509.3
Microsoft Azure

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

509.3
Microsoft Azure

SANS

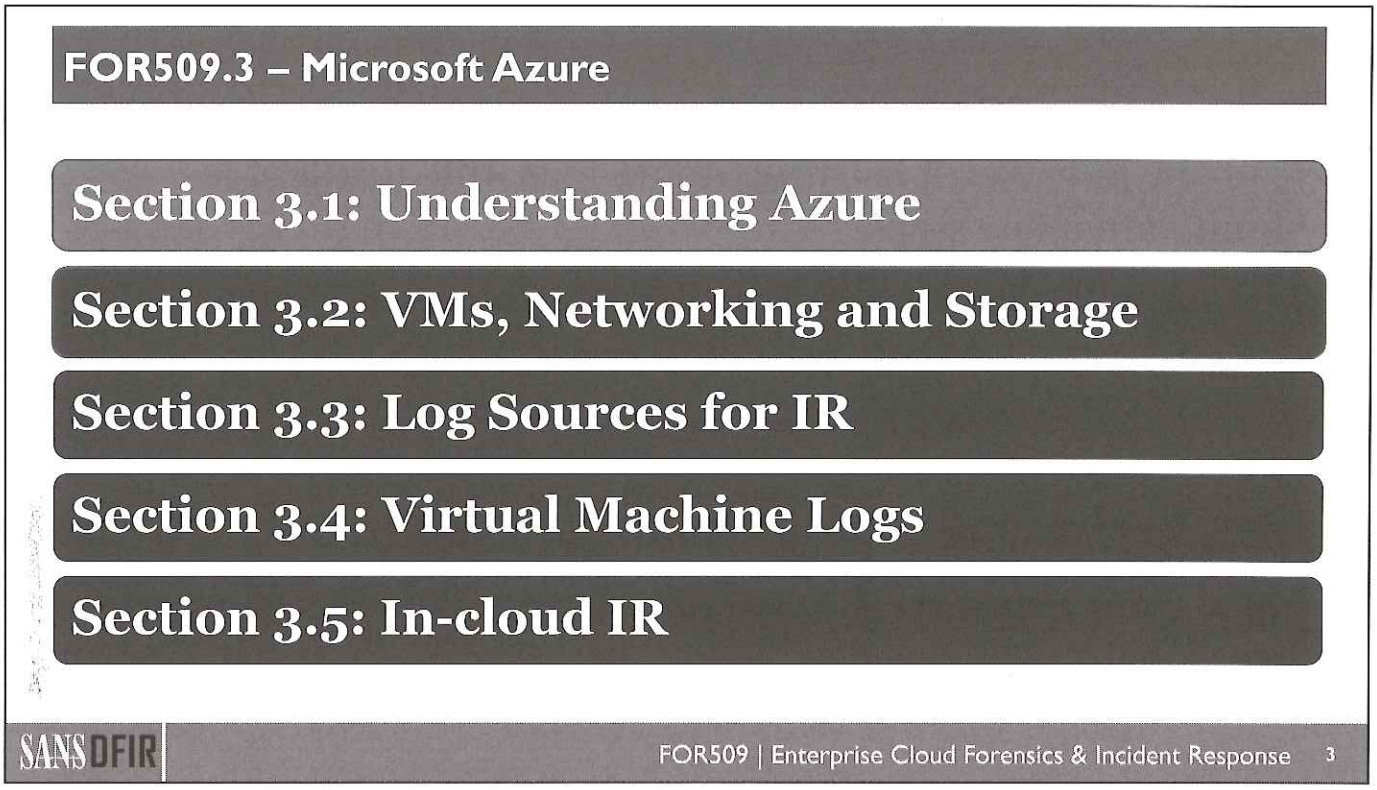
THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

FOR509.3 | Enterprise Cloud Forensics and Incident Response

Microsoft Azure

This page intentionally left blank.

This page intentionally left blank.

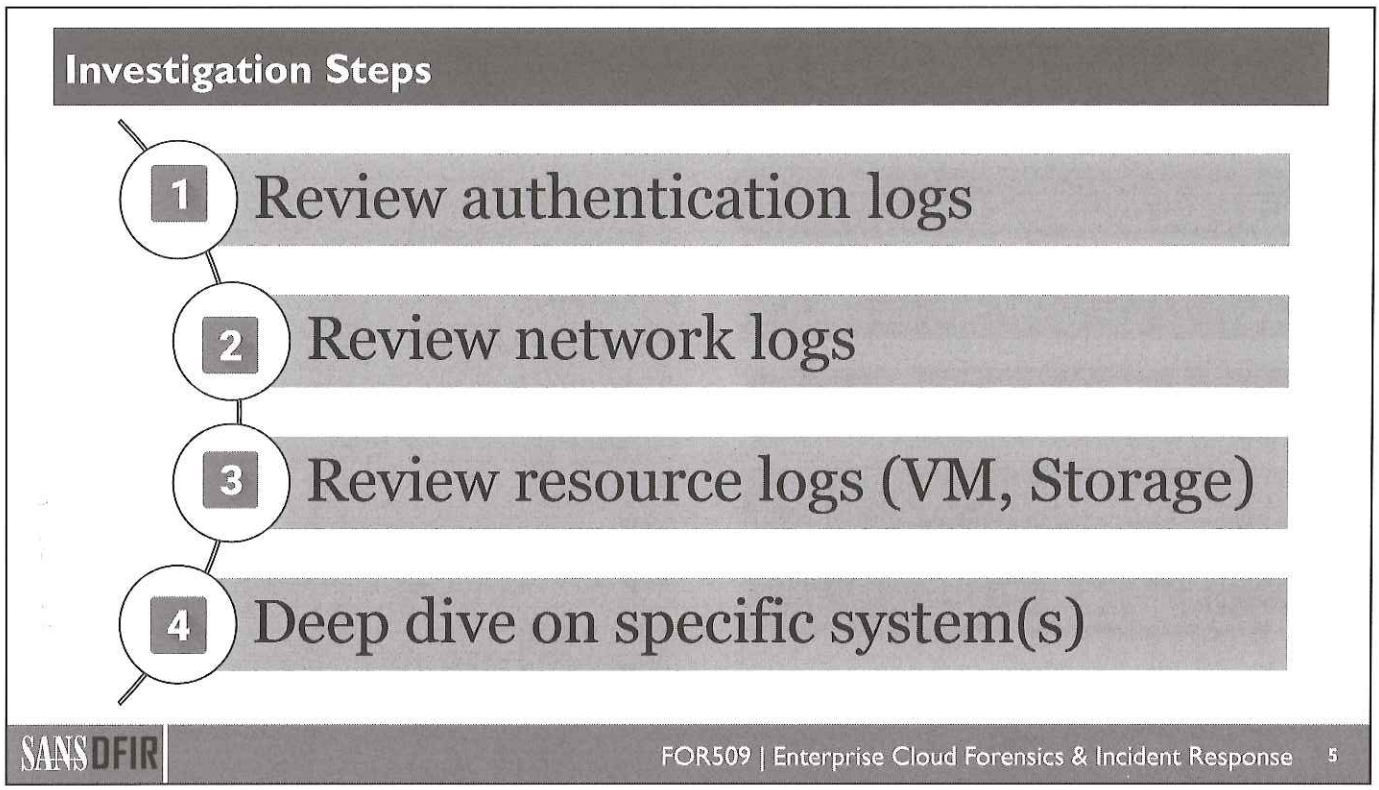


This page intentionally left blank.

Sample Incident: Pymtechlabs

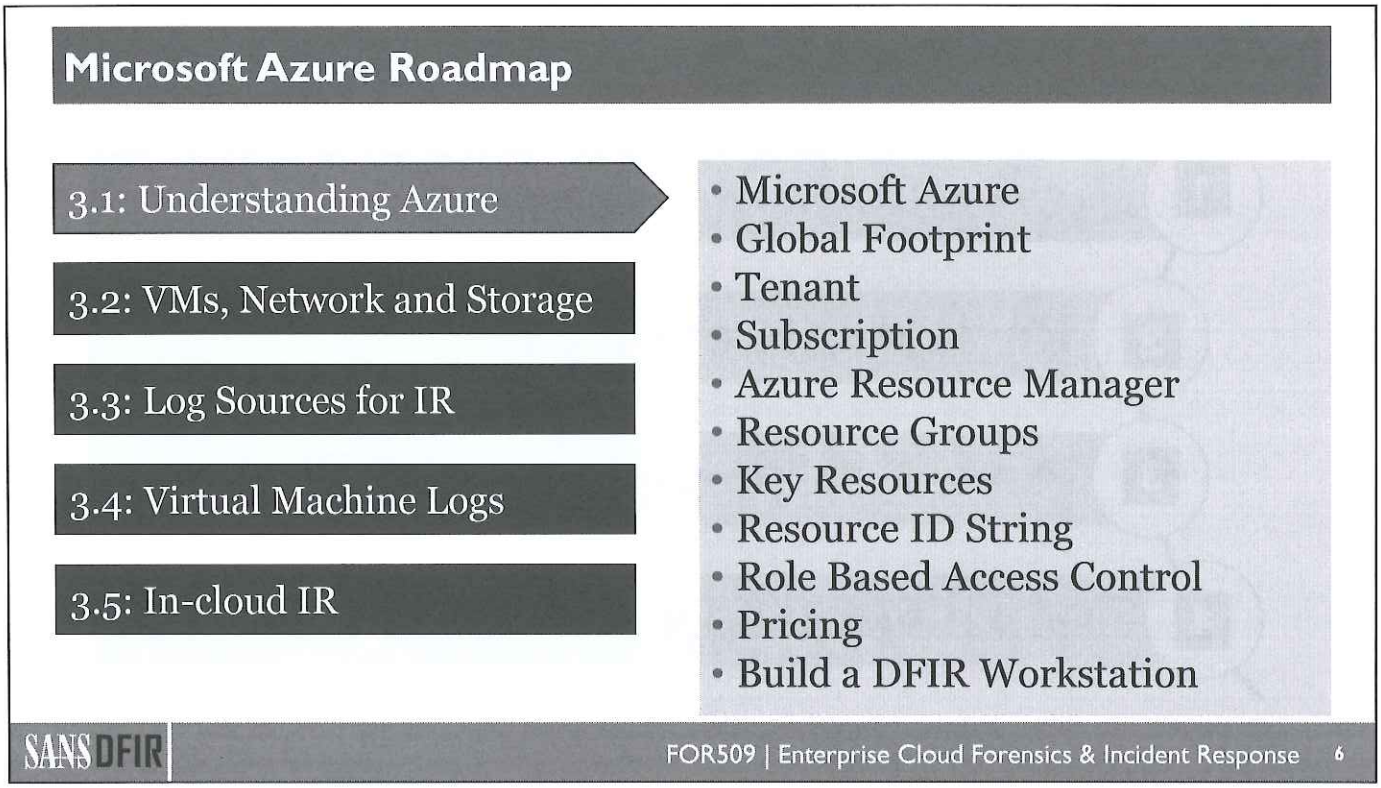
- For the purpose of learning Azure logs, this incident is limited to a few steps
- Focus on understanding the uniqueness of Azure logs
- Question to take back after this class: are Azure logs configured correctly and available to me in my environment?
- Logs can be reviewed in different ways inside the Azure console, but most companies will use a SIEM
- Labs leverage SOF-ELK, and all relevant data has been pre-loaded

To further the educational experience, we have created a simple scenario that will facilitate learning about the different logs. All the data has been imported into SOF-ELK to facilitate your analysis.

