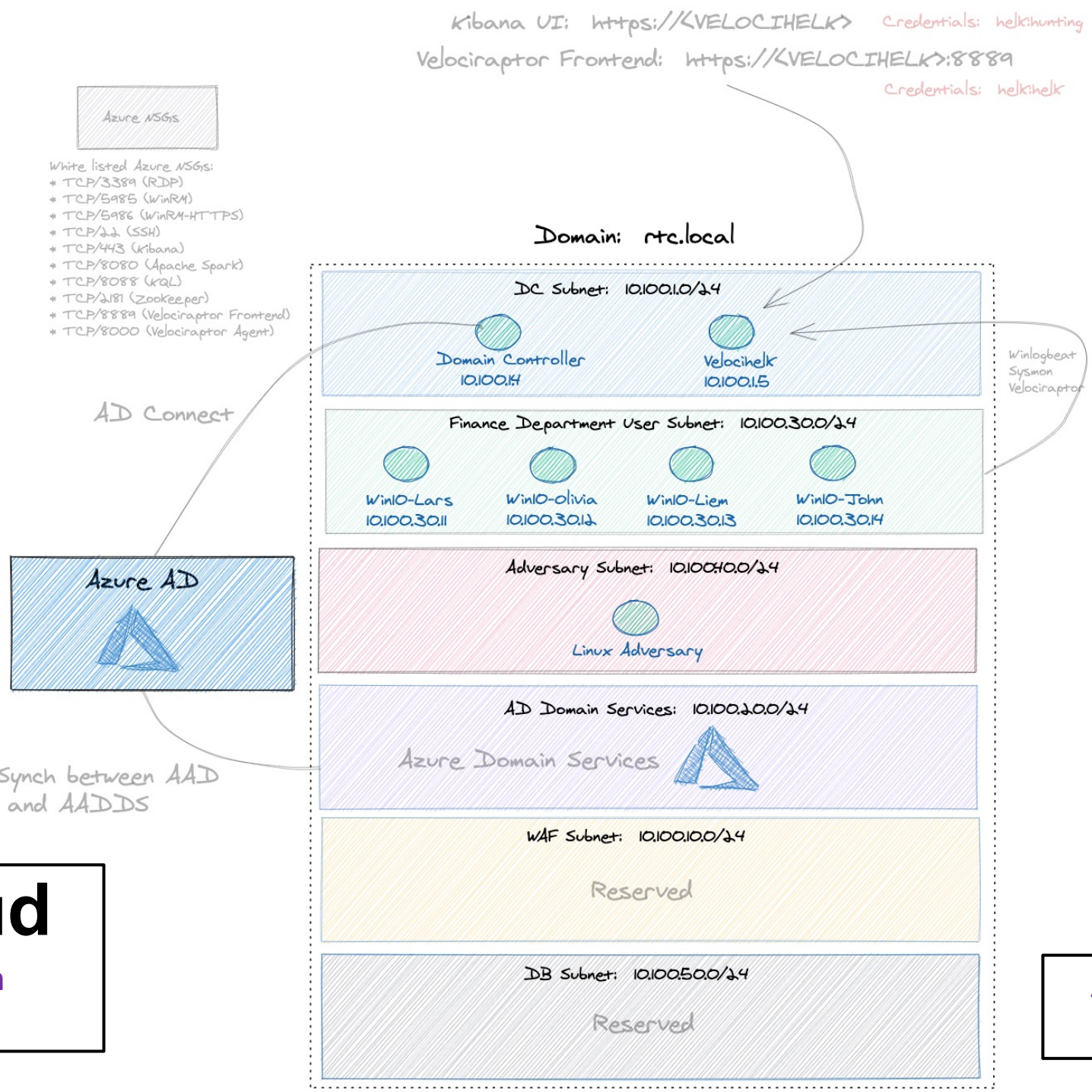

Kibana UI: https://<VELOCIHELK> Credentials: helk:hunting
Velociraptor Frontend: https://<VELOCIHELK>:8889
Credentials: helk:helk
Azure NSG
AWS Security Groups
White listed Azure NSGs:
* TCP/3389 (RDP)
* TCP/5985 (WinRM)
* TCP/5986 (WinRM-HTTPS)
* TCP/22 (SSH)
* TCP/443 (Kibana)
* TCP/8080 (Apache Spark)
* TCP/8088 (KQL)
* TCP/2181 (Zookeeper)
* TCP/8889 (Velociraptor Frontend)
* TCP/8000 (Velociraptor Agent)
Azure or AWS
DC Subnet: 10.100.1.0/24
Velocihelk
10.100.1.5
Winlogbeat
Sysmon
Velociraptor
Finance Department User Subnet: 10.100.30.0/24
Win10-Lars
10.100.30.11

BlueCloud Simulation Lab



- Easily spin up a small Detection Engineering lab in AWS or Azure
- Logging server runs Velociraptor + HELK (velocihelk)
- Windows endpoint instrumented with Velociraptor agent that auto-registers
- Windows endpoint instrumented with Winlogbeat that ships Sysmon logs using Kafka transport to HELK
- Three tools on endpoint for adversary simulation
  - Atomic Red Team, Elastic Detection RTA, and APTSimulator

BlueCloud is currently single Windows host.

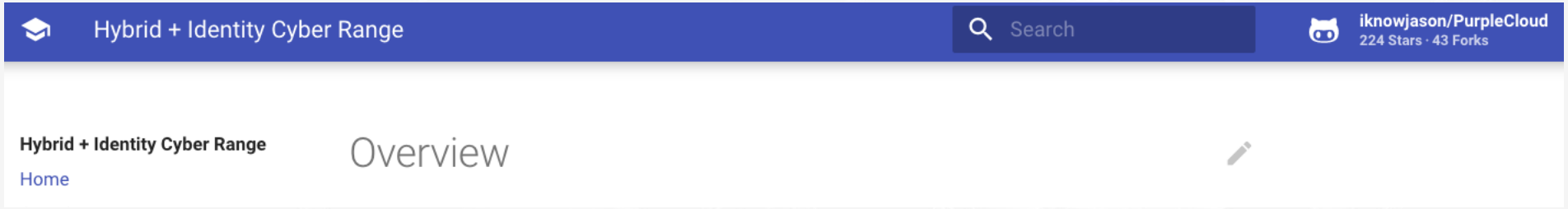


PurpleCloud

A little tool to play with Azure Identity

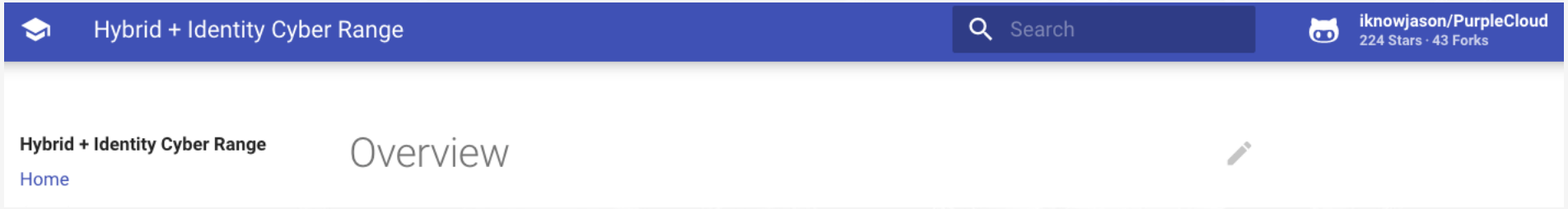
Azure AD lab creation tool

PurpleCloud Simulation Lab (1)



- ***PurpleCloud*** is an open-source tool that automates creation of simulation labs in Azure
  - **Site:** https://www.purplecloud.network
  - **Author:** Jason Ostrom
  - Terraform code generators that create unique labs for different use cases

PurpleCloud Simulation Lab (2)

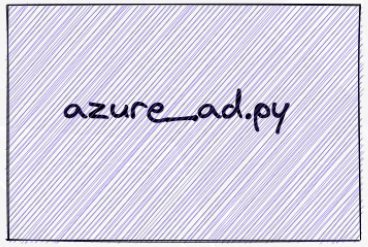


- ***What it is***
  - "Build your own lab" style of lab creation (unstructured)
  - For security researchers, Blue, Red, other security enthusiasts
  - Run attack simulations and understand defenses
  - Bug Bounty
  - Mix and match labs to create a custom, Hybrid Identity enterprise
  - User Story: used it for creating a Detection Engineering training class
- ***What it is not***
  - Guided, structured vulnerability labs

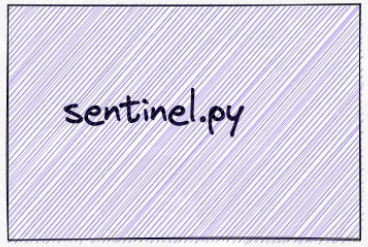
Generators Overview

Create an AD Lab + SIEM

Create an Azure AD Lab



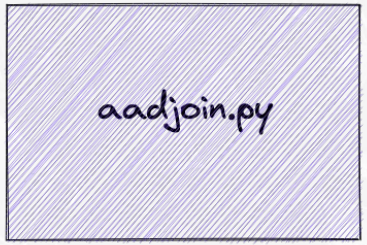
Create an Azure Sentinel Lab



Create an App Consent Phishing

Create an ADFS Lab

Azure AD Join VMs



Create a managed identity lab

Create Azure Storage lab

Generator Usage & Workflow

- **Step 1:** Run python terraform code generator
- **Step 2:** Initialize terraform configuration
- **Step 3:** Create a terraform plan
- **Step 4:** Apply terraform plan
- **Step 5:** Terraform destroy

```
% cd generators/adfs
% python3 adfs.py
% terraform init
% terraform plan –out=plan.run
% terraform apply plan.run
% terraform destroy
```

Azure Active Directory

Azure AD Lab Generator

- Creates an ***Azure Active Directory*** lab filled with Azure users, groups, and applications
- **Key Features**
  - Randomly generate Azure AD users using faker library for simulated users
  - Customizable list of Groups Added (Default: 11)
  - Customizable list of Azure AD Applications and Service Principals (Default: 7)
  - Users auto-assigned randomly into Groups
  - Includes a vulnerable privilege escalation scenario and attack scripts
- **Example Usage**

```
% python3 azure_ad.py --upn rtcfingroup.com --count 500 --apps 3 --groups 5
```

First of its kind Azure AD open-source tool to auto-generate Azure AD lab

Create 1,000 Azure AD users

```
sec588@slingshot:~/tools/PurpleCloud$ python3 azure_ad.py --upn tecniqa.co -c 1000
[+] Number of users desired:  1000
[+] upn suffix:  tecniqa.co
[+] Creating unique user list
    [-] Duplicate user Christopher Bailey ~ not adding to users list
    [-] Duplicate user Michelle Hernandez ~ not adding to users list
    [-] Duplicate user Michelle Hamilton ~ not adding to users list
    [-] Duplicate user Michael Miller ~ not adding to users list
    [-] Duplicate user Cody Martin ~ not adding to users list
    [-] Duplicate user Tammy Brown ~ not adding to users list
    [-] Duplicate user David Bell ~ not adding to users list
    [-] Duplicate user Michael Taylor ~ not adding to users list
    [-] Duplicate user William Johnson ~ not adding to users list
    [-] Duplicate user James Reed ~ not adding to users list
[+] Number of users added into list:  1000
[+] Number of duplicate users filtered out:  10
[+] Creating output files for Azure AD Users
    [+] Users csv file:  azure_users.csv
    [+] Username txt file:  azure_usernames.txt
    [+] Email addresses txt file:  azure_emails.txt
    [+] Terraform file:  users.tf
sec588@slingshot:~/tools/PurpleCloud$
```

Outputs Files in Text and CSV for other tools

```
sec588@slingshot:~/tools/PurpleCloud$ wc -l azure_users.csv azure_usernames.txt azure_emails.txt
 1000 azure_users.csv
 1000 azure_usernames.txt
 1000 azure_emails.txt
 3000 total
sec588@slingshot:~/tools/PurpleCloud$
sec588@slingshot:~/tools/PurpleCloud$ more azure_users.csv
Wendy Pierce,wendypierce,wendypierce@tecniqa.co
Jennifer Hardy,jenniferhardy,jenniferhardy@tecniqa.co
Allen Camacho,allencamacho,allencamacho@tecniqa.co
Matthew Miller,matthewmiller,matthewmiller@tecniqa.co
Samantha Mejia,samanthamejia,samanthamejia@tecniqa.co
Sean Guerrero,seanguerrero,seanguerrero@tecniqa.co
Ashley Parks,ashleyparks,ashleyparks@tecniqa.co
Nancy Cole,nancycole,nancycole@tecniqa.co
Deanna Castillo,deannacastillo,deannacastillo@tecniqa.co
Aaron White,aaronwhite,aaronwhite@tecniqa.co
Valerie Sherman,valeriesherman,valeriesherman@tecniqa.co
Jackie Gibson,jackiegibson,jackiegibson@tecniqa.co
Michael Taylor,michaeltaylor,michaeltaylor@tecniqa.co
Judith Rivera,judithrivera,judithrivera@tecniqa.co
```