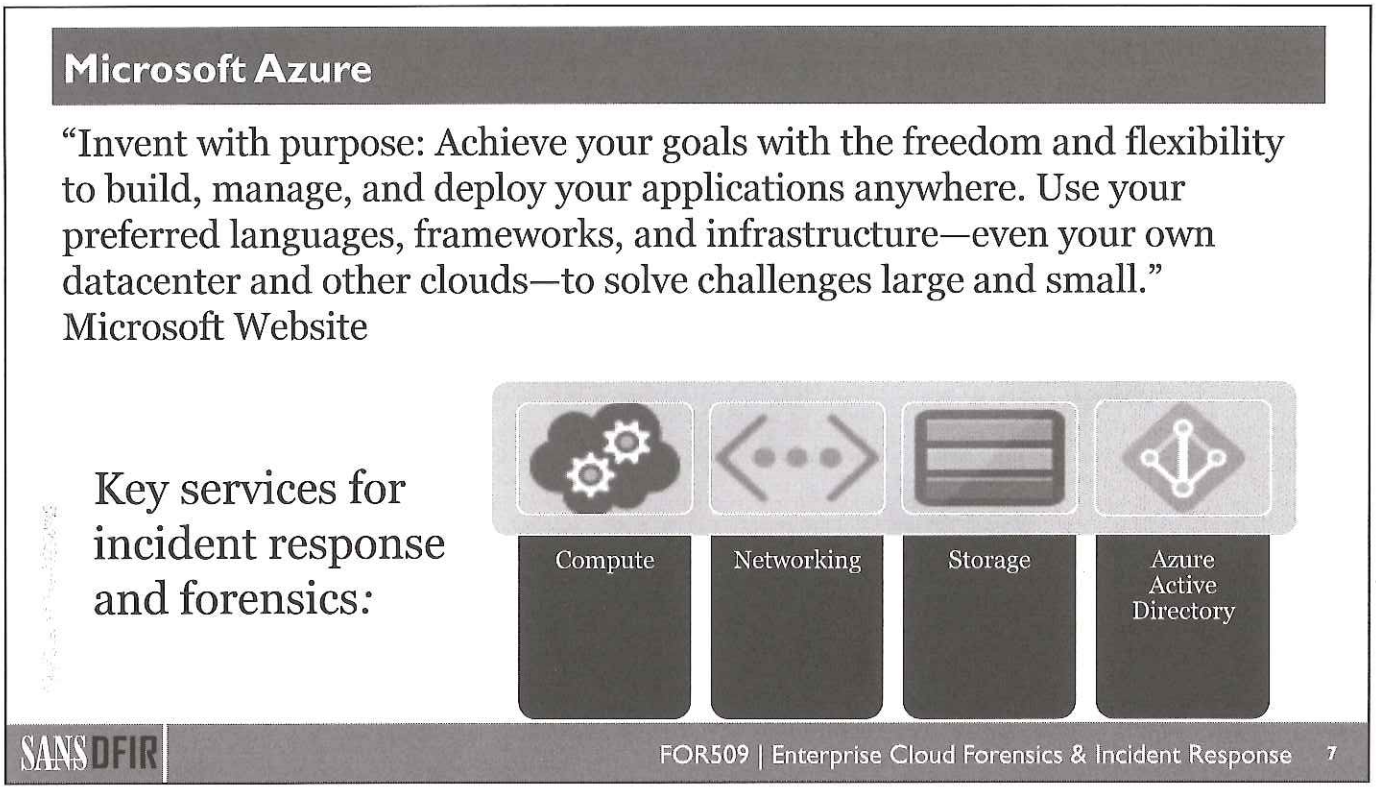


While every cloud investigation will be a bit different, in general we recommend the following steps:

1. **Review the global log sources**. For Azure, this is mainly the authentication logs from Azure Active Directory. There is an audit log for tenant-wide operations which should also be reviewed if called for (usually, that would indicate big trouble since that log entails changes to the tenant itself).
2. **Review network logs**. These might come from the Azure Network Security Group or third-party firewalls depending on your environment.
3. **Review resource logs** such as virtual machines and storage logs. Virtual machine logs would indicate if VMs have been added or removed. Storage logs would show if data has been written or read. There are many other resource logs to potentially review if they have been properly configured (which unfortunately is rarely the case).
4. Finally, if needed, employ more traditional forensic methods and **deep dive on a specific system**. This can be done in-cloud, therefore minimizing cost and speeding up your investigation.



This page intentionally left blank.



Microsoft describes Azure with the tagline "Invent with purpose". They further summarize Azure by saying "Achieve your goals with the freedom and flexibility to build, manage, and deploy your applications anywhere. Use your preferred languages, frameworks, and infrastructure—even your own datacenter and other clouds—to solve challenges large and small."[1]

As stated by Microsoft, Azure is an eco-system of resources that can be molded to achieve anything you want. It's a large eco-system so we will need to focus on the elements you are most likely to encounter during an incident response or forensic investigation. To that purpose, we will investigate where useful logs are stored, how to access them, and how to interpret them.

We will focus on four products: compute, networking, storage, and Azure active directory.

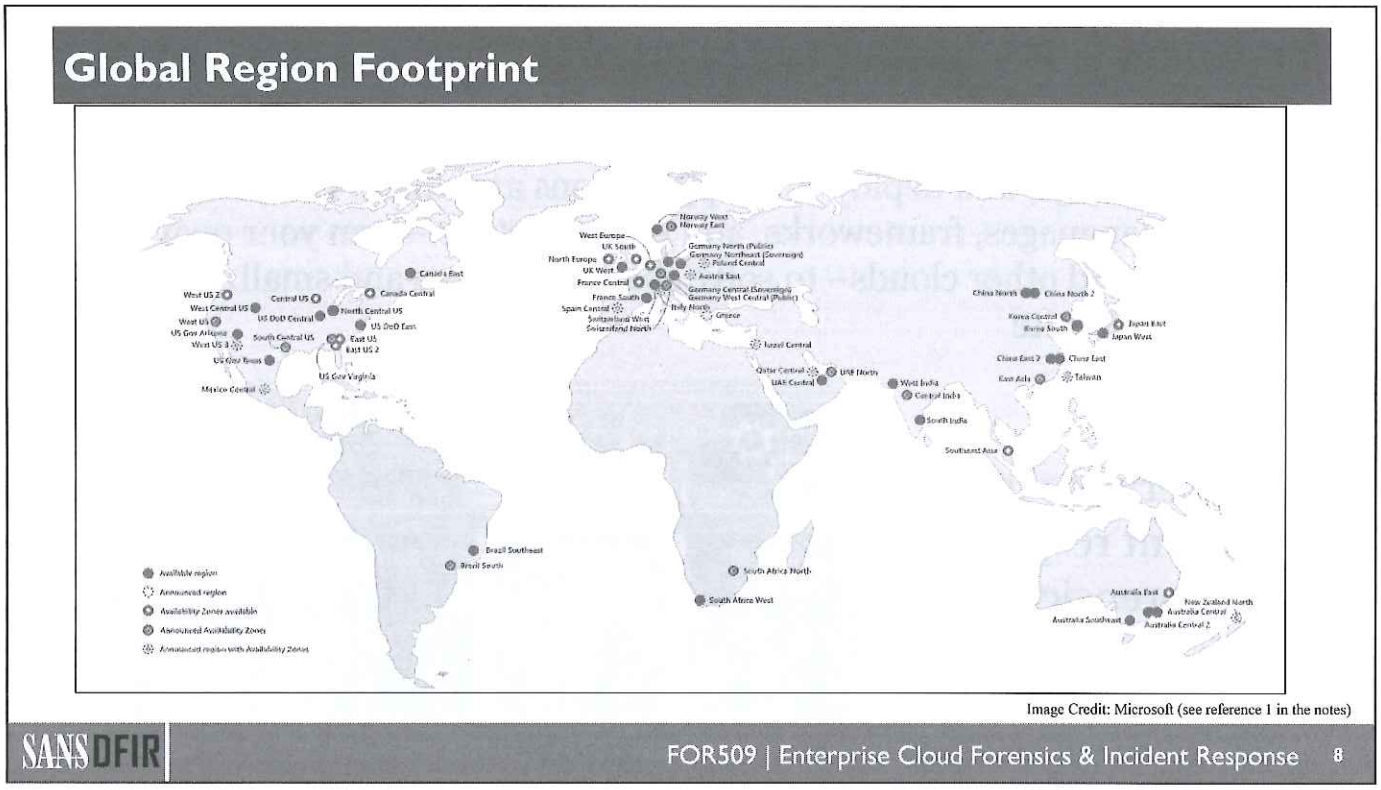
Before we can start, we must cover some Azure fundamentals, so everyone has a good foundation. We will discuss the regions where you may find Azure services as well as concepts such as availability zones, tenants, subscriptions, Azure resource manager, and role-based access control. We will finish this section with a discussion on pricing as it may impact how you conduct your investigations.

All icons courtesy of Microsoft Azure Cloud and AI Symbol/Icon Set.[2]

References:

[1] https://for509.com/azureintro

[2] https://for509.com/azureicons



When defining resources in Azure, you must select a region to host these resources. This allows you to meet specific data residency and compliance requirements. As you can see from the map, Azure has numerous regions to choose from.[1] What constitutes a region?

Microsoft defines a region as "a set of datacenters deployed within a latency-defined perimeter and connected through a dedicated regional low-latency network".[2]

Let's breakdown Microsoft's definition:

- "a set of datacenters" means that each region has more than one datacenter.
- "latency defined perimeter" means that the datacenters are physically close to each other.
- "regional low-latency network" means that they are connected via fiber.

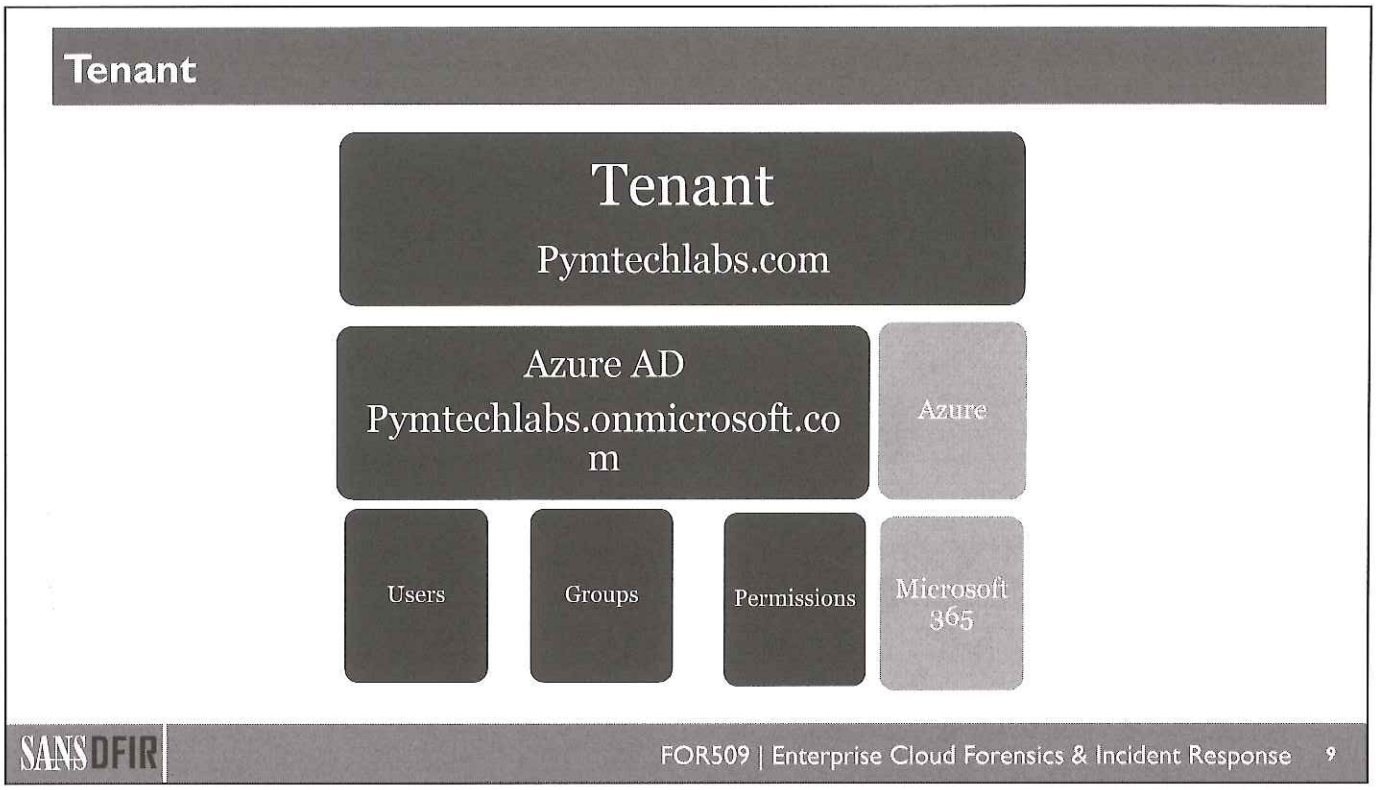
Microsoft's definition of a region simply means that each region has two or more data centers close to each other connected via fiber. Furthermore, each region is paired with another. For example, East US is paired with West US. Each pair is within the same geography.

One very important concept for us is that data stays within a region unless explicitly transferred via a global service. This is done to ensure data residency. It also means that when you perform your investigation, you must be careful where you create your investigation resources.

While unlikely to affect you for your investigation purposes, some services are only available in certain regions. However, core services are available in all regions. Core services is what we will use to facilitate our investigations.

References:

[1] https://for509.com/azuregeo
[2] https://for509.com/azureregions



To get started in Azure, you must first "build" a tenant which will contain your identities, subscriptions, licenses, and resources. You can think of a tenant as a container.

Your tenant represents a set of services assigned to your organization. Tenants are typically associated with your top-level DNS domain name, for example pymtechlabs.com in this class. Tenants are assigned to a specific geographical location.[1]

For each tenant there is a dedicated Azure Active Directory (also called AAD or Azure AD) instance. Azure AD is critical as it manages users, groups, and permissions for all applications. In that regard, Azure and Microsoft 365 are considered applications.

Azure AD Tenants are globally unique and use the domain 'onmicrosoft.com'. So, for the Pymtechlabs tenant, the Azure AD Tenant is pymtechlabs.onmicrosoft.com.

Users can only belong to a single tenant. They may be guests of other tenants.

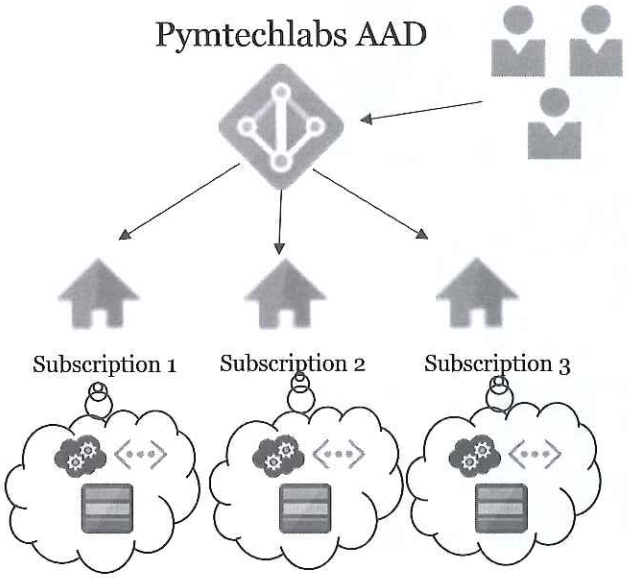
While rare, it's possible to have multiple tenants. This may be the results of merger and acquisitions, or requirements for specific administrative isolation due to local laws.

Each tenant may contain multiple subscriptions which we will discuss in the next slides.

References:

[1] https://for509.com/microsofttenant

Subscriptions



- Organizations/Tenants can have multiple Azure subscriptions
- They all share a single Azure Active Directory for that Tenant
- Users and subscription permissions are defined in AAD
- Unpaid subscription fees will impact all resources within that subscription only

Now that we have our tenant setup, we need to define one or more subscriptions. As stated by Microsoft, "a subscription is an agreement with Microsoft to use one or more Microsoft cloud platforms or services, for which charges accrue based on either a per-user license fee or on cloud-based resource consumption."[1]

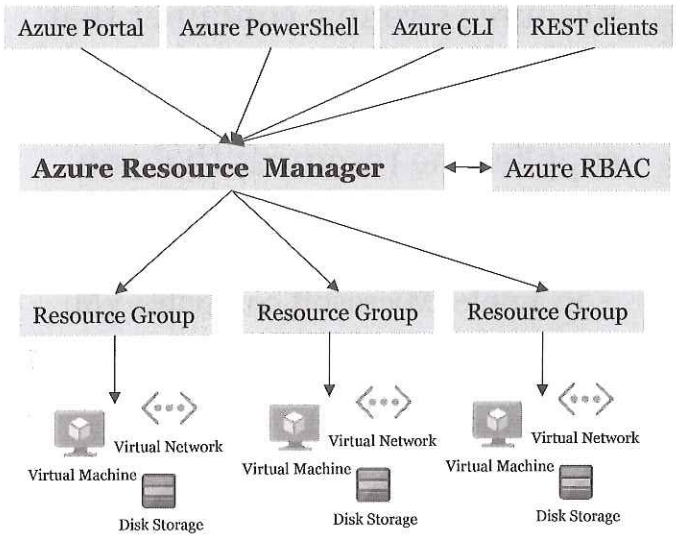
Companies will generally setup numerous subscriptions in order to keep track of charges incurred by different project. This is a very important concept as you will need permissions for each subscription that holds the resource(s) you are investigating.

We will now discuss how resources are created and permissions to these resources managed.

References:

[1] https://for509.com/subscriptions

Azure Resource Manager



- Underlying service for deploying and managing resources in Azure
- Consistent management layer: interfaces with Azure Portal, Azure PowerShell, Azure CLI, and REST clients
- Resource Manager Template (infrastructure as code). Very similar to AWS cloud formation

Before we can discuss virtual networks, virtual machines, and storage, we need to understand how these are deployed and managed. The answer is the Azure Resource Manager which provides a management layer that enables the creation, updating, and deletion of resources.[1]

The advantage of the Azure Resource Manager is that it can take instructions from many different interfaces: Azure Portal, Azure PowerShell, Azure CLI, and REST clients. Yet, the result will be the same irrespective of your choice of interface.

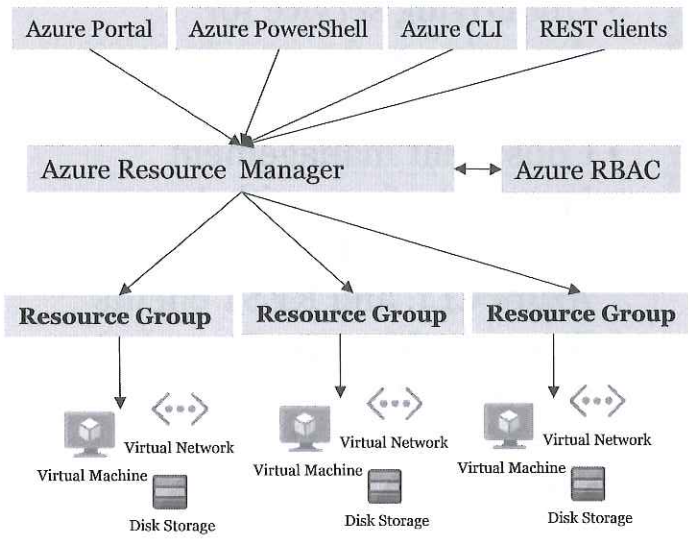
Azure Resource Manager also supports templates (in JSON format) that enables you to deploy resources consistently and repeatedly. This is sometimes referred to as "infrastructure as code". This is very similar to AWS cloud formation.