Create Azure Applications and Groups: Auto-assign users into groups

```
sec588@slingshot:~/tools/PurpleCloud$ python3 azure_ad.py --upn tecniqa.co --apps 3 --groups 5
[+] No users specified ~ creating 100 users by default
[+] upn suffix:  tecniqa.co
[+] Desired applications enabled:   3
[+] Desired groups enabled:   5
[+] Creating unique user list
[+] Number of users added into list:  100
[+] Number of duplicate users filtered out:  0
[+] Creating output files for Azure AD Users
    [+] Users csv file:  azure_users.csv
    [+] Username txt file:  azure_usernames.txt
    [+] Email addresses txt file:  azure_emails.txt
    [+] Terraform file:  users.tf
[+] Creating terraform file:  apps.tf
[+] Creating terraform file:  groups.tf
    [+] Adding all Azure users to this group:  Users
sec588@slingshot:~/tools/PurpleCloud$
```

100 users are randomly placed into 5 different Azure AD Groups.

```
# Azure AD Group
resource "azuread_group" "Sales_Team" {
  display_name = "${var.upn_suffix} - Sales"
  security_enabled = true
  members = [
    azuread_user.user1.object_id,
    azuread_user.user4.object_id,
    azuread_user.user5.object_id,
    azuread_user.user9.object_id,
    azuread_user.user15.object_id,
    azuread_user.user17.object_id,
    azuread_user.user19.object_id,
    azuread_user.user20.object_id,
    azuread_user.user26.object_id,
```

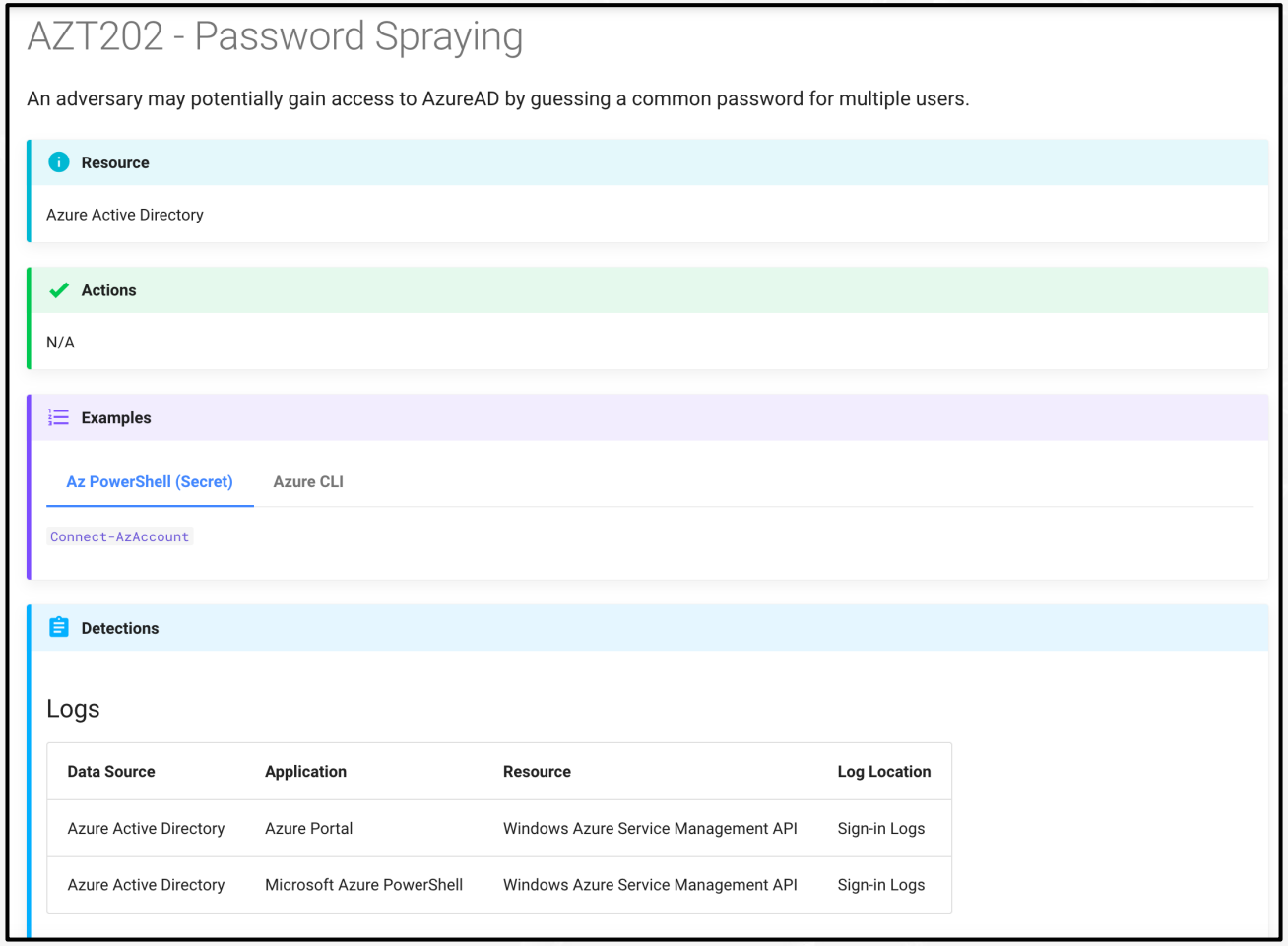
Service Principal Abuse Attack Scenario Created

```
sec588@slingshot:~/tools/PurpleCloud$ python3 azure_ad.py -c 25 --upn tecniqa.co --apps 7 -aa -ga -pra
[+] Number of users desired:  25
[+] upn suffix:  tecniqa.co
[+] Desired applications enabled:   7
[+] Creating unique user list
[+] Number of users added into list:  25
[+] Number of duplicate users filtered out:  0
[+] Creating output files for Azure AD Users
    [+] Users csv file:  azure_users.csv
    [+] Username txt file:  azure_usernames.txt
    [+] Email addresses txt file:  azure_emails.txt
    [+] Terraform file:  users.tf
[+] Creating terraform file:  apps.tf
    [+] Assigning the Privileged Role Administrator to MailReader_Application
    [+] Assigning the Global Administrator role to HelpDesk_Application
sec588@slingshot:~/tools/PurpleCloud$
```

Hat tip and credit to security researchers (Andy Robbins, Dirk-jan Mollema) for their writeups on this issue. Original articles are included in references section.

Adversary Behaviors

- Simulate adversary techniques mapped to the Azure Threat Research Matrix: https://microsoft.github.io/Azure-Threat-Research-Matrix/



AZT202 - Password Spraying

An adversary may potentially gain access to AzureAD by guessing a common password for multiple users.

Resource

Azure Active Directory

Actions

N/A

Examples

Az PowerShell (Secret) | Azure CLI

```
Connect-AzAccount
```

Detections

Logs

Data Source	Application	Resource	Log Location
Azure Active Directory	Azure Portal	Windows Azure Service Management API	Sign-in Logs
Azure Active Directory	Microsoft Azure PowerShell	Windows Azure Service Management API	Sign-in Logs

Detection Engineering (1)

- Start the process of Detection Engineering with Adversary Emulations
- **Password Spray:** Attacker rotates IP addresses using Amazon API Gateway

User sign-ins (interactive) | User sign-ins (non-interactive) | Service principal sign-ins | Managed identity sign-ins

Date	Request ID	User	Application	Status	IP address	Location	Conditional Access
10/26/2022, 7:33:56 PM	f2a7a2df-460d-41f5-8b...	Jason Ostrom	Azure Portal	Success	150.249.204.46	Chiyoda-Ku, Tokyo, JP	Not Applied
10/26/2022, 9:10:16 AM	d7682cfd-4803-4964-9...	Rachel Mcdonald	Office 365 Management	Success	35.181.128.179	Paris, Paris, FR	Not Applied
10/26/2022, 9:10:12 AM	1e505348-f238-43ff-8c...	Anna Ramirez	Microsoft Teams	Success	3.12.219.188	Columbus, Ohio, US	Not Applied
10/26/2022, 9:10:08 AM	eaaa4454-d6b2-4929-a...	Tammy Stevenson	Microsoft Azure Active ...	Success	15.228.151.21	Sao Paulo, Sao Paulo, BR	Not Applied
10/26/2022, 9:10:04 AM	2ee6a6a3-db87-426c-8f...	Jamie Wiggins		Failure	16.171.48.192	Stockholm, Stockholms ...	Not Applied
10/26/2022, 9:09:59 AM	c45cdf44-8820-4ef8-8af...	Bethany Yates	Microsoft Authenticator...	Success	3.101.200.140	San Jose, California, US	Not Applied
10/26/2022, 9:09:56 AM	89257053-71a8-48fd-8...	Emily Freeman		Failure	3.72.33.135	Frankfurt Am Main, Hes...	Not Applied
10/26/2022, 9:09:51 AM	039d1f53-f4f1-422f-8b7...	Jason Stevens		Failure	13.40.207.210	London, Greater Londo...	Not Applied
10/26/2022, 9:09:46 AM	24981e82-f0b8-436b-b...	Andrew Bowman	SkypeForBusinessAuth	Interrupted	35.93.127.125	Boardman, Oregon, US	Not Applied
10/26/2022, 9:09:43 AM	1af28d2e-a85a-4d18-9...	Robert Vasquez		Failure	44.206.4.236	Ashburn, Virginia, US	Not Applied
10/26/2022, 9:09:39 AM	9cd90787-1685-475a-8...	James Bennett	Office365 Shell WCSS-C...	Failure	108.128.160.235	Dublin, Dublin, IE	Not Applied
10/26/2022, 9:09:34 AM	fef9b58e-20c9-4758-b9...	Barbara Jones		Failure	13.38.133.137	Paris, Paris, FR	Not Applied
10/26/2022, 9:09:30 AM	5315239c-f007-45a8-be...	Crystal Baxter	Microsoft Exchange Onl...	Success	3.15.35.73	Columbus, Ohio, US	Not Applied
10/26/2022, 9:09:26 AM	032fc0ef-ce07-43d7-90...	Amy Martinez	Microsoft Office	Success	18.229.99.159	Sao Paulo, Sao Paulo, BR	Not Applied
10/26/2022, 9:09:21 AM	dec2c525-1e2e-4a70-a...	Joel Walker		Failure	16.171.49.70	Stockholm, Stockholms ...	Not Applied
10/26/2022, 9:09:17 AM	12d547c0-6767-4e4b-b...	Christopher Carr	Office365APIEditor	Interrupted	13.52.201.217	San Jose, California, US	Not Applied
10/26/2022, 9:09:14 AM	ae03351f-c742-4eff-b8...	Kristin Smith	Microsoft Azure	Success	3.66.172.9	Frankfurt Am Main, Hes...	Not Applied
10/26/2022, 9:09:09 AM	ebcdc5cc-a9ee-49c1-a5...	Luis Thompson	ADIbizaUX	Failure	13.40.205.187	London, Greater Londo...	Not Applied

Adversary runs a Password Spray and rotates their source IP address using Amazon API Gateway. Note the different location and user in each request.