Azure Resource Manager applies access controls based on Azure role-based access control (RBAC) which we will discuss shortly.

References:

[1] https://for509.com/resourcemanager

Resource Groups



- Resource Groups: container that holds related resources
- Resource Providers: service that supplies the resources that you can deploy and manage through resource manager
  - Example: Microsoft.compute=VM resources
  - Microsoft.storage=Storage account resources
- Resources: VM, Networking, Storage Accounts, etc.

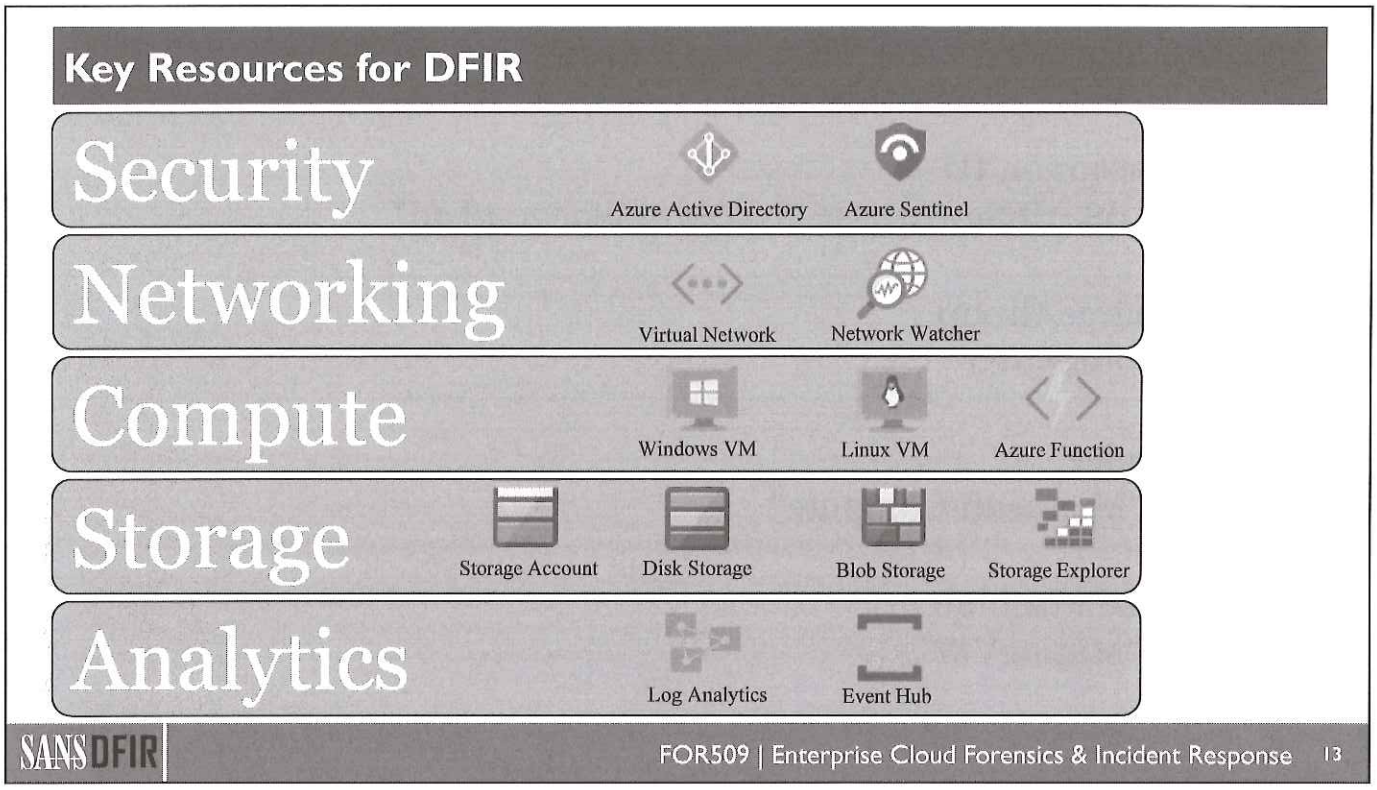
Resources (Virtual machines, networking, storage accounts, databases, etc.) should be grouped inside Resource Groups based on their purpose or commonality. In other words, a resource group is a container that holds related resources. You may have as many resource groups as you wish.

One of the main advantages of resource groups is that you can deploy, update, or delete all resources contained in that resource groups at once. You can also set role-based access for each resource group.

Investigative hint: ideally you will be granted permissions at the subscription level. However, you could encounter a situation where you are only granted permission to a specific resource group. Be aware of this limitation as you will have a very narrow view of the infrastructure being used. This is not a good situation, and you should attempt to get higher level permissions to have a complete view for your investigation.

A resource provider is a service that supplies the resources that you can deploy and manage through the resource manager. For example, if you want to deploy a virtual machine, the Microsoft.compute resource provider will be invoked. Similarly, a storage account will invoke the Microsoft.storage resource provider.

The two items you will interact with the most are resource groups and resources. It's important to know about Azure Resource Manager and Resource Providers to have a complete picture, but they perform their jobs in the background.



Microsoft Azure offers hundreds of products.[1]

For the purposes of incident response and forensics we will focus on just a few products:

- Compute
  - Virtual Machines: Supports Windows and Linux virtual machines
  - Azure Functions: Serverless solution to implement compute-on-demand
- Storage
  - Disk Storage: Persistent storage from virtual machines
  - Blob Storage: REST-based object storage for unstructured data
  - Storage Accounts: Container for disk and blob storage
  - Storage Explorer: View and interact with Azure storage resources
- Networking
  - Virtual Networks: Provisioned private networks
  - Network Watcher: Network performance monitoring and diagnostics
- Security
  - Azure Active Directory: Identity management
  - Azure Security Center: Unified security management
  - Azure Sentinel: cloud-native SIEM and intelligent security analytics
- Analytics
  - Log Analytics: collect, search, and visualize logs
  - Event Hubs: real-time data ingestion service

References:

[1] https://for509.com/azureproducts

Azure Resource ID Definitions

- Subscription ID
  "d841fb8e-c0c7-46fd-ad91-3689e704d1fd"
- Resource Group
  "Research"
- Provider
  "Microsoft.Compute"
- Virtual Machine
  "MiningVM"

Now that we understand the hierarchy that defines an Azure resource, let's look at the internal notation.

- Subscription ID: Globally Unique Identifier (GUID) that belongs to your tenant.
- Resource Group: user-generated name
- Provider: name of the service that supplies the resources that you can deploy and manage through resource manager
- Resource: specified by its type and the name given by the user

In the next slide, we will see how Azure puts everything together in a single URI.

Azure Resource ID Strings

- Resource ID string for the VM
/subscriptions/d841fb8e-c0c7-46fd-ad91-3689e704d1fd/resourceGroups/Research/providers/Microsoft.Compute/virtualMachines/MiningVM

- Resource ID string for the OS disk
/subscriptions/d841fb8e-c0c7-46fd-ad91-3689e704d1fd/resourceGroups/Research/providers/Microsoft.Compute/disks/MiningVM_disk1_213aa18e15cb44a68812d435fff3c508

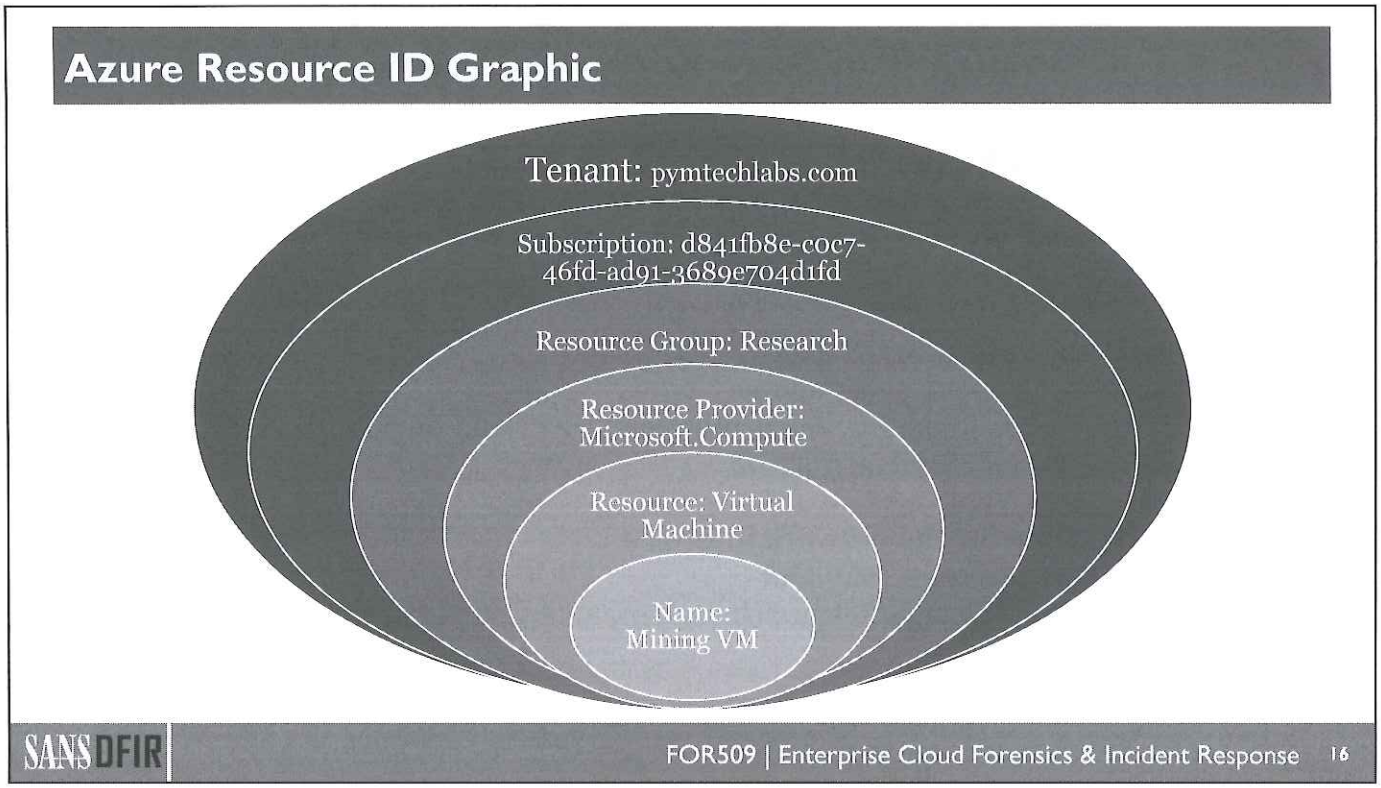
- Resource ID string for the network interface
/subscriptions/d841fb8e-c0c7-46fd-ad91-3689e704d1fd/resourceGroups/Research/providers/Microsoft.Network/networkInterfaces/miningvm106