Detection Engineering (2)

- Password Spraying using Amazon API Gateway can be detected and blocked
  - Use Amazon's published IP prefixes to parse all API Gateway prefixes
  - Lookup IP address of password attempt against the list
  - Instrument SIEM, WAF, or Firewall to detect a threshold of password attempts and block

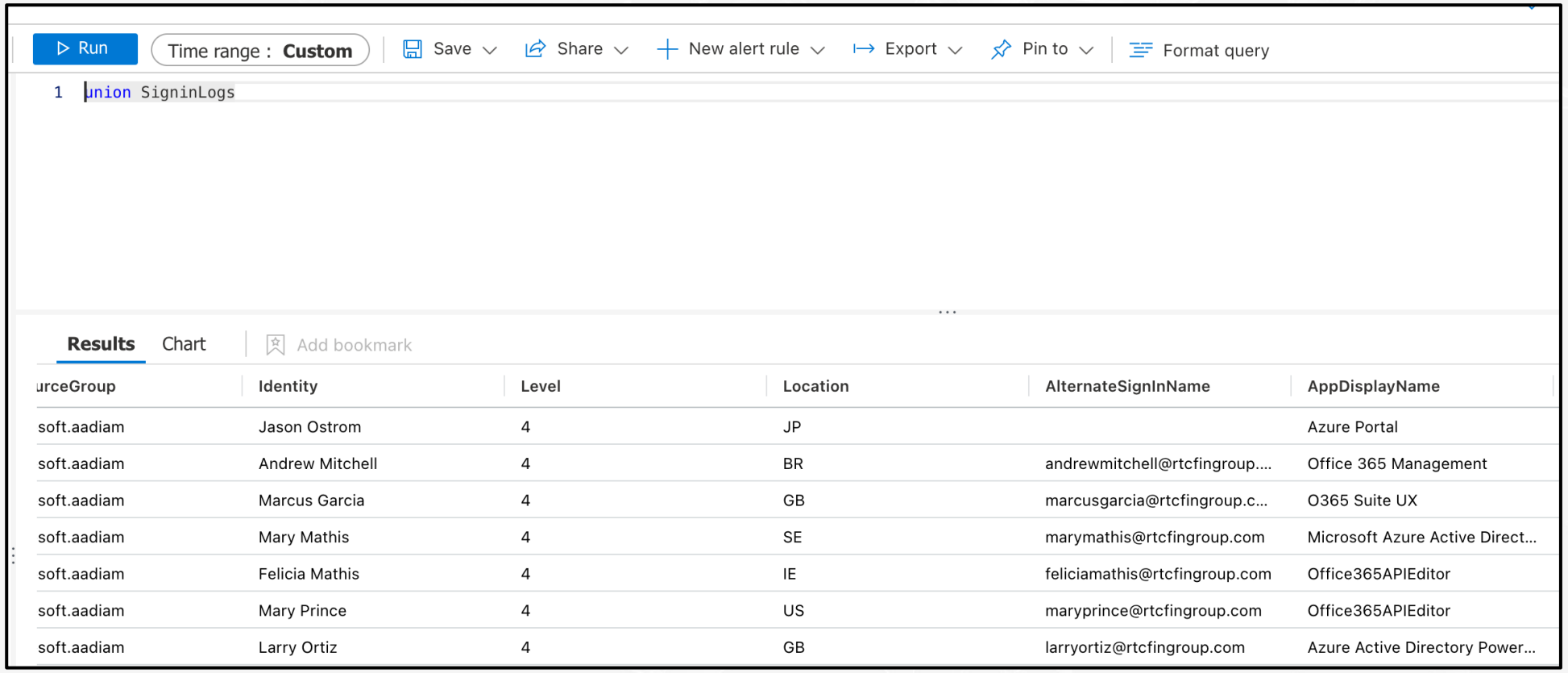
```
curl https://ip-ranges.amazonaws.com/ip-ranges.json | jq -r '.prefixes[] | select(.service=="API_GATEWAY") | .ip_prefix'
```

```
sec588@slingshot:~/Desktop/requests-ip-rotator$ ./cmd.sh
[+] Listing all Amazon API Gateway prefixes from ip-ranges.json
140.179.144.128/25
140.179.176.0/23
52.81.135.128/25
52.81.137.0/24
52.81.216.0/23
161.189.148.0/23
52.82.127.0/24
```

Sentinel Detections

- Use Azure Sentinel and KQL queries for detection improvement
- Example KQL: ***union SigninLogs***

Build KQL detections and automate processes for detection engineering. Map Azure-specific TTPs using the Azure Threat Research Matrix.



urceGroup	Identity	Level	Location	AlternateSignInName	AppDisplayName
soft.aadiam	Jason Ostrom	4	JP		Azure Portal
soft.aadiam	Andrew Mitchell	4	BR	andrewmitchell@rtcfingroup....	Office 365 Management
soft.aadiam	Marcus Garcia	4	GB	marcusgarcia@rtcfingroup.c...	O365 Suite UX
soft.aadiam	Mary Mathis	4	SE	marymathis@rtcfingroup.com	Microsoft Azure Active Direct...
soft.aadiam	Felicia Mathis	4	IE	feliciamathis@rtcfingroup.com	Office365APIEditor
soft.aadiam	Mary Prince	4	US	maryprince@rtcfingroup.com	Office365APIEditor
soft.aadiam	Larry Ortiz	4	GB	larryortiz@rtcfingroup.com	Azure Active Directory Power...

Service Principal Privilege Abuse Primitives

AZT405 - Azure AD Application

Adversaries may abuse the assigned permissions on an Azure AD Application to escalate their privileges.

ID	Name	Description	Action	Resources
AZT405.1	Application Role	By compromising a user, user in a group, or service principal that has an application role over an application, they may be able to escalate their privileges by impersonating the associated service principal and leveraging any privileged assigned application role.		AzureAD
AZT405.2	Application API Permissions	By compromising a service principal whose application has privileged API permissions, an attacker can escalate their privileges to a higher privileged role.		AzureAD
AZT405.3	Application Registration Owner	By compromising an account who is an 'Owner' over an application that is configured with additional roles or API permissions, an attacker can escalate their privileges by adding a certificate or credentials & logging in as the service principal.		AzureAD

Detection Engineering & Sentinel

- Example KQL

```
AuditLogs
|where OperationName =="Update application – Certificates and secrets management" and
Category=="ApplicationManagement"
```

- Detect Service Principal Logins
- Detect changes to App Role Assignments
- Detect adding secrets to applications

Results | Chart | Add bookmark

TimeGenerated [UTC]	ResourceId	OperationName	OperationVersion	Category
10/30/2022, 8:13:35.333 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Add member to group	1.0	GroupManagement
10/30/2022, 8:13:35.399 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Add member to group	1.0	GroupManagement
10/30/2022, 8:14:13.769 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Add member to group	1.0	GroupManagement
10/30/2022, 8:46:26.790 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Update service principal	1.0	ApplicationManagement
10/30/2022, 8:46:26.868 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Update application – Certificates and secrets managem...	1.0	ApplicationManagement
10/30/2022, 8:46:26.884 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Update application	1.0	ApplicationManagement
10/30/2022, 8:46:41.003 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Add member to role	1.0	RoleManagement
10/30/2022, 8:47:41.325 AM	/tenants/1a82558d-66e0-48b0-b370-72df4caf1852/pr...	Add member to role outside of PIM (permanent)	1.0	RoleManagement

Active Directory with SIEM & Velociraptor

Active Directory + SIEM Lab Generator (1)

- ***ad.py*** is a Terraform generator for an Active Directory Lab created with Azure VMs and includes support for Velociraptor and a SIEM using Hunting ELK
- **Key Features**
  - Create a custom IaaS Active Directory environment with Azure VMs
  - Deploys a SIEM (Hunting ELK) and endpoints instrumented with Sysmon/Winlogbeat/Velociraptor/Atomic Red Team
  - Endpoints include PurpleSharp
  - Downloads Azure AD Connect msi installer onto DC's desktop
- **Example Usage**

```
% python3 ad.py --domain_controller --ad_domain rtcfingroup.com --admin RTCAdmin --password
MyPassword012345 --csv users.csv --endpoints 2 --domain_join
```

Active Directory + SIEM Lab Generator (2)

- Create a customizable, realistic, large AD environment
  - Import users from CSV, or randomly generate users
  - Random Generator creates as many users as you desire
- Automatically creates OU, AD Groups, and assigns users into OU, Groups
- Automatic Domain Join: Configurable Domain Join per VM
- Auto-logon Domain users with Domain credentials, for realistic simulations (Interactive Type 2 Logon, Mimikatz)
- Great for practicing or learning Active Directory

Simulate an On-Premise Active Directory Lab

Create an AD Domain with 500 AD users. Create three Windows 10 Professional endpoints, joining them to the domain.

```
sec588@slingshot:~/tools/PurpleCloud$ python3 ad.py --domain_controller --ad_domain stora.io --admin StoraAd
min --ad_users 500 --endpoints 3 --domain_join --auto_logon
[+] Public IP address detected:  99.182.28.252
[+] Setting Azure NSG Whitelist to:  99.182.28.252
[+] Local Admin account name:   StoraAdmin
[+] Setting AD Domain to build AD DS: stora.io
[+] Creating unique user list
    [-] Duplicate user Joshua Jones ~ not adding to users list
[+] Number of users added into list:  500
[+] Number of duplicate users filtered out:  1
[+] Number of Windows 10 Pro endpoints desired:  3
[+] Using default Resource Group Name:  PurpleCloud
[+] Using default location:  eastus
[+] Domain Join is set to true
[+] Auto Logon is set to true
[+] Creating the main terraform file:  main.tf
[+] Creating the providers terraform file:  providers.tf
[+] Creating the nsg terraform file:  nsg.tf
[+] Building Windows 10 Pro
  [+] Number of systems to build:  3
    [+] Getting default configuration template for Windows 10 Pro
    [+] Base Hostname: win10
    [+] Administrator Username: StoraAdmin
    [+] Administrator Password: 9a80sgQzCK
```

Users Placed into OU and AD Groups automatically

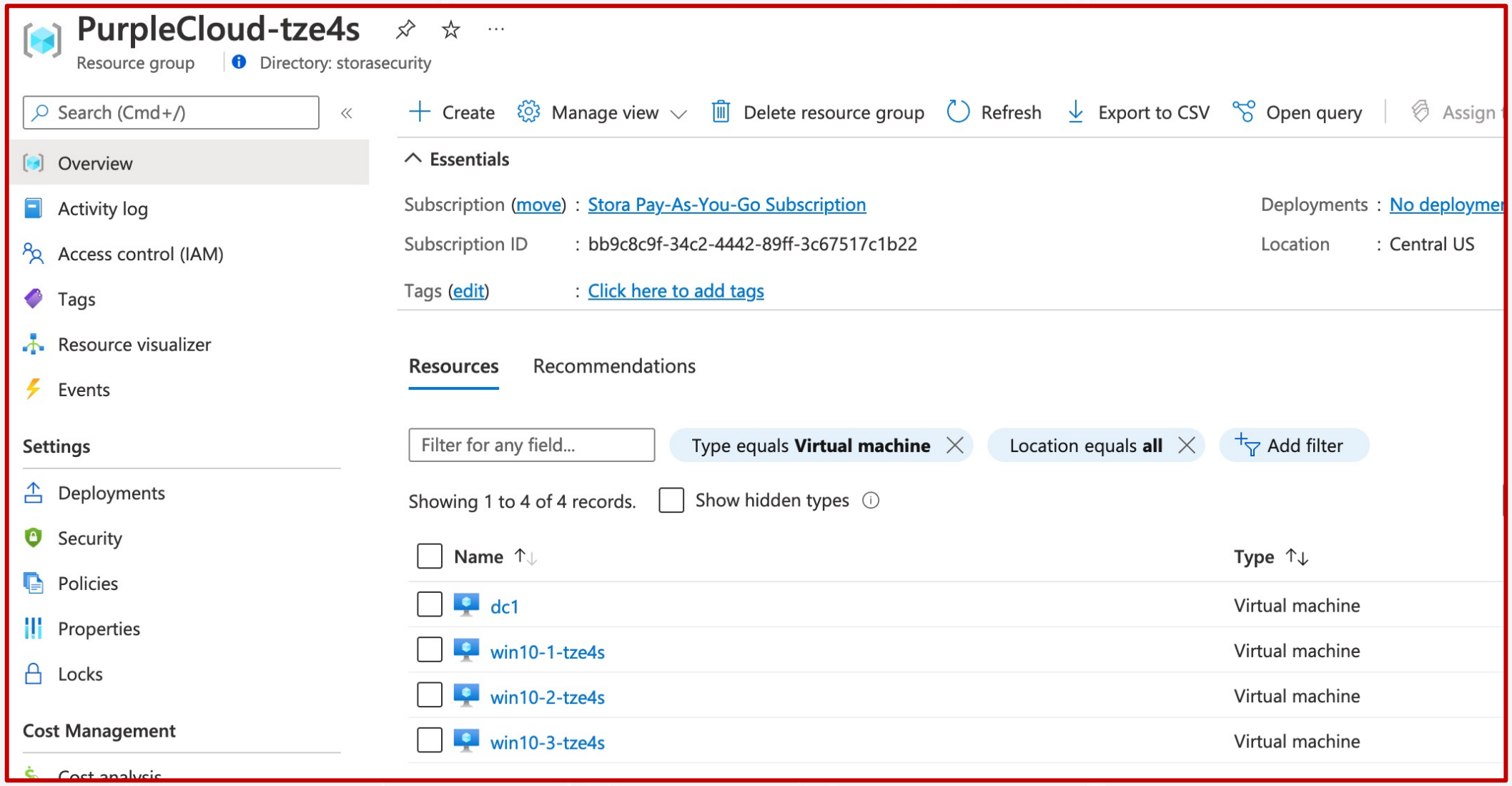
```
sec588@slingshot:~/tools/PurpleCloud$ wc -l ad_users.csv
506 ad_users.csv
sec588@slingshot:~/tools/PurpleCloud$ more ad_users.csv
name,upn,password,groups,oupath,domain_admin
Lars Borgerson,larsborgerson@stora.io,M1naKXXy4n,IT,OU=IT;DC=stora;DC=io,False
Olivia Odinsdottir,oliviaodinsdottir@stora.io,BHvyO4tR9w,IT,OU=IT;DC=stora;DC=io,True
Liem Anderson,liemanderson@stora.io,tV6QPh9cmo,IT,OU=IT;DC=stora;DC=io,False
John Nilsson,johnnilsson@stora.io,83sopkFaNh,IT,OU=IT;DC=stora;DC=io,False
Jason Lindqvist,jasonlindqvist@stora.io,Y9EGolh8kO,IT,OU=IT;DC=stora;DC=io,True
Brittany Martin,brittanymartin@stora.io,mE6YdstoM4,Engineering,OU=Engineering;DC=stora;DC=io,False
Danielle Gill,daniellegill@stora.io,mE6YdstoM4,Engineering,OU=Engineering;DC=stora;DC=io,False
James Grant,jamesgrant@stora.io,mE6YdstoM4,IT,OU=IT;DC=stora;DC=io,False
Victor Johnson,victorjohnson@stora.io,mE6YdstoM4,IT,OU=IT;DC=stora;DC=io,False
Mary Johnson,maryjohnson@stora.io,mE6YdstoM4,Marketing,OU=Marketing;DC=stora;DC=io,False
Emily Byrd,emilybyrd@stora.io,mE6YdstoM4,Executive,OU=Executive;DC=stora;DC=io,False
Scott Jones,scottjones@stora.io,mE6YdstoM4,Executive,OU=Executive;DC=stora;DC=io,False
Sherry Oneal,sherryoneal@stora.io,mE6YdstoM4,Legal,OU=Legal;DC=stora;DC=io,False
John Contreras,johncontreras@stora.io,mE6YdstoM4,Sales,OU=Sales;DC=stora;DC=io,False
Kenneth Hickman,kennethhickman@stora.io,mE6YdstoM4,Sales,OU=Sales;DC=stora;DC=io,False
Brandi Mcdaniel,brandimcdaniel@stora.io,mE6YdstoM4,Executive,OU=Executive;DC=stora;DC=io,False
```