Every item in Azure has an associated Universal Resource Identified (URI) that follows the format:

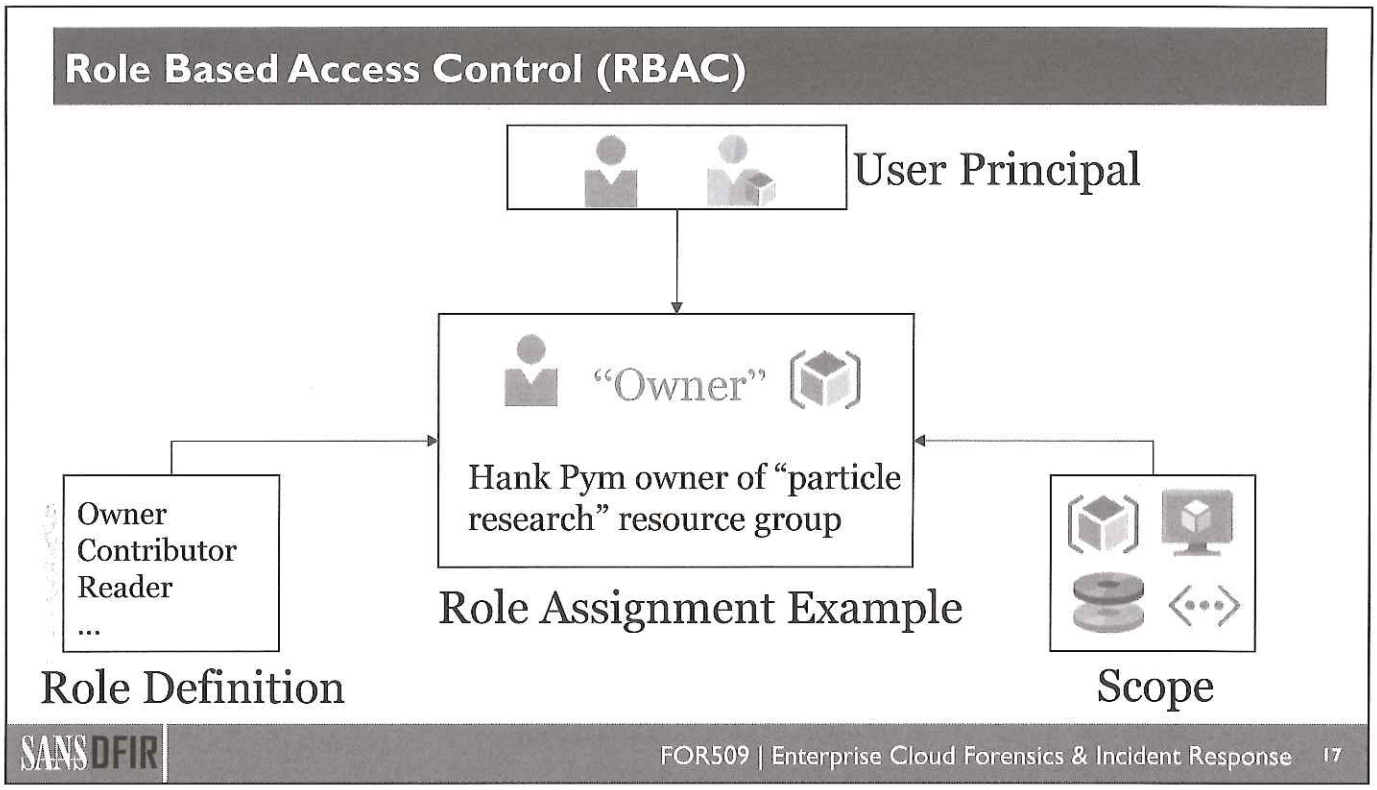
/subscription/<SubscriptionId>/resourceGroups/<resourcegroupname>/providers/<providername>/<resourceType>/<resourcename>

In the MiningVM example, we have a resource ID string for the VM itself, one for the OS disk, and one for the network interface. If we were to assign a public IP address to that VM, we would also have a resource ID string for that IP address.

This notation is important if you want to access Azure resourced via the CLI or PowerShell.



In this slide, you can see a graphical representation of the Azure Universal Resource Identified (URI) for our virtual machine called "Mining VM".



Azure Role Based Access Control (RBAC) lets you manage who has access to what resource and what they can do with that resource. Azure RBAC is an authorization system built on Azure Resource Manager.

To control access to resources, you create role assignments. There are three elements to a role assignment:

1. **Security Principal**
   - An object representing an entity such as a user or group, which can access the resource
2. **Role Definition**
   - A collection of permissions such as read, write, and delete
   - Azure has several built-in roles.[1] The most common are `Contributor`, `Owner`, and `Reader`
   - When performing an investigation, it's preferable to be granted `Owner` permission for maximum flexibility
3. **Scope**
   - Specify which role can access a resource or resource group
   - Scopes can be specified at four levels: management group, subscription, resource group, resource

There are many nuances to Azure RBAC, and this only covers a high-level overview that you are likely to encounter in your investigations. The Microsoft documentation should be referenced for an in-depth explanation of Azure RBAC.[2]

References:

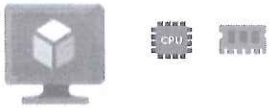
[1] https://for509.com/azureroles
[2] https://for509.com/azurerbac

Pricing

Clouds use a per-consumption model, which means everything you do has a cost

Temporary Costs

- Virtual Machine are priced based on the number of CPUs and amount of memory
- They only accrue cost when running
- You can make sure to shut them down when not in use



Virtual Machine

Persistent Costs

- Disks and snapshots are priced based on performance and quantity
- They accrue cost all the time until deleted

Persistent Disk

Disk Snapshot

We need to talk about pricing as it may impact your investigations. Like any other cloud provider, Azure charges on a per-consumption model, which means you have to pay for everything as you use it. For our purposes the charges we need to be concerned with are virtual machine cost, storage cost, and possibly data transfer costs. Data transfer costs can be avoided by conducting as much of the investigation in the cloud and specifically, in the same region as your victim machines.

Virtual machine cost has two components:

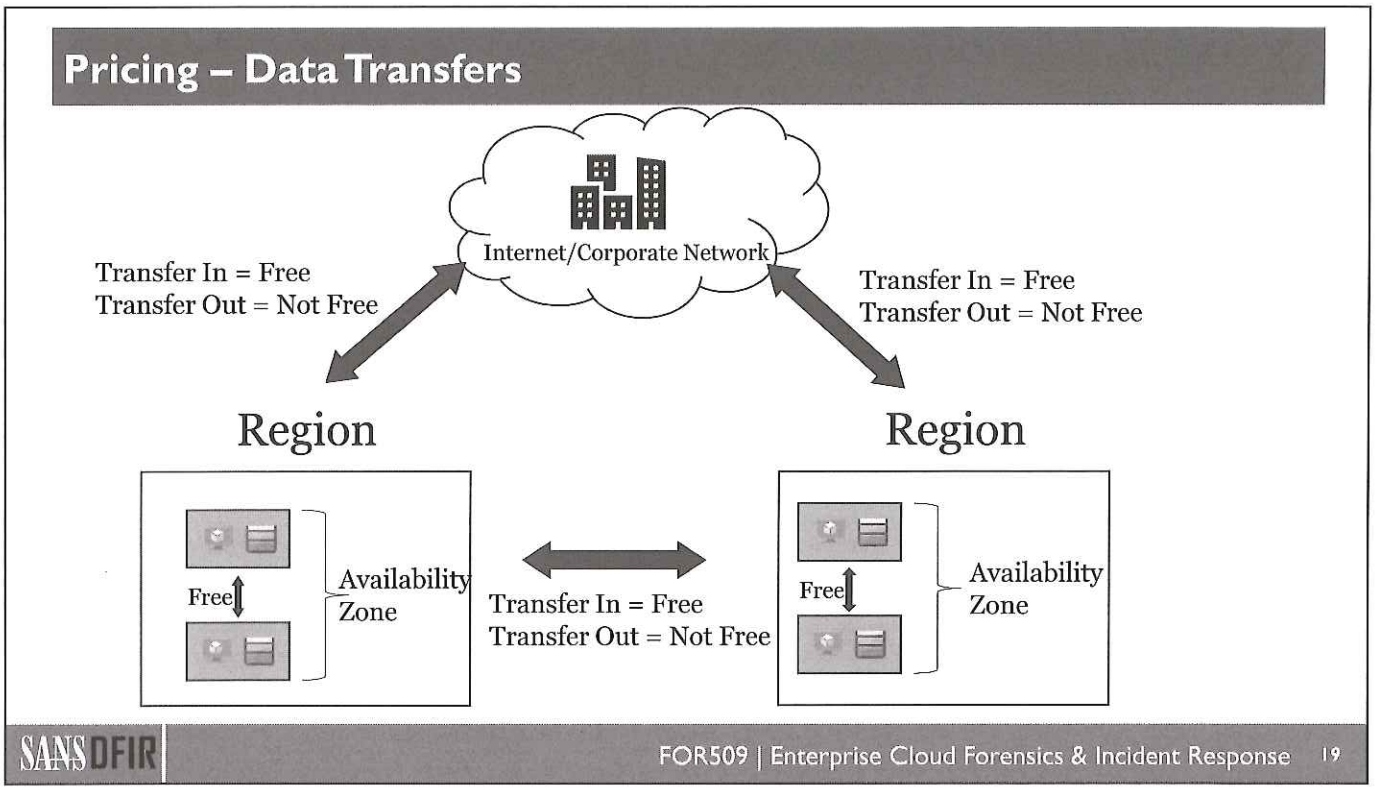
1. Memory and CPU provisioned for the VM
2. Length of time you keep the machine running

Storage is more complex. We will cover the different types of storage available in the next section. For the purpose of pricing, the main components are:

1. Performance
2. Quantity

You will use a lot of storage since the first step of any investigation is to snapshot the disks of the victim machines.