Each Windows 10 Pro Endpoint has a custom Terraform file created (for further editing if desired)

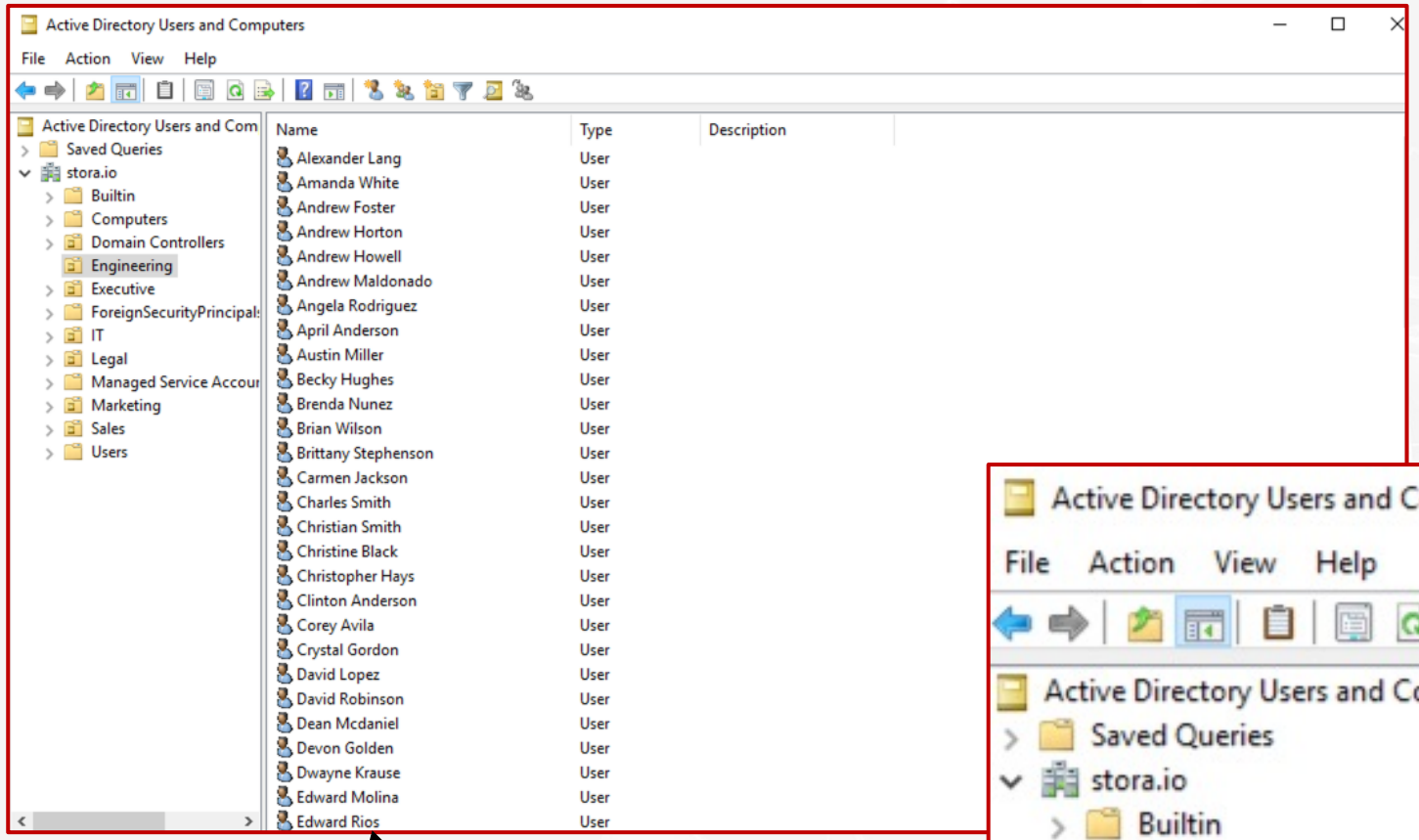
```
[+] Building Windows 10 Pro Endpoint 1
  [+] Hostname: win10-1
  [+] IP address: 10.100.20.10
  [+] Auto Logon Domain user
    [+] Getting the default ad user and password
    [+] Auto Logon this Win10 Pro to AD User:  Alan Gill
    [+] Username:  alangill
    [+] Password:  mE6YdstoM4
  [+] Setting Domain Controller for this endpoint to join domain:  10.100.10.4
  [+] Created terraform: win10-1.tf
```

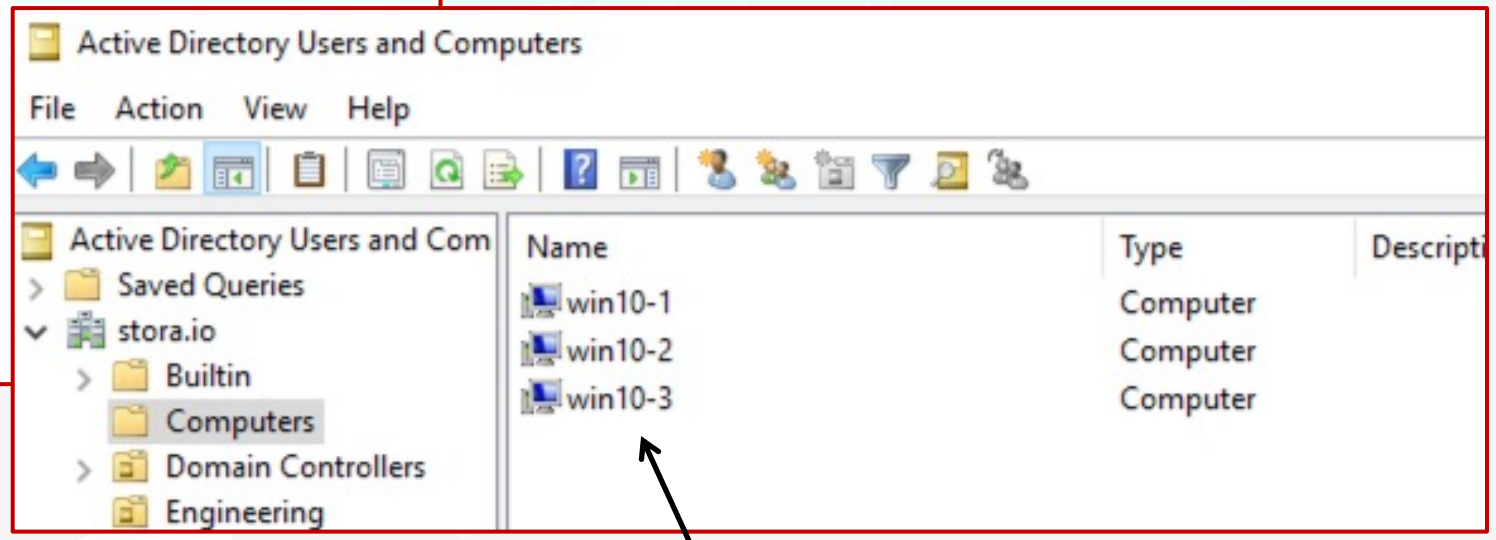
```
sec588@slingshot:~/tools/PurpleCloud$ ls -al win10-*
-rw-rw-r-- 1 sec588 sec588 5524 Jul  9 11:21 win10-1.tf
-rw-rw-r-- 1 sec588 sec588 5520 Jul  9 11:21 win10-2.tf
-rw-rw-r-- 1 sec588 sec588 5518 Jul  9 11:21 win10-3.tf
sec588@slingshot:~/tools/PurpleCloud$
```

Virtual Machines Created



Active Directory Created with 3 Domain Joined

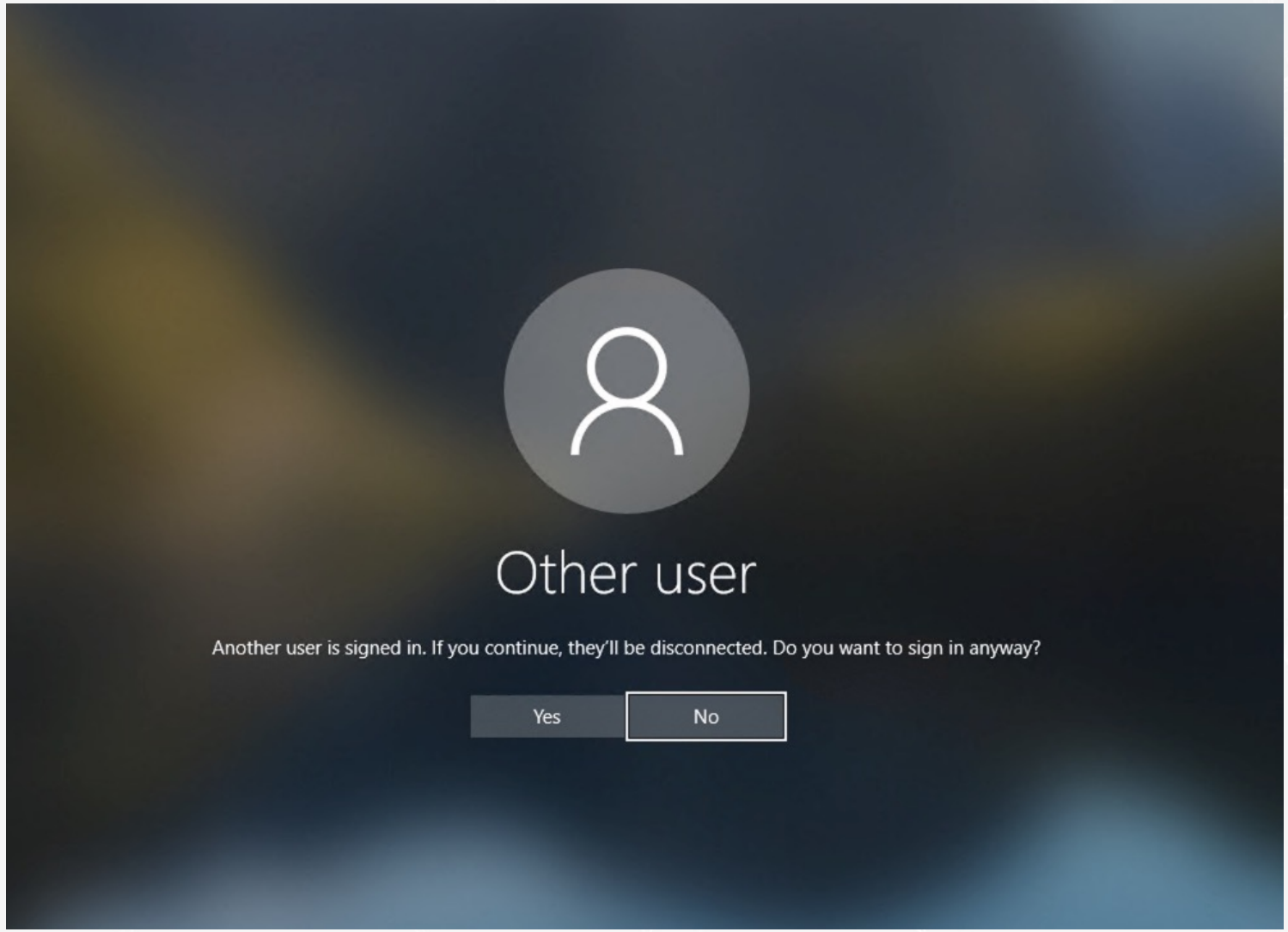




500 Domain Users assigned into different OU and AD Groups.

Three Windows 10 Pro joined to the domain based on Python script.

Auto Logon Domain Users with AD credentials



With this feature, you can practice lateral movements across domain joined systems and extracting domain credentials from LSASS memory.

Passwords in AD default to Strong, but customizable

```
[+] Default Local Administrator Credentials on all Windows
  [+] Username:  StoraAdmin
  [+] Password:  MiXz47PXnh
[+] Built AD DS Domain: stora.io
```

Default behavior is to auto-generate a strong password and assign to all users, putting into CSV file.

```
Lars Borgerson,larsborgerson@stora.io,4NgLlSurEg,IT,OU=IT;DC=stora;DC=io,False
Olivia Odinsdottir,oliviaodinsdottir@stora.io,NzZw2AzjTy,IT,OU=IT;DC=stora;DC=io,True
Liem Anderson,liemanderson@stora.io,dtdBP8USiT,IT,OU=IT;DC=stora;DC=io,False
John Nilsson,johnnilsson@stora.io,EMX0BjYoN8,IT,OU=IT;DC=stora;DC=io,False
Jason Lindqvist,jasonlindqvist@stora.io,CcoW4whUK6,IT,OU=IT;DC=stora;DC=io,True
Samantha Simmons,samanthasimmons@stora.io,c2aiIXy4ef,Sales,OU=Sales;DC=stora;DC=io,False
Renee Ramirez,reneeramirez@stora.io,c2aiIXy4ef,Sales,OU=Sales;DC=stora;DC=io,False
```

Specify your desired password for all users via command line parameter.

```
sec588@slingshot:~/tools/PurpleCloud$ python3 ad.py --domain_controller --ad_domain stora.io --admin StoraAd
min --password MyPassword012345 --ad_users 500 --endpoints 3 --domain_join --auto_logon
```

Build Hunting ELK + Velociraptor with one endpoint

```
sec588@slingshot:~/tools/PurpleCloud$ python3 ad.py --helk --endpoint 1
[+] Public IP address detected:  99.182.28.252
[+] Setting Azure NSG Whitelist to:  99.182.28.252
[+] Number of Windows 10 Pro endpoints desired:  1
[+] Using default Resource Group Name:  PurpleCloud
[+] Using default location:  eastus
[+] HELK server enabled 10.100.30.4
[+] Installing velociraptor, winlogbeat agents on endpoints and exporting logs to HELK
[+] Creating the main terraform file:  main.tf
[+] Creating the providers terraform file:  providers.tf
[+] Creating the nsg terraform file:  nsg.tf
[+] Building Windows 10 Pro
  [+] Number of systems to build:  1
    [+] Getting default configuration template for Windows 10 Pro
    [+] Base Hostname: win10
    [+] Administrator Username: RTCAdmin
    [+] Administrator Password: 1C4qvSRIFj
    [+] Join Domain: false
    [+] Auto Logon Domain User: false
    [+] Install Sysmon: true
    [+] Install Atomic Red Team (ART): true
    [+] Forwarding winlogbeat logs to 10.100.30.4
    [+] Installing velociraptor and registering to 10.100.30.4
    [+] Subnet Association: user_subnet
  [+] Building Windows 10 Pro Endpoint 1
```

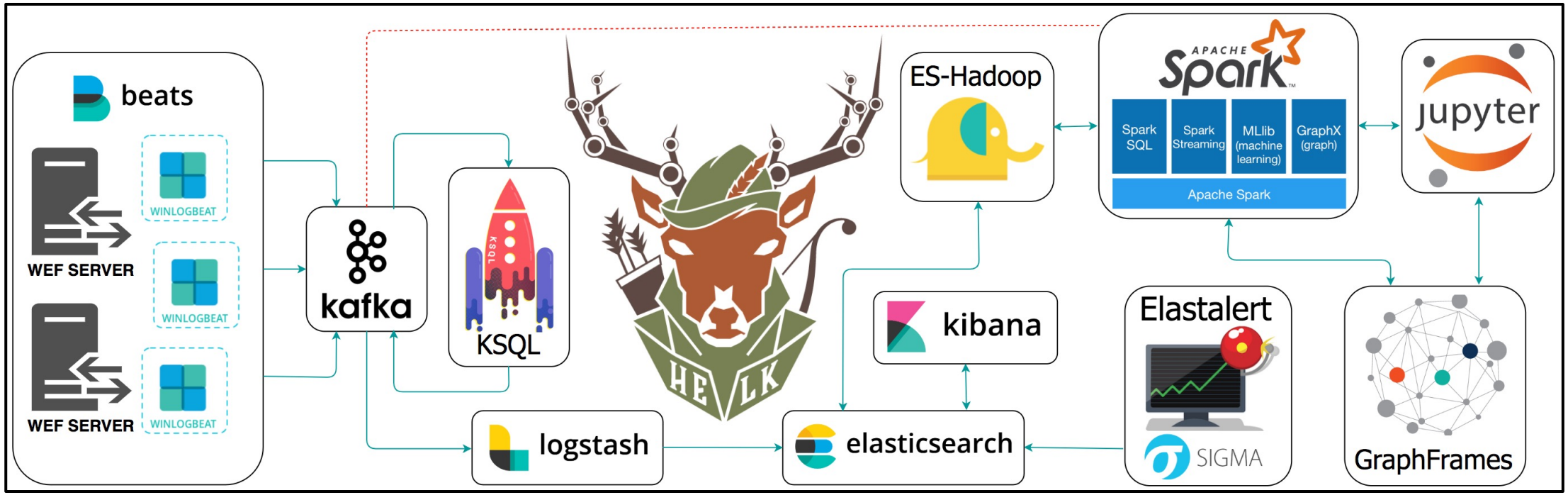
PupleCloud IaaS Advantage

- Advantage of ***PurpleCloud:***
- Practice simulations against Domain Joined workstations with users logged in with domain user credentials
- Simulate lateral movement in AD enterprise
- Simulate extraction of domain user credentials from memory
- Windows and Sysmon logs shipped to a SIEM (HELK)
  - Sysmon v14
  - Customize your XML (uses SwiftOnSecurity)
- Study forensic artifacts with Velociraptor

PurpleCloud is a solid lab to learn Active Directory

HELK: The Hunting ELK (1)

- ***PurpleCloud*** automatically builds HELK as the SIEM
  - https://github.com/Cyb3rWard0g/HELK
- Endpoints ship Sysmon logs via Winlogbeat agent
- HELK hardware option #4 is built for Jupyter Notebooks + ElastAlert



HELK: The Hunting ELK (2)

- Endpoints configured with Sysmon Version 14
- SwiftOnSecurity XML configuration
- Files included in Github repository and can be customized
  - winlogbeat.yml
  - Sysmon.zip
  - Symonconfig-export.xml

```
user@host ad % cd files/velocihelk
user@host velocihelk %
user@host velocihelk % ls
helk.sh.tpl
sysmonconfig-export.xml
winlogbeat.yml.tpl
user@host velocihelk %
```

Velociraptor Live Response

- ***PurpleCloud*** automatically install a Velociraptor Server
- Deploys Velociraptor agent on Windows systems
- https://github.com/Velocidex/velociraptor
- Endpoint visibility tool for digital forensics and live response
- Uses Velociraptor Query Language (VQL) to interrogate hosts and pull forensic artifacts

Hybrid Identity

Hybrid Identity Lab

- Advantage of ***PurpleCloud***
- Creates an Azure AD lab + On-Premise AD lab
- Automates creation of a mixed-use Hybrid Identity lab
- Drops latest Azure AD Connect on DC's Desktop
  - Use Azure AD Connect agent to synchronize users from on-premise to Azure AD
  - Saves time and effort doing manual installation
- Attack simulations against Azure AD Joined Windows 10 and Hybrid Joined devices that are also joined to On-Premise AD

We are seeing a large growth of companies deploying Hybrid cloud, mixing On-Premise with Cloud systems, & synchronizing users into the Cloud.

Microsoft Sentinel

Microsoft Sentinel Lab Generator

- Creates a ***Microsoft Sentinel*** and log analytics workspace lab with endpoints
- **Key Features**
  - Endpoints install Azure Monitoring agents and send logs to Log Analytics workspace
  - Endpoints install Sysmon v14
  - Sentinel Data Connector for Office
  - Sentinel Data Connector for AAD
  - Supports full Active Directory deployment with Domain Join (same as ad.py)
- **Example Usage**

```
% python3 sentinel.py --domain_controller --ad_domain rtcfingroup.com --admin RTCAdmin --password
MyPassword012345 --csv users.csv --endpoints 2 --domain_join
```

Detection Engineering & Sentinel

- Microsoft Sentinel KQL for Windows Event Logs:
  - Windows Event Logs: ***union SecurityEvent***
  - Sysmon: ***union Event***
- Run *PurpleSharp*

```
PS C:\tools>
PS C:\tools> .\PurpleSharp.exe /t T1055.002,T1055.003,T1055.004
10/29/2022 04:26:27 [*]  Starting T1055.002 Simulation on win10-1
10/29/2022 04:26:27 [*]  Simulator running from C:\tools\PurpleSharp.exe with PID:9408 as win10-1\RTCAdmin
10/29/2022 04:26:27 [*]  Process notepad.exe with PID:7624 started for the injection
10/29/2022 04:26:28 [*]  Calling OpenProcess on PID:7624
10/29/2022 04:26:28 [*]  Calling VirtualAllocEx on PID:7624
10/29/2022 04:26:28 [*]  Calling WriteProcessMemory on PID:7624
10/29/2022 04:26:28 [*]  Calling CreateRemoteThread on PID:7624
10/29/2022 04:26:28 [*]  Simulation Finished
10/29/2022 04:26:28 [*]  Playbook Finished
```

PurpleSharp is a C# adversary simulation tool that executes adversary techniques. It can be used to generate and study Sysmon logs that are shipped to Sentinel. https://github.com/mvelazc0/PurpleSharp

PurpleSharp is downloaded to all Windows 10 systems and can be accessed: C:\tools\PurpleSharp.exe

Source	Microsoft-Windows-Sysmon
EventLog	Microsoft-Windows-Sysmon/Operational
Computer	win10-1.rtcfingroup.com
EventCategory	1
EventLevel	4
EventLevelName	Information
UserName	NT AUTHORITY\SYSTEM
ParameterXml	<Param>-</Param><Param>2022-10-29 04:16:04.008</Param><Param>{DAC2892A-A904-635C-BF00-000000000600}</Param><Param>3904</Param> 0000000}</Param><Param>0x3e7</Param><Param>0</Param><Param>System</Param><Param>MD5=B073F18D23BE85799A640147AF9ABA99,SHA256
EventData	<DataItem type="System.XmlData" time="2022-10-29T04:16:04.0103235+00:00" sourceHealthServiceId="E96E82D9-734B-7A20-AE83-99DFDAF5E91F"> y">Microsoft Corporation</Data><Data Name="OriginalFileName">cscript.exe</Data><Data Name="CommandLine">"C:\windows\system32\cscript.exe" / 1C54DDCDC0651A1E0D8AA5F03D11D6B0521177F,IMPHASH=2B44D2206B9865383429E9C1524F1CAC</Data><Data Name="ParentProcessGuid">{dac2
EventID	1

Consent Phishing

Application Consent Phishing Lab

- Creates a multi-tenant Azure AD application that can be used for consent phishing attack and defense
- **Key Features**
  - Multi-tenant Azure AD application that can be used to practice app consent phishing
  - Customizable Graph API permissions (Default: Mail.Read, Files.Read)
  - Script parameter to specify custom application name
  - Script parameter to specify redirect_uri
- **Example Usage**

```
% python3 phishing_app.py --name PhishingApp --redirect_uri https://www.evilcorp.io/get_token
```

Azure Threat Research Matrix

AZT203 - Malicious Application Consent

An adversary may lure a victim into giving their access to a malicious application registered in AzureAD.