Storing logs also requires storage which is one of the main reasons you will find that logs are often not configured beyond the Azure defaults.



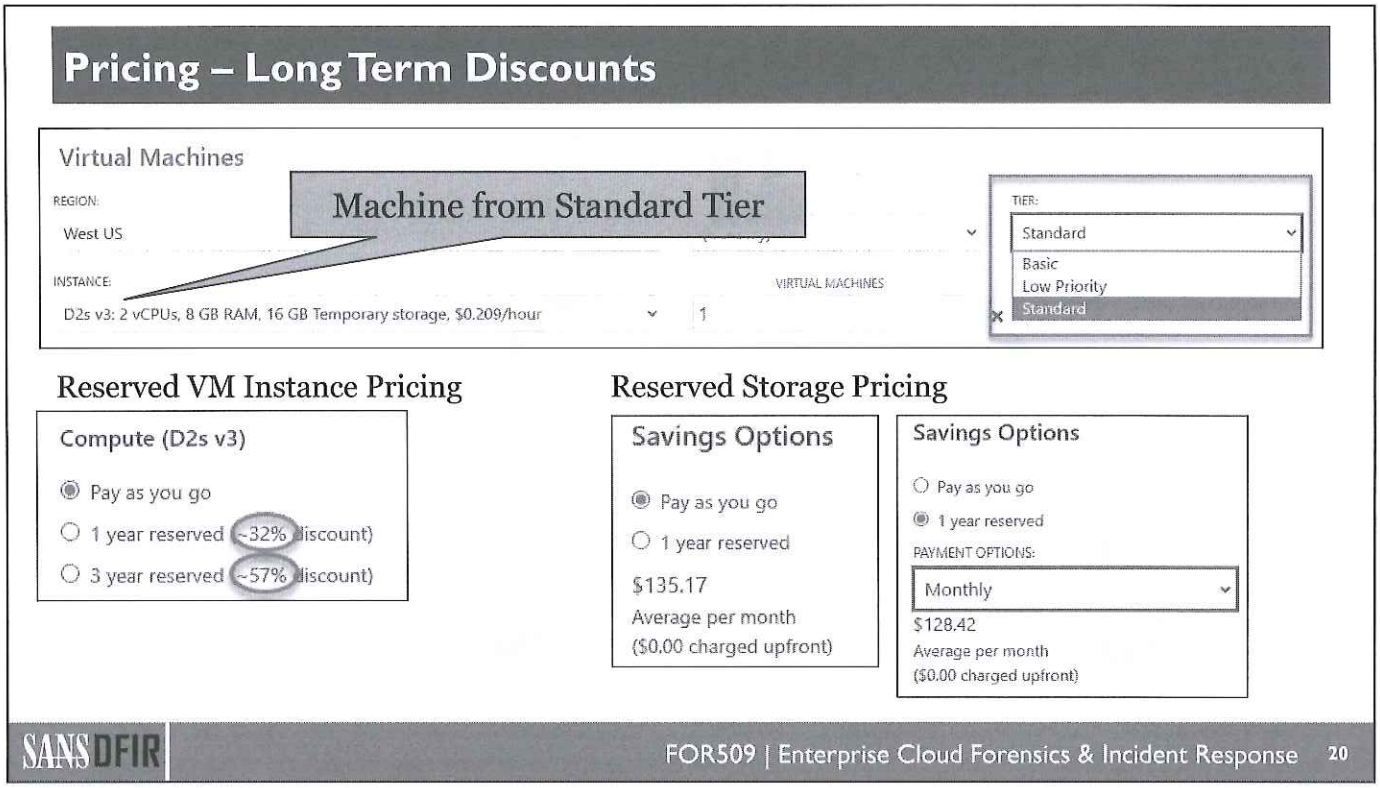
In addition to the costs for the virtual machines and the storage, you may incur cost if you move data from one region to another. The best way to avoid data transfer costs is to leave the data in the same region as your victim machine. If you must transfer the data, it's helpful to know when you will be charged.

Microsoft defines an availability zone as "an isolated location inside of an Azure Region, and has its own independent power source, network, and cooling". Transfers within an availability zone are free. However, data transferred between regions will incur costs. As a rule, incoming transfers are free but outgoing transfers aren't. Microsoft provides pricing in the "bandwidth pricing details" document.[1] If you need to transfer a large amount of data, Microsoft offers a number of solutions including AzCopy, Azure Data Box, and Azure Data Factory.[2]

References:

[1] https://for509.com/bandwidthpricing

[2] https://for509.com/largedatasets



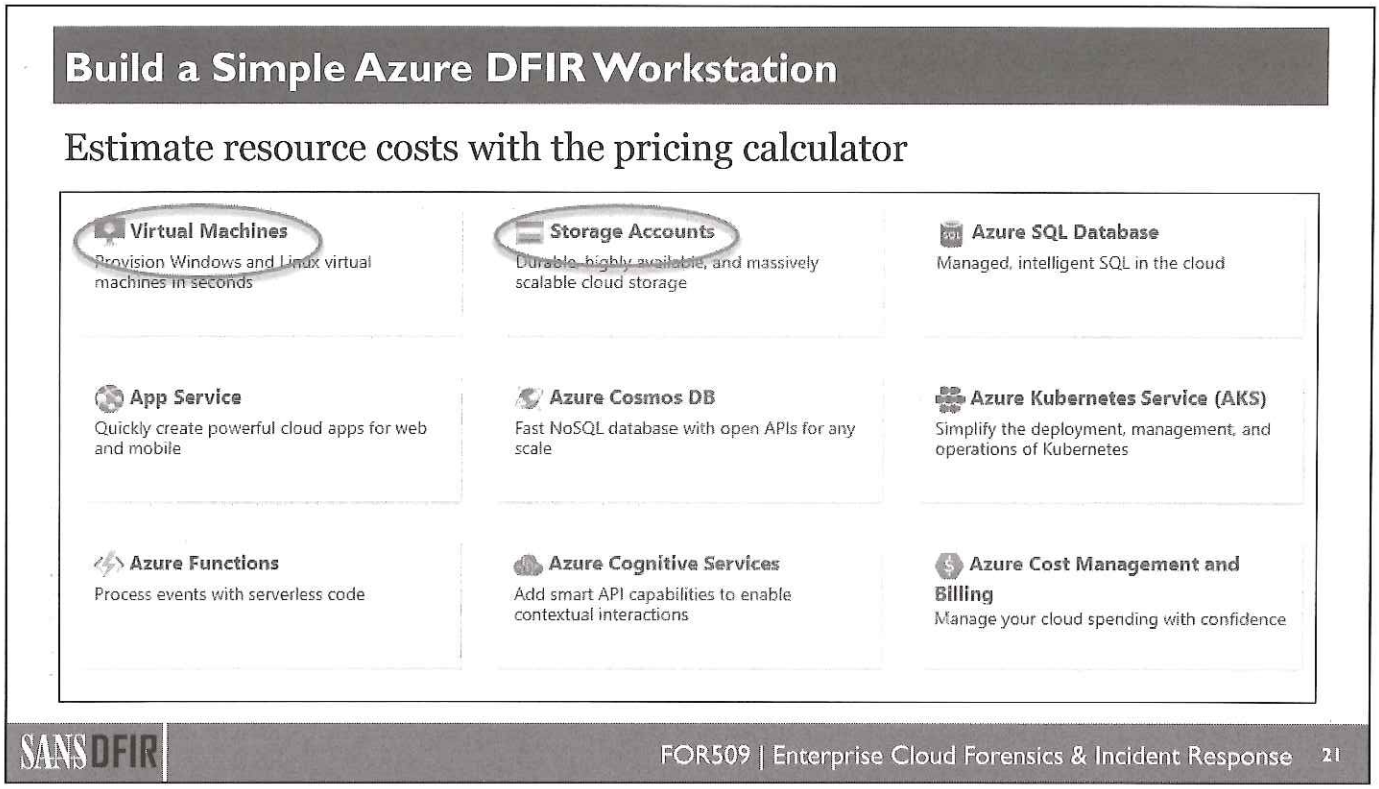
You can save money on certain products if you use Reserved Instances. Reserved Instances is Microsoft's terminology for committing to a 1-year or 3-year term. Not all products offer Reserved Instances, and the discount will vary depending on the region.

Reserved Instances could be helpful in certain cases where your investigation will take a long time to complete. They may be particularly helpful for storage which, as we previously discussed, is kept for a long time. Reserved virtual machine instances can have significant discounts.[1]

Virtual machines come in 3 different tiers: Basic, Low Priority, and Standard. You always want to choose a machine from the standard tier for your investigations.

References:

[1] https://for509.com/reservedvm



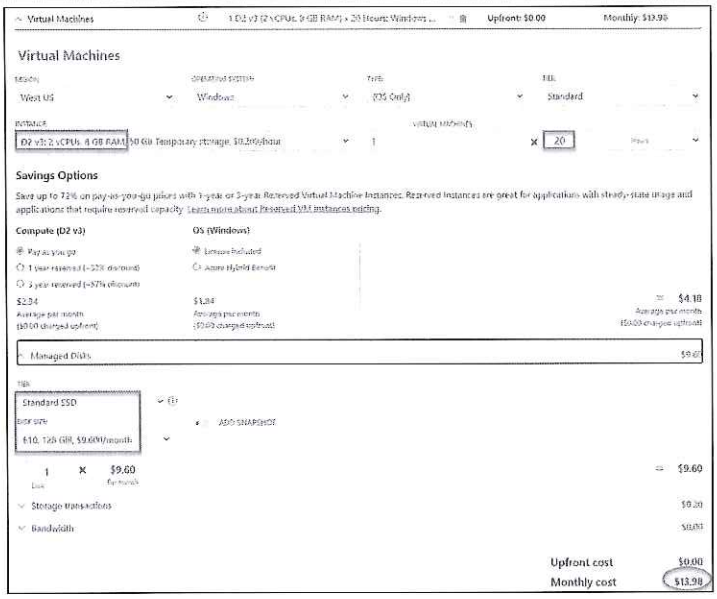
As an example, for a basic scenario, you will snapshot the disk of the victim machine, create a VM for your investigation and attach that snapshot to that VM (as a separate drive). The best way to estimate your cost for such a scenario is to use the Azure pricing calculator.[1]

By selecting "Virtual Machines" and "Storage Accounts" you will add these products to your estimate as shown in the next slides.