Resource

Azure Active Directory

Actions

Any user can consent to an application which will impersonate that user's privileges.

Examples

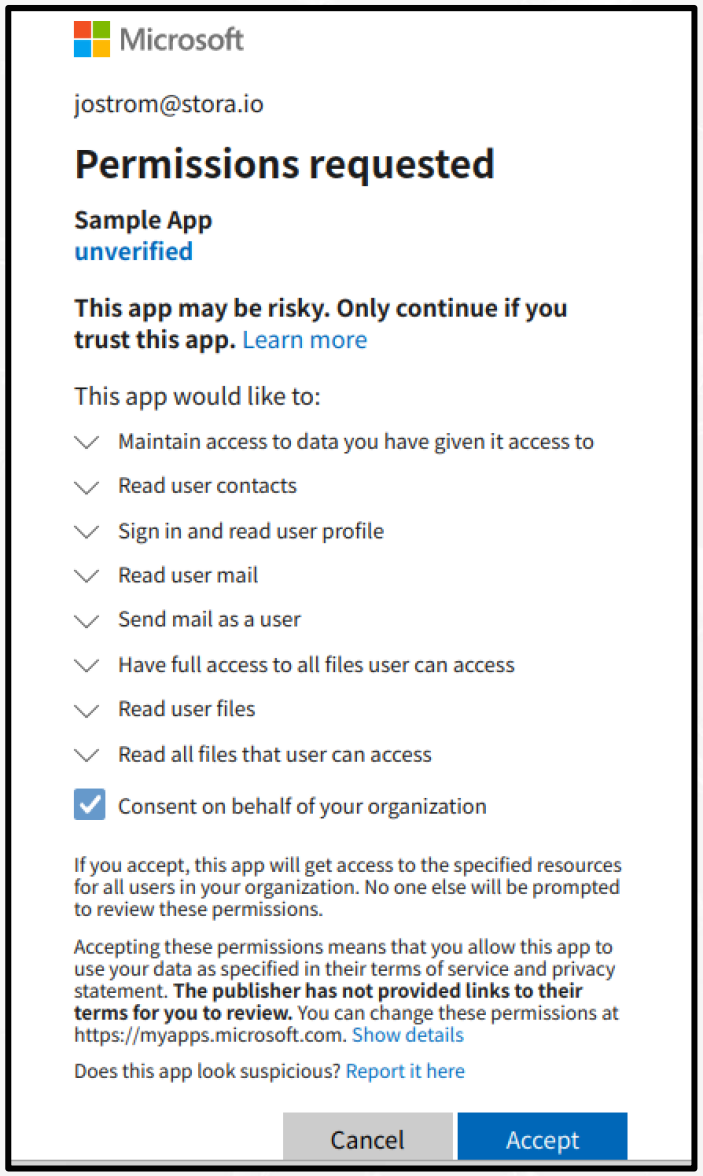
N/A

Detections

Logs

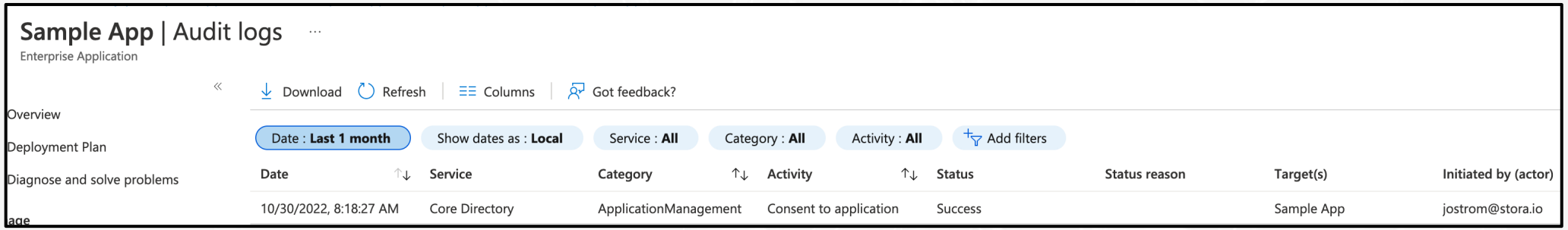
Data Source	Application	Resource	Log Location
Azure Active Directory	N/A	AAD	Log Analytics

User Consents to Application

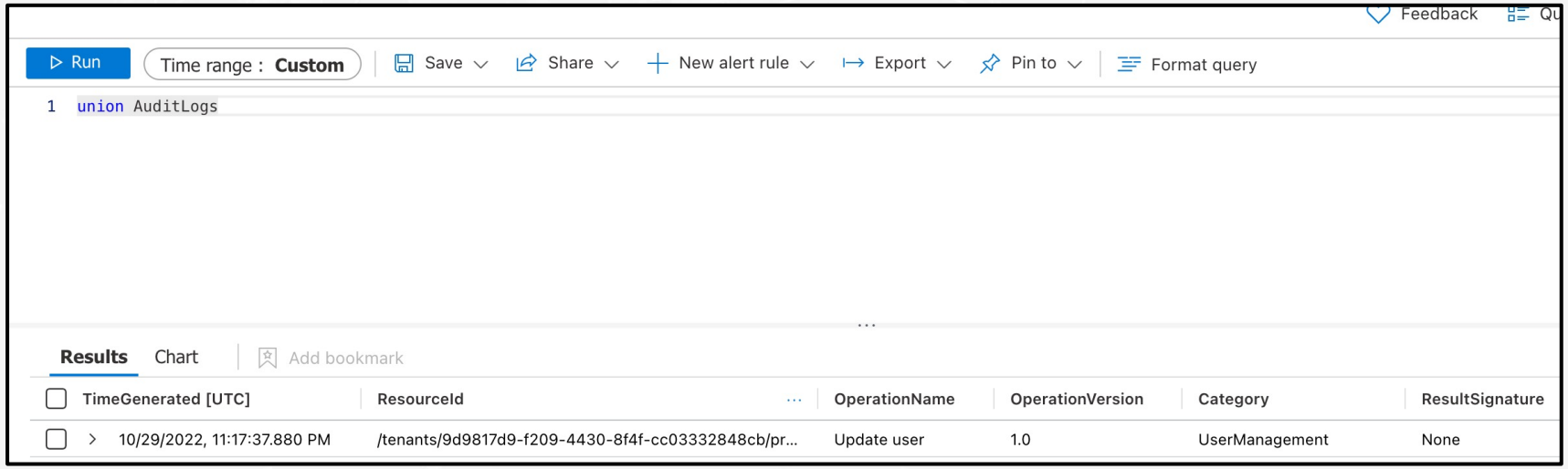


Detection Engineering & Sentinel

- Azure Portal: ***Audit Logs***
- Example KQL: ***union AuditLogs***



Audit logs can trigger alarms on user behavior for application consent to application permissions.



Managed Identity

Managed Identity Lab Generator

- Creates a ***Managed Identity*** lab with an Azure VM Identity automatically created and assigned
- **Key Features**
  - Creates one Azure Virtual Machine with a managed identity assigned
  - Default user assigned identity with role of reader on the subscription
  - Script parameter to change role to owner or contributor
  - Randomly generates one Azure AD user with a role of Virtual Machine Contributor
  - Script parameter to add a system-assigned identity to the Virtual Machine
  - Creates storage account, containers, blobs, shares, and key vaults
- **Example Usage**

```
% python3 managed_identity.py -u rtcfingroup.com -n rtcfin -l eastus -a RTCAdmin -p MyPassword012345 -ua reader -sa
```

Managed Identity Simulations

AZT601.1 - Steal Managed Identity JsonWebToken: Virtual Machine IMDS Request

By utilizing access to IMDS, an attacker can request a JWT for a Managed Identity on an Azure VM if they have access to execute commands on the system.

Resource

Virtual Machine

Actions

- Microsoft.Compute/virtualMachines/write
- Microsoft.Compute/virtualMachines/extensions/*

Examples

PowerShell | Azure Portal

```
powershell.exe -c $a=Invoke-Restmethod -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01"&"resource=https://management.azure.com/' -Method GET -Headers @{Metadata='true'} -UseBasicParsing;$a.access_token
```

Acquire an Identity Access Token

```
PS C:\Users\RTCAdmin>
PS C:\Users\RTCAdmin> Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -Headers @{Metadata="true"} -UseBasicParsing


StatusCode        : 200
StatusDescription : OK
Content           : {"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6IjJaUXBKM1VwYmpBWVhZR2FYRUpsOGxWMFRPSS
                    IsImtpZCI6IjJaUXBKM1VwYmpBWVhZR2FYRUpsOGxWMFRPSSJ9.eyJhdWQiOiJodHRwczovL21hbmFnZW1lbnQuYXp1cmUuY29t
                    Ly...
RawContent        : HTTP/1.1 200 OK
                    Content-Length: 1709
                    Content-Type: application/json; charset=utf-8
                    Date: Sun, 30 Oct 2022 08:20:41 GMT
                    Server: IMDS/150.870.65.797

                    {"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSU...
```

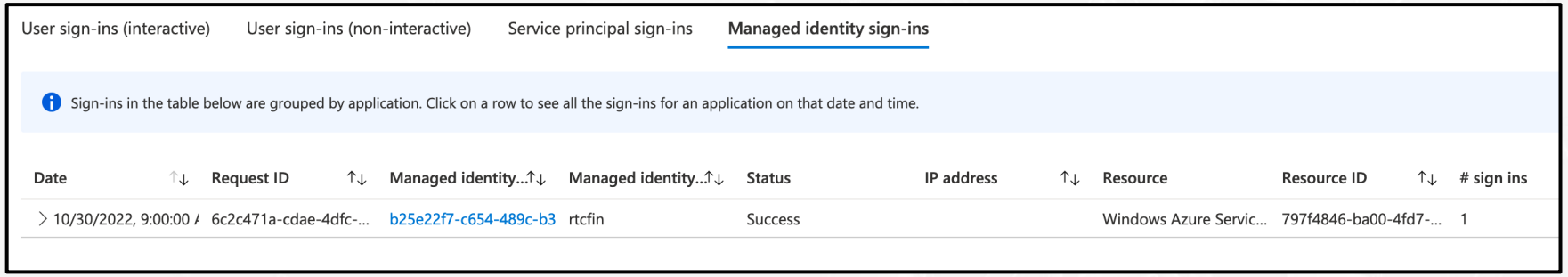
An attacker with RDP access can acquire the Identity's access token or use az vm run command.

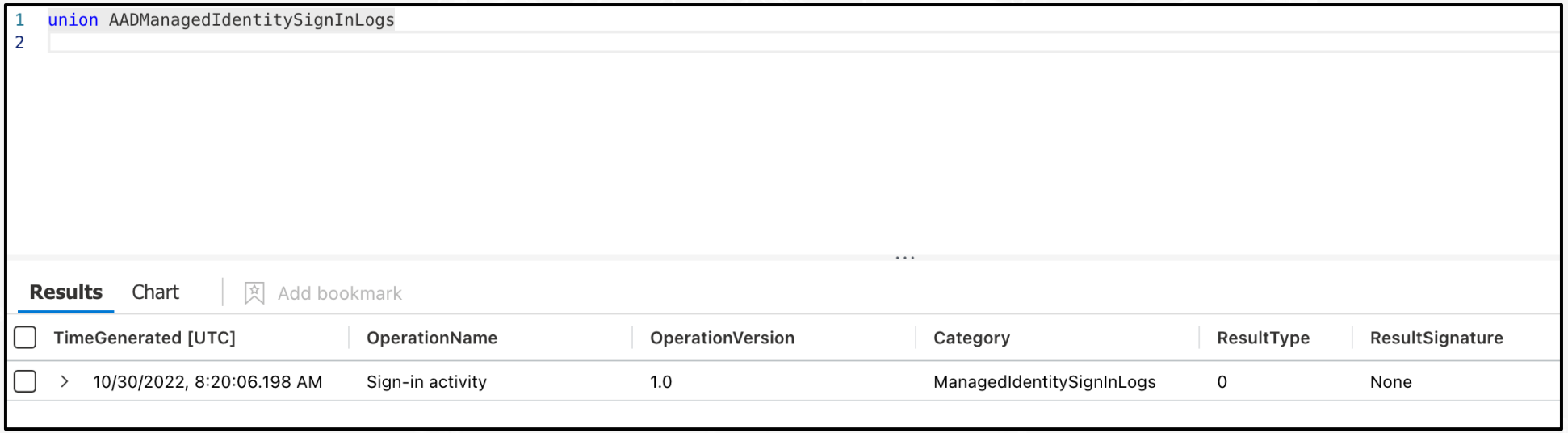
An attacker can sign into Azure AD using the VM's managed identity. Sign in logs can track this behavior.

```
jason@Azure:~$ spID=$(az resource list -n rtcfin --query [*].identity.principalId --out tsv)
jason@Azure:~$ echo $spID
9235040a-a847-43c9-96cf-d84427dbfe85
jason@Azure:~$
jason@Azure:~$ az login --identity
[
  {
    "environmentName": "AzureCloud",
```

Detection Engineering & Sentinel

- Azure Portal: ***Managed Identity sign-ins***
- Example KQL: ***union AADManagedIdentitySignInLogs***





Storage Lab

Storage Lab Generator

- Creates an ***Azure Storage*** security lab
- **Azure Resources Created**
  - Storage account
  - Three storage containers with different permission levels (private, blob, container)
  - Two azure shares
  - Upload of fake, sensitive files to shares and containers as blobs
  - Key vault with secrets, private keys, and certificates
- **Example Usage**

```
% python3 storage.py
```

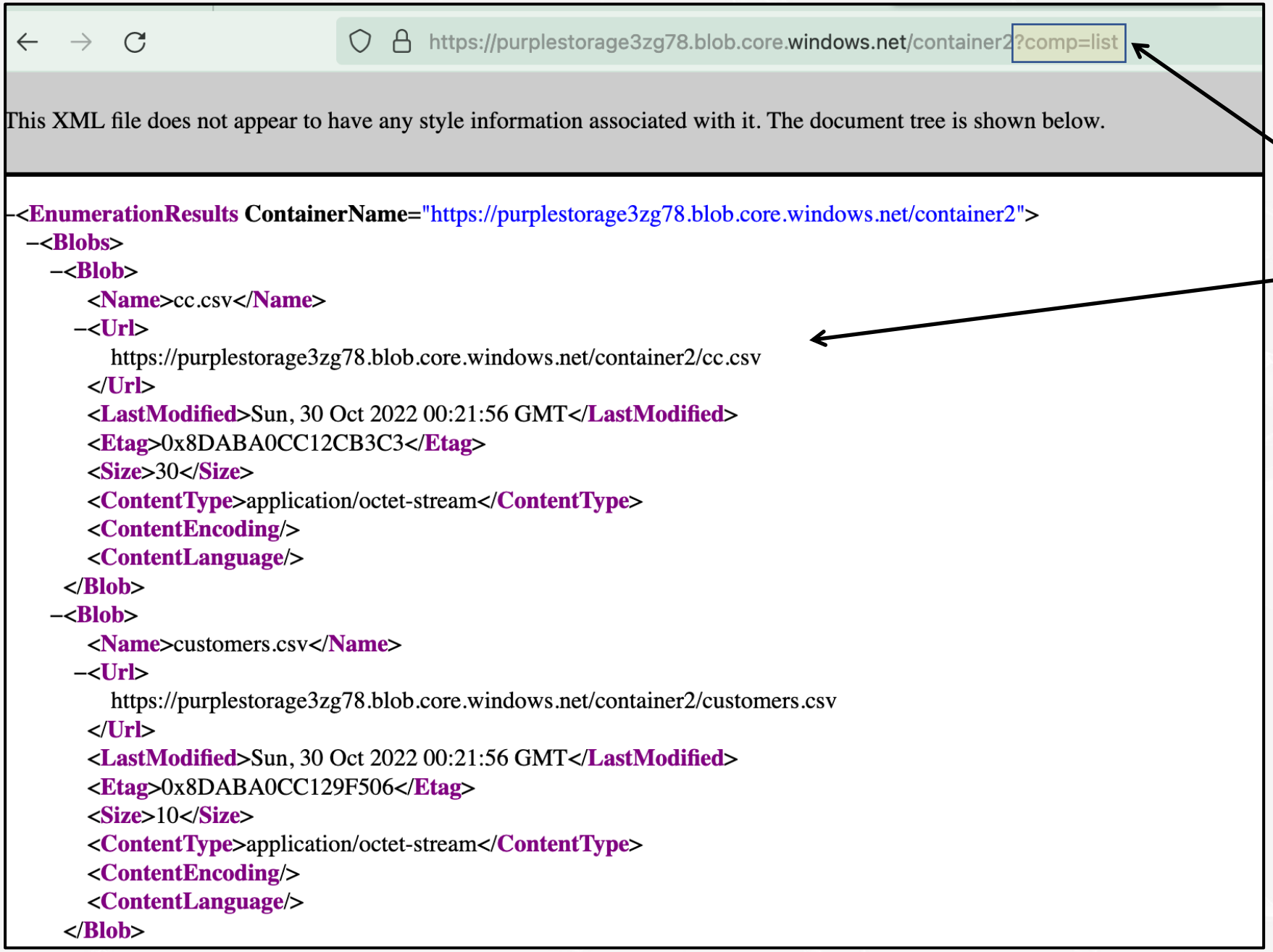
Security Testing for Anonymous Blob Access

Name	Last modified	Public access level
container1	10/30/2022, 9:21:55 AM	Blob
container2	10/30/2022, 9:21:55 AM	Container
container3	10/30/2022, 9:21:55 AM	Private

Name	Modified	Access tier	A
cc.csv	10/30/2022, 9:21:56 AM	Hot (Inferred)	
customers.csv	10/30/2022, 9:21:56 AM	Hot (Inferred)	
finance.xlsx	10/30/2022, 9:21:56 AM	Hot (Inferred)	
hr.xlsx	10/30/2022, 9:21:56 AM	Hot (Inferred)	

The ***Container*** access level enables indexing of files.

The simulation lab uploads sample files.



Appending a string of "***?comp=list***" to a storage container with "container" access level will reveal all of the files in the target container.

ADFS Federation

ADFS Lab Generator

Credit to Roberto Rodriguez and his Azure Simuland tool. The ADFS Lab Generator was inspired by Simuland.

- Creates an ***Active Directory Federation Services (ADFS)*** lab.
- **Key Features**
  - Deploys an ADFS server joined to a created Domain Controller with AD Domain
  - Deploys Azure Sentinel
  - Deploys Azure monitoring agent on ADFS server and ADFS Audit log best practices for shipping logs into Sentinel
  - Supports self-signed ADFS certificate deployment (Default)
  - Supports trusted CA certificate import using optional script parameters
  - Implements ADFS Audit log best practices
- **Example Usage**

```
% python3 adfs.py --trusted_cert adfs.pfx --pfx_password password
```

ADFS Security Auditing logs

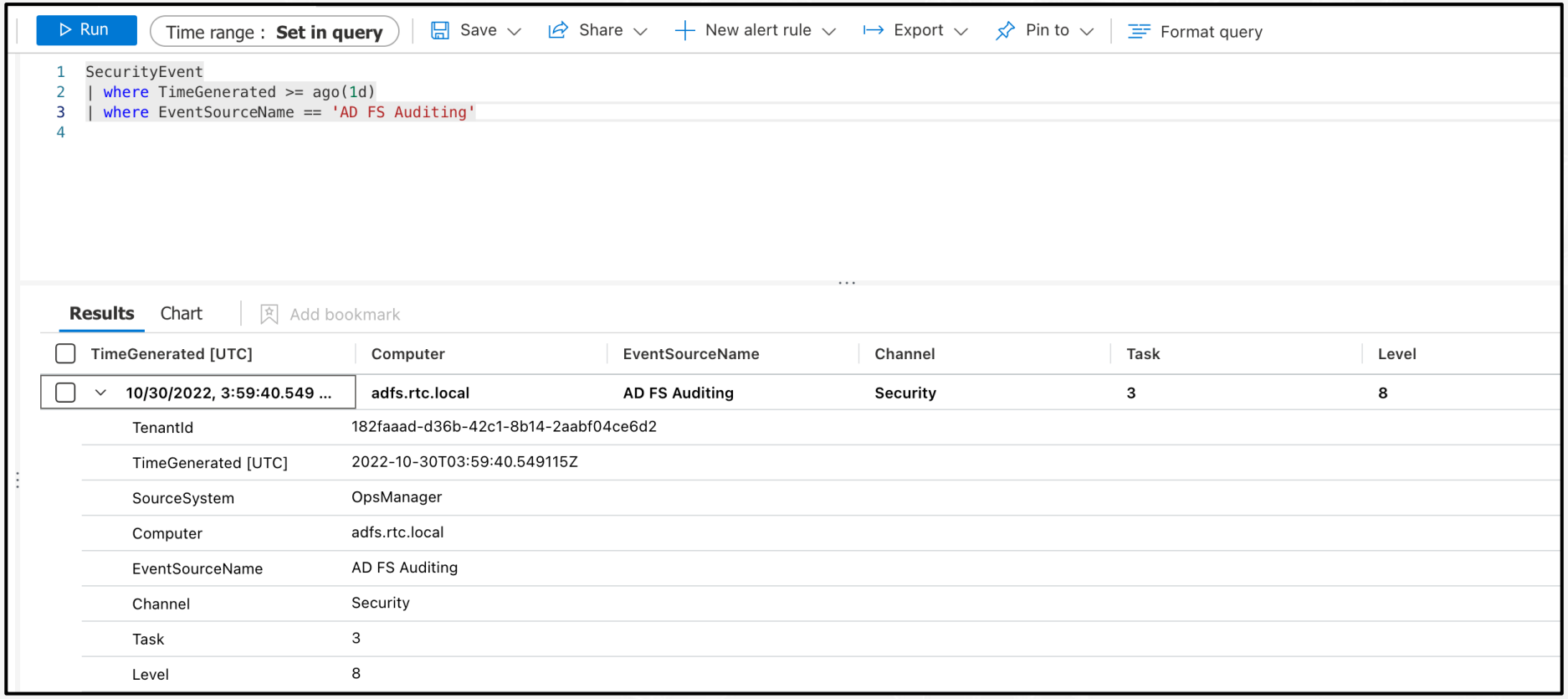
- Generate authentication events for success and failure

Credit to Roberto Rodriguez for his blog on enabling ADFS Security Auditing.

Detection Engineering & Sentinel

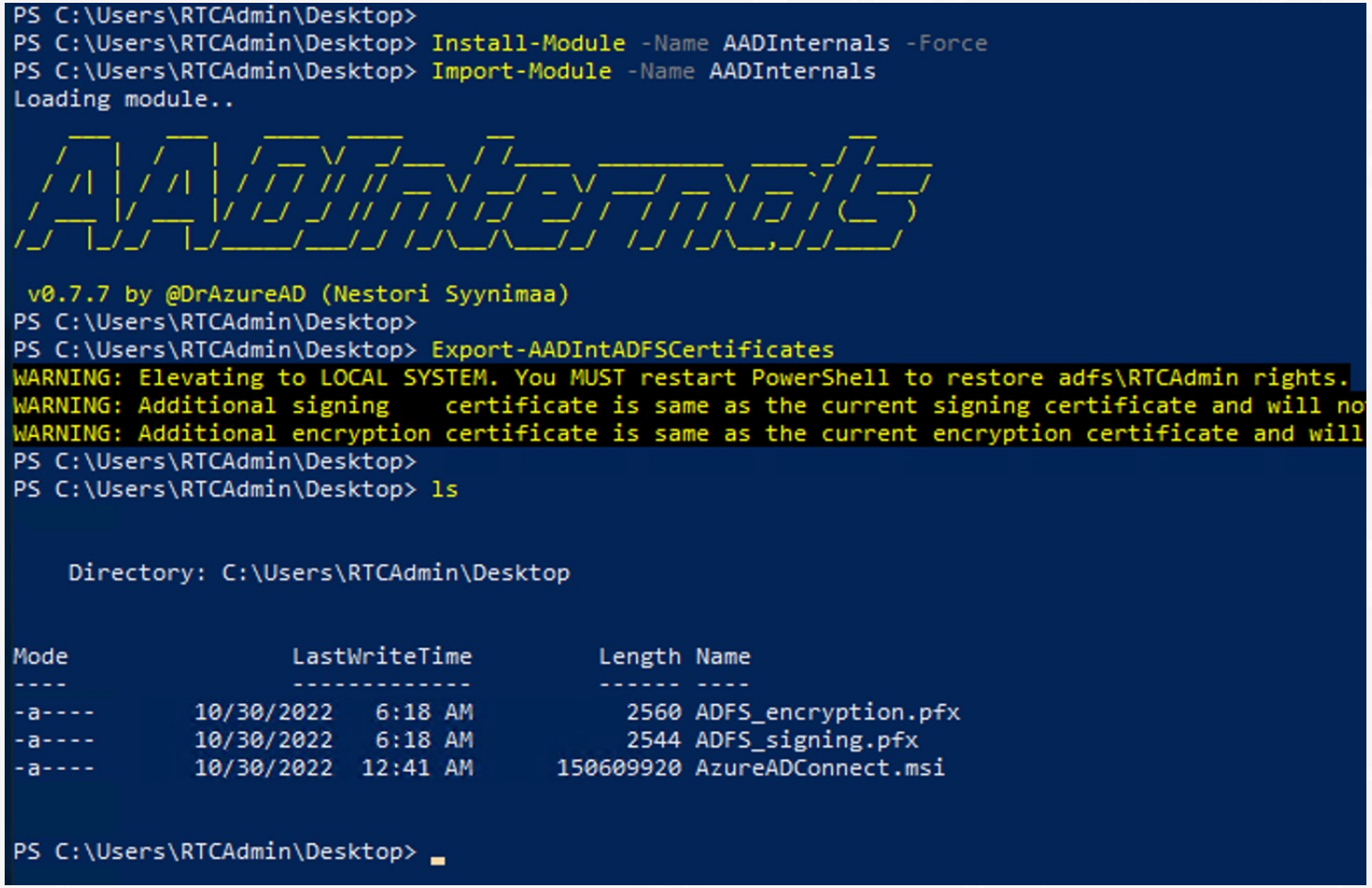
- Example KQL

```
SecurityEvent
| where TimeGenerated >= ago(1d)
| where EventSourceName == 'AD FS Auditing'
```



Extract Token Signing Certificate Used in ADFS

- *Golden SAML*



Credit to Dr. Nestori Syynimaa, his Golden SAML research, and his AADInternals tool.

https://aadinternals.com

Instrument Sysmon Log Detections with KQL

- ***KQL:*** union Event

```
1  union Event
```

Results | Chart | Add bookmark

TimeGenerated [UTC]	Source	EventLog	Computer	EventCategory
10/30/2022, 2:10:45.911 ...	Microsoft-Windows-Sysmon	Microsoft-Windows-Sysmo...	adfs.rtc.local	13

TenantId	182faaad-d36b-42c1-8b14-2aabf04ce6d2
SourceSystem	OpsManager
TimeGenerated [UTC]	2022-10-30T02:10:45.9118749Z
Source	Microsoft-Windows-Sysmon
EventLog	Microsoft-Windows-Sysmon/Operational
Computer	adfs.rtc.local
EventCategory	13
EventLevel	4
EventLevelName	Information
UserName	NT AUTHORITY\SYSTEM

Azure AD Join

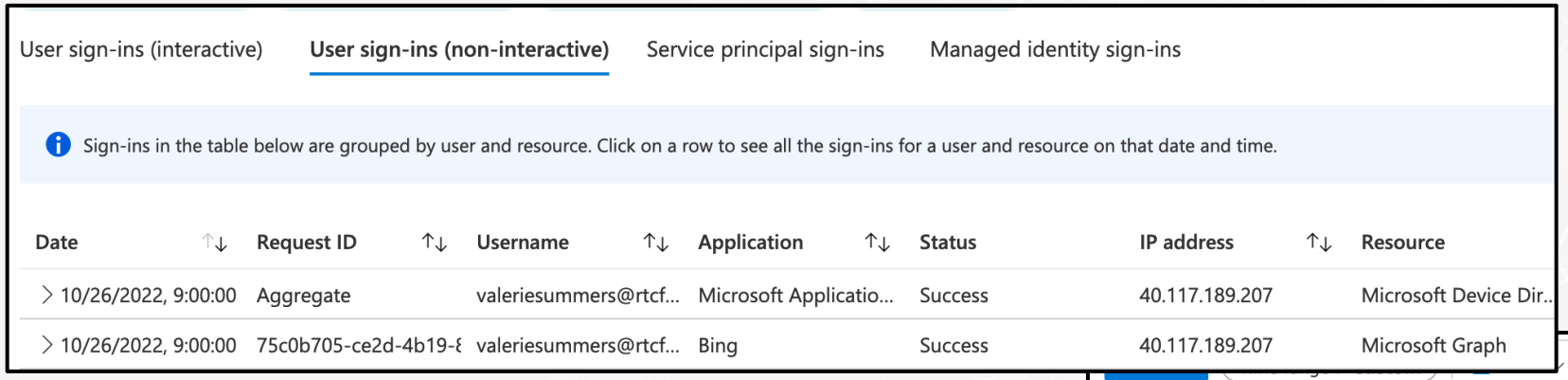
Azure AD Join Lab Generator

- Creates an ***Azure AD Join*** security lab with Azure Virtual Machines joined to Azure Active Directory.
- **Key Features**
  - Creates optional number of Windows 10 Endpoints and automatically joins them to Azure Active Directory
  - Deploys a system assigned Identity and optional user assigned identity on all Azure VMs
  - Creates simulated Azure AD users with role of Virtual Machine Administrator Login and Virtual Machine User Login
- **Example Usage**

```
% python3 aadjoin.py --endpoints 1 -u rtcfingroup.com
```

Detection Engineering & Sentinel

- Azure Portal: ***User sign-ins (non-interactive)***
- Example KQL: ***union AADNonInteractiveUserSignInLogs***
- Connection IP addresses appear sourced from an Azure data center IP



Purple Teaming Cloud Identity

PurpleCloud Use Cases

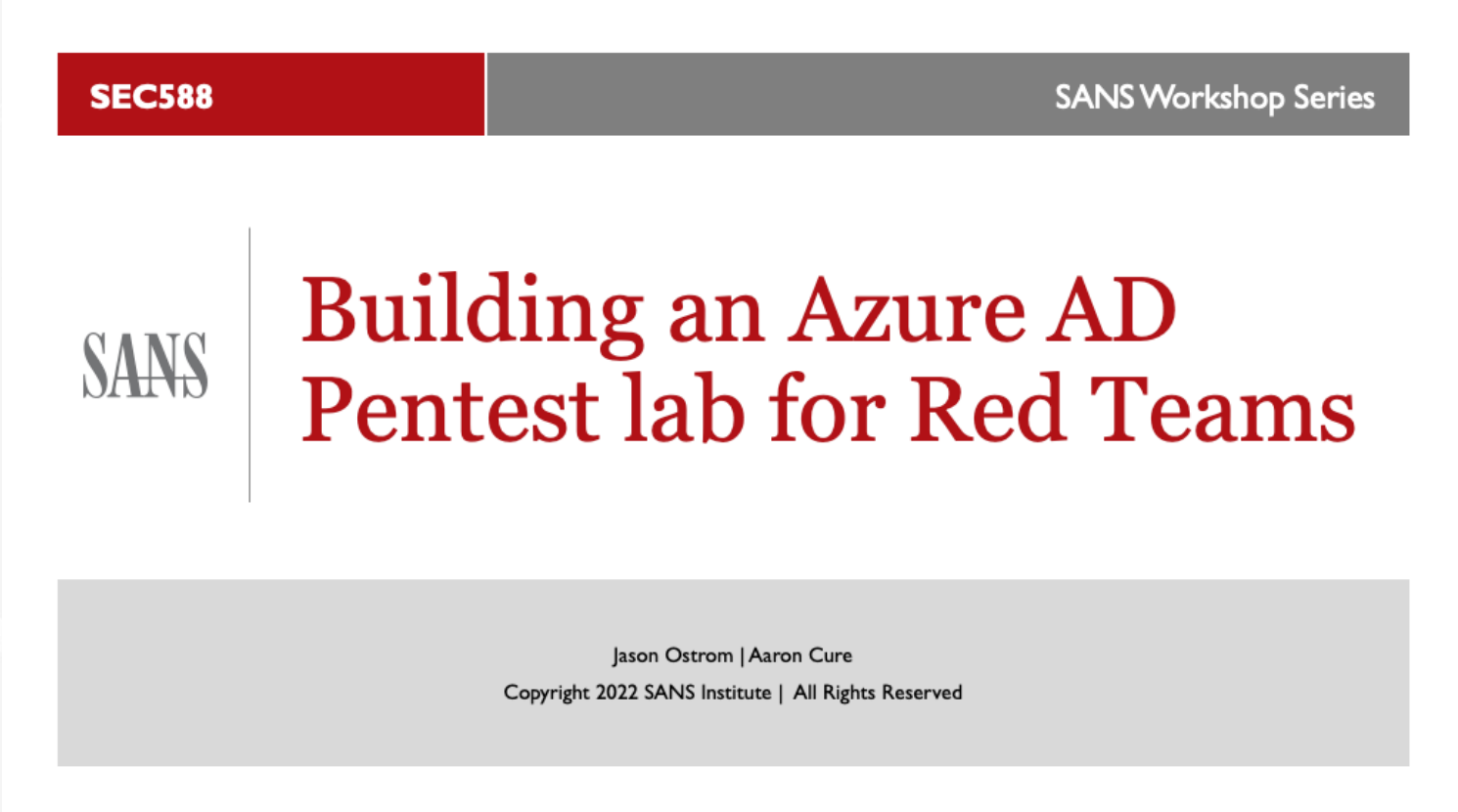
- ***PurpleCloud*** enables anyone to auto-create an Azure AD security lab for a variety of use cases:
- Mapping Azure Threat Research Matrix to required security controls
- Create an Azure AD Lab mirroring customer tenant, to practice privilege escalation
- App Consent phishing campaigns + Social Engineering
- Create the lab with exact number of Azure AD users, to practice recon tooling, username enumeration, password spraying behavior
- Blue teams to instrument Azure sign-in logs correctly + Detection Engineering + Purple Teaming exercises
- R&D security research for new vulnerabilities or techniques

PurpleCloud can be used for Purple Teaming exercises

Demo

Free SANS Workshop: Building an Azure AD Pentest lab for Red Teams

- **Free guided scenario vulnerability lab with playbook**
- **Registration URL:** https://www.sans.org/webcasts/sans-workshop-building-azure-pentest-lab-red-teams



Thank you for attending SANS Pen Test Hackfest!

- Thank you for attending my session!

- **Contact Information**
  - Jason Ostrom
  - Twitter: @securitypuck
  - Email: jostrom@stora.io
  - Mastodon: @securitypuck@infosec.exchange

- **SANS Offensive Ops Discord:** Consider joining the SANS Offensive Ops Discord server.
  - https://sansurl.com/discord
  - Find us on #sec588

References

References

- [1] “Azure Privilege Escalation via Service Principal Abuse”
  - https://bit.ly/3OUgAhN
- [2] “Azure AD privilege escalation – Taking over default application permissions as Application Admin”
  - https://bit.ly/3R0jByC
- [3] “About Detection Engineering”
  - https://cyb3rops.medium.com/about-detection-engineering-44d39e0755f0
- [4] “Azure Threat Research Matrix”
  - https://microsoft.github.io/Azure-Threat-Research-Matrix/
- [5] “Lateral Movement With Managed Identities of Azure Virtual Machines”
  - https://m365internals.com/2021/11/30/lateral-movement-with-managed-identities-of-azure-virtual-machines/
- [6] ”Azure Simuland”
  - https://simulandlabs.com/README.html
- [7] “Enabling AD FS Security Auditing and Shipping Event Logs to Microsoft Sentinel”
  - https://techcommunity.microsoft.com/t5/microsoft-sentinel-blog/enabling-ad-fs-security-auditing-and-shipping-event-logs-to/ba-p/3610464