References:

[1] https://for509.com/pricingcalculator

DFIR Workstation – VM Price



Virtual Machine Estimate:

- 2 vCPUs
- 8GB RAM
- 128GB SSD for OS drive
- 20 hours uptime for the investigation

- Higher performance machine may be required depending on the investigation

Scoping out a virtual machine is very simple. Select the machine size based on the number of virtual CPUs (or cores) and amount of memory. The operating system drive can be small since we will mount the snapshot as a separate drive.

Azure offers everything from tiny machines with a single core and 0.75GB RAM, all the way to ginormous machines with 416 cores and 11400GB of RAM. Yes, that's correct over 11TB or RAM!!! Obviously, the hourly price will be proportional going from pennies per hour to well over a hundred dollars per hour.

Note that pricing will be different based on the region you select. All examples in this class are shown in US dollars.

Some of the Azure virtual machines offer cores and other vCPUs. This can be confusing.

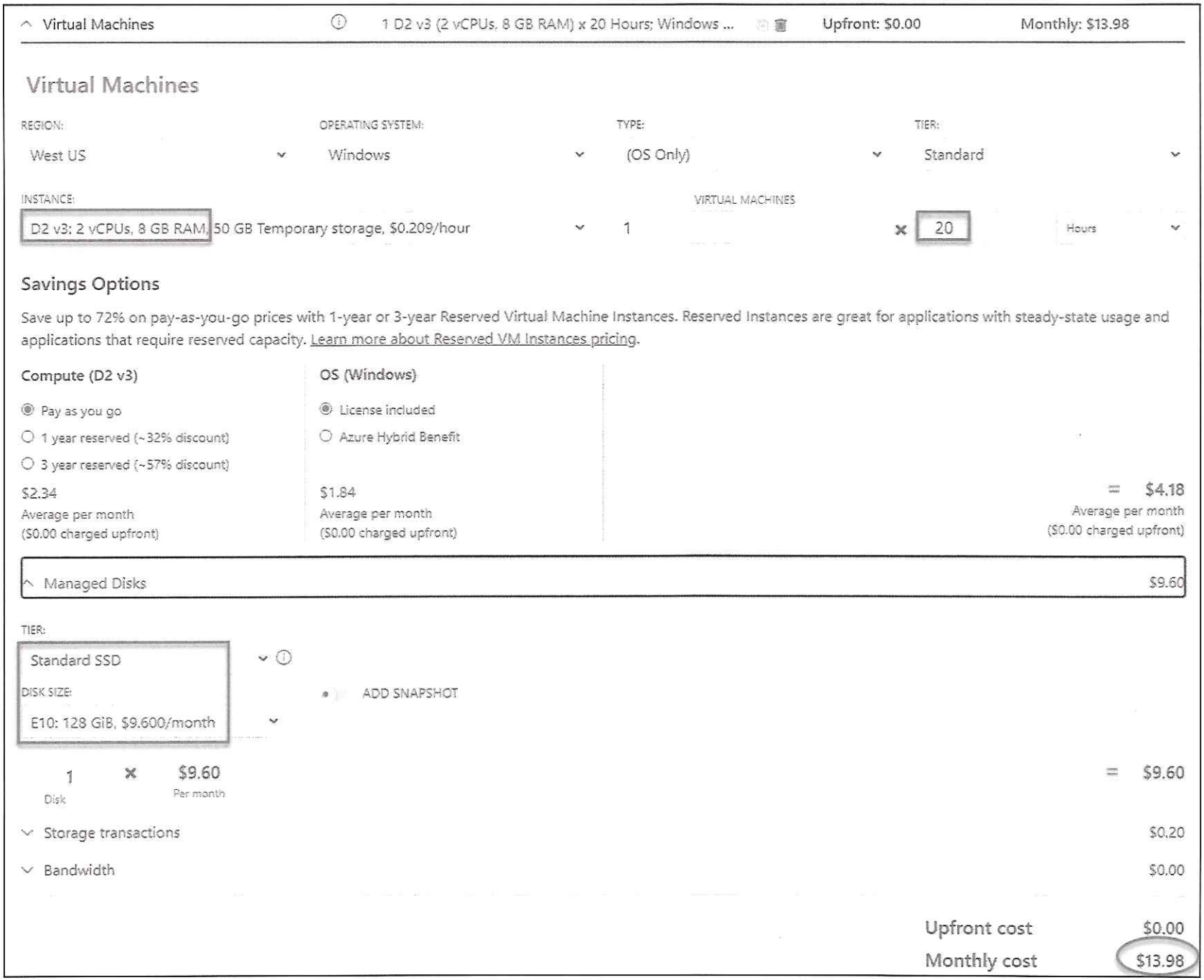
- A core is a physical unit of a CPU
- A virtual CPU (vCPU) is a physical CPU that's assigned to a virtual machine

The advantage of the vCPU is the use of hyperthreading.

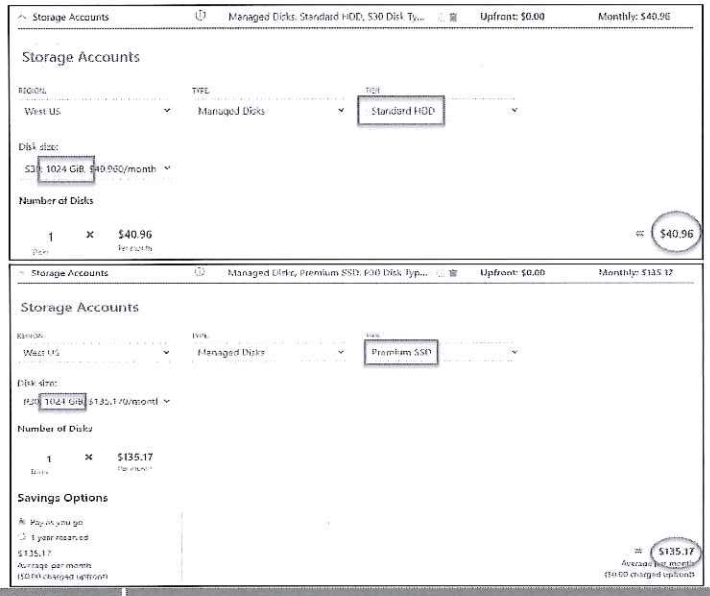
Usually, a machine with 2 vCPU and 8GB RAM is sufficient for our purposes. If you need higher performance, you can refer to the benchmarks published by Microsoft for Windows VMs.[1]

References:

[1] https://for509.com/vmbenchmark



DFIR Workstation – Storage Price

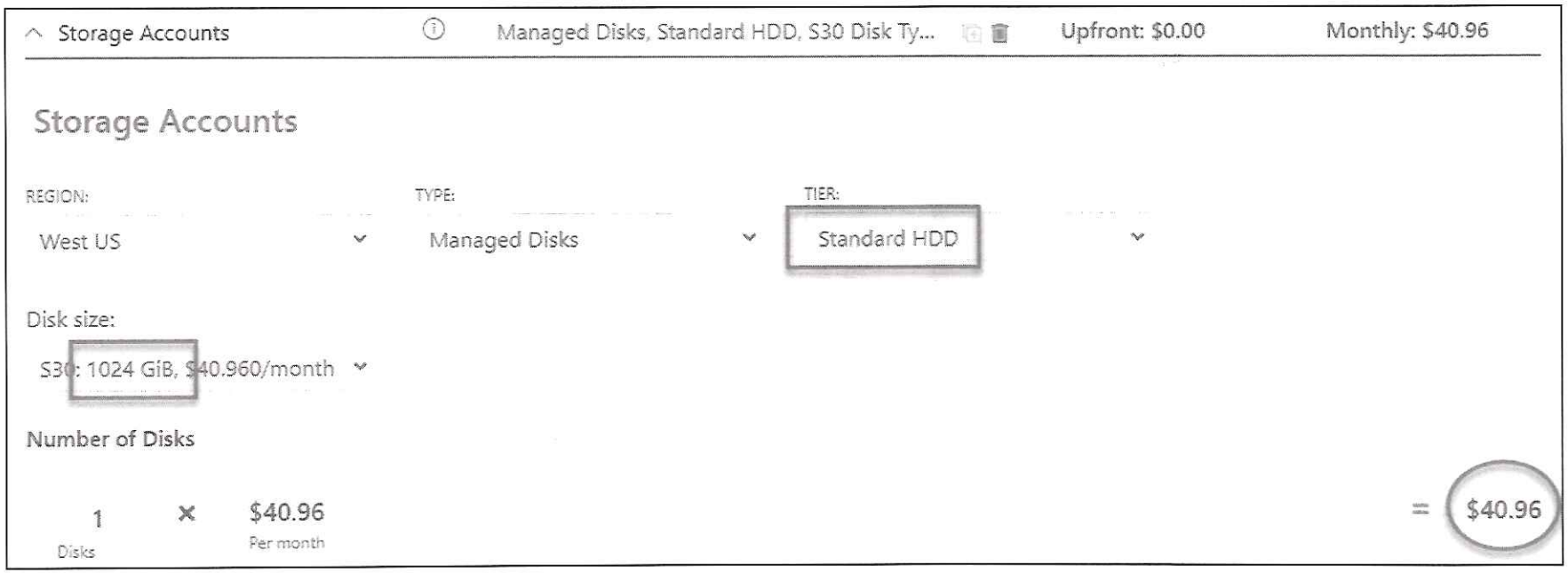


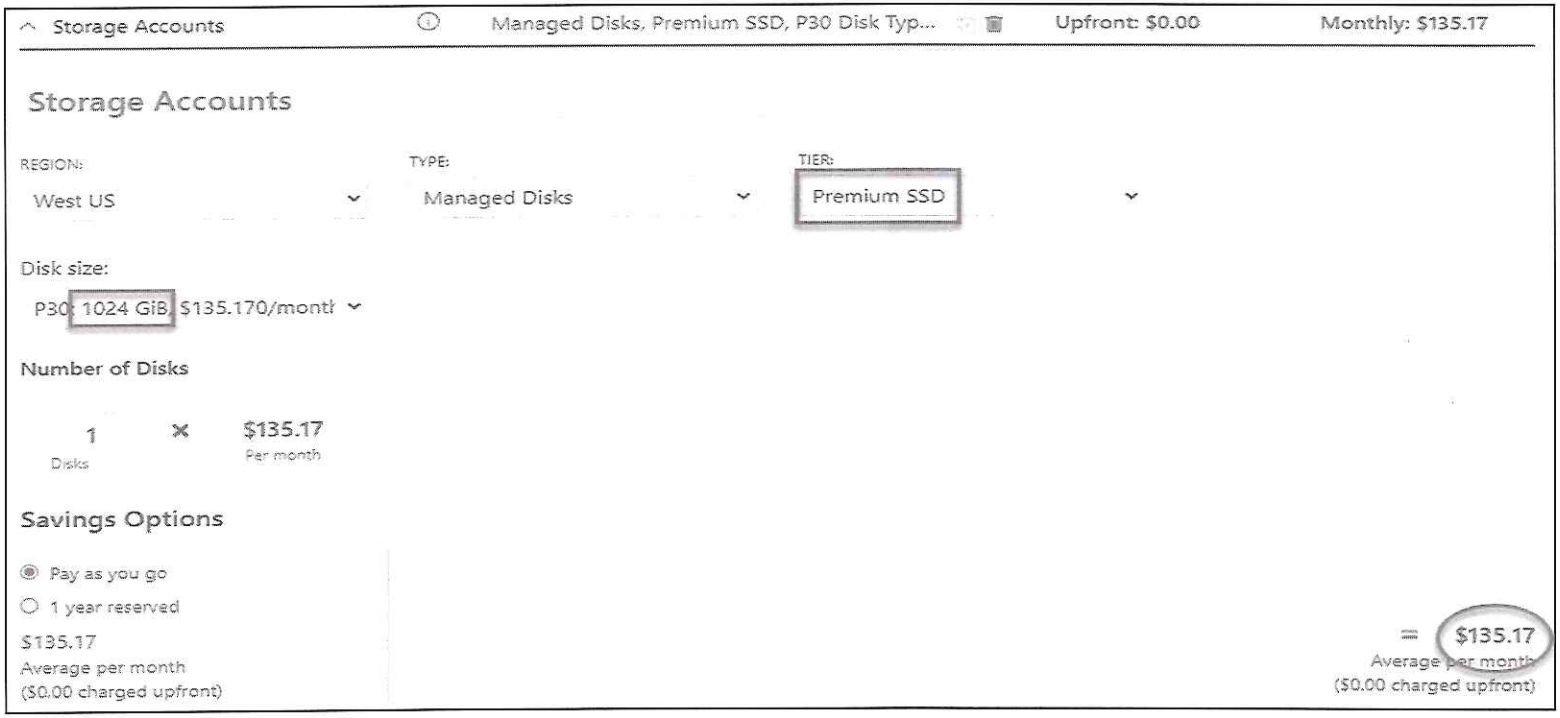
- 1st storage account is for the snapshot, so a Standard HDD is sufficient
- 2nd storage account is to apply the snapshot to and mount on the forensic VM. Performance is key, so Premium SSD is selected

The first step of our investigation will require that we make a snapshot of our victim's disk (1024GB disk as an example). We will therefore need to create 2 managed disks: one to hold the snapshot and one to apply the snapshot to.

In order to save money, we will use a standard HDD to create our snapshot. However, we will apply the snapshot to a Premium SSD as we will mount that drive on our forensic VM and run our analysis tools against that data.

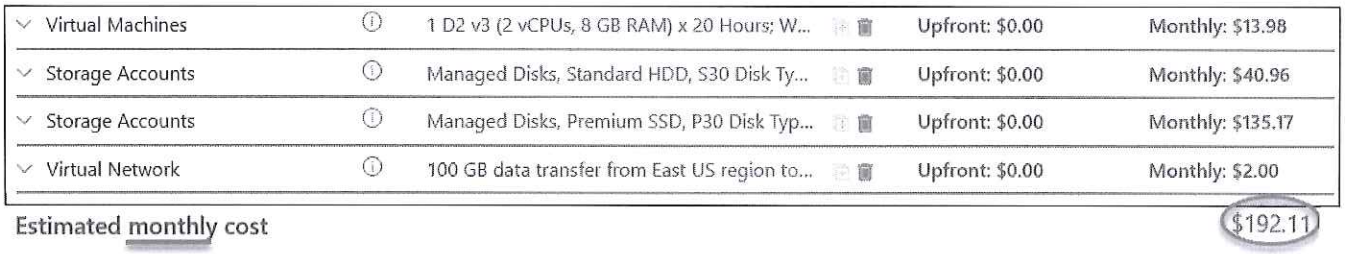
You can see there is a big price difference between standard storage and premium solid state storage. Also, remember that while the virtual machine can be shut down when not in use, the storage is persistent and will exists for the duration of your investigation.





DFIR Workstation - Summary

Example monthly pricing for a "small" investigation where all the work is conducted in the cloud.



Virtual Machines	ⓘ	1 D2 v3 (2 vCPUs, 8 GB RAM) x 20 Hours; W...	Upfront: $0.00	Monthly: $13.98
Storage Accounts	ⓘ	Managed Disks, Standard HDD, S30 Disk Ty...	Upfront: $0.00	Monthly: $40.96
Storage Accounts	ⓘ	Managed Disks, Premium SSD, P30 Disk Typ...	Upfront: $0.00	Monthly: $135.17
Virtual Network	ⓘ	100 GB data transfer from East US region to...	Upfront: $0.00	Monthly: $2.00

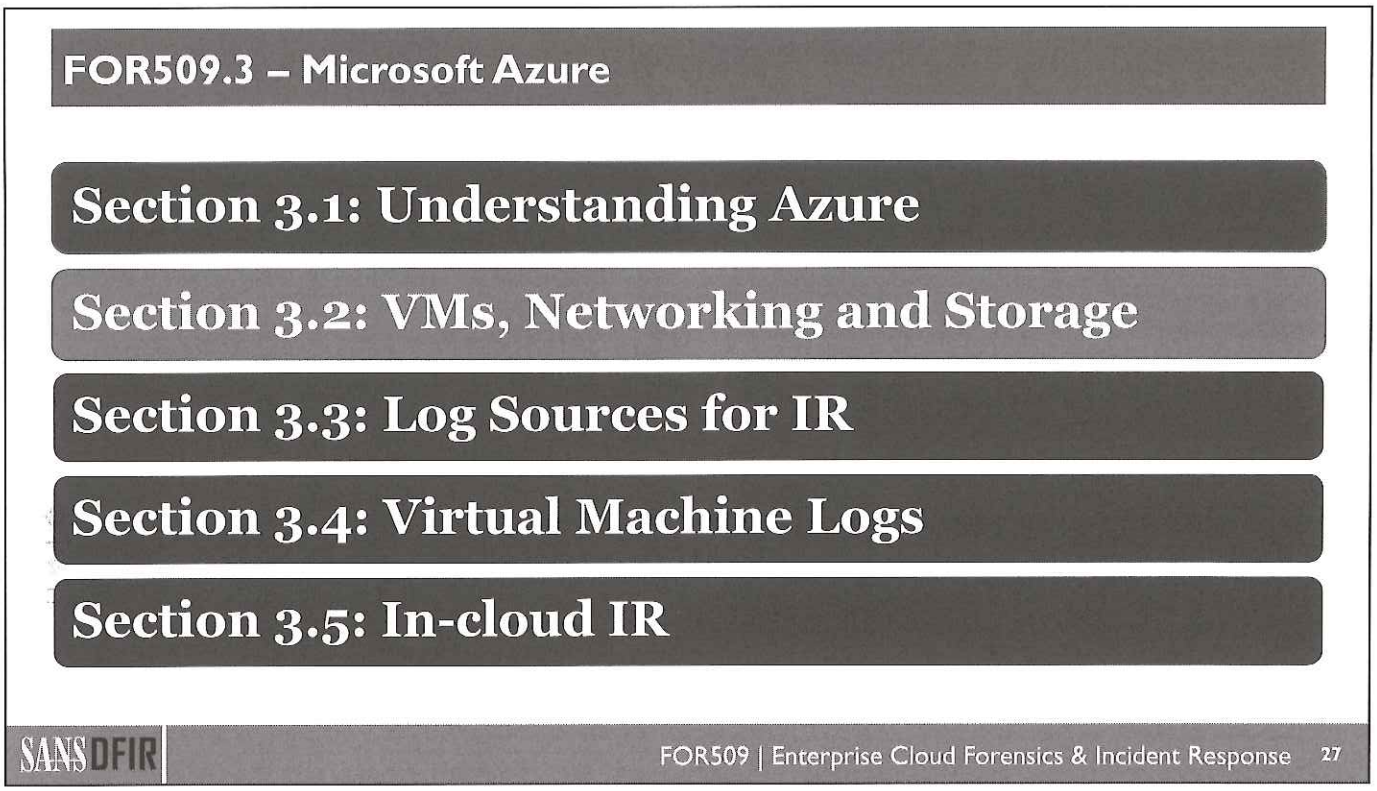
Estimated monthly cost $192.11

SANS DFIR FOR509 | Enterprise Cloud Forensics & Incident Response 26

Before starting an investigation in the cloud, it's important to have an idea of the costs that are likely to be incurred. This is a very simple example on how you may use the pricing calculator. Many investigations will involve a lot more virtual machines and storage which will significantly increase the cost. You may also use a log analytics workspace which will require its own storage. We will cover log analytics workspaces later in the class.

The pricing calculator's purpose is only to provide an estimate of charges. It shouldn't be used for billing purposes. Azure provides detailed billing information in the portal under the subscription service.

While we have mentioned it before, it bears repeating that virtual machines can be shut down when not in use, but storage is persistent and will continue to incur costs until deleted. For this reason, it's important to select the right kind of storage that balances price and performance. Conducting the investigation in a timely manner and cleaning up un-used resources will also keep costs down.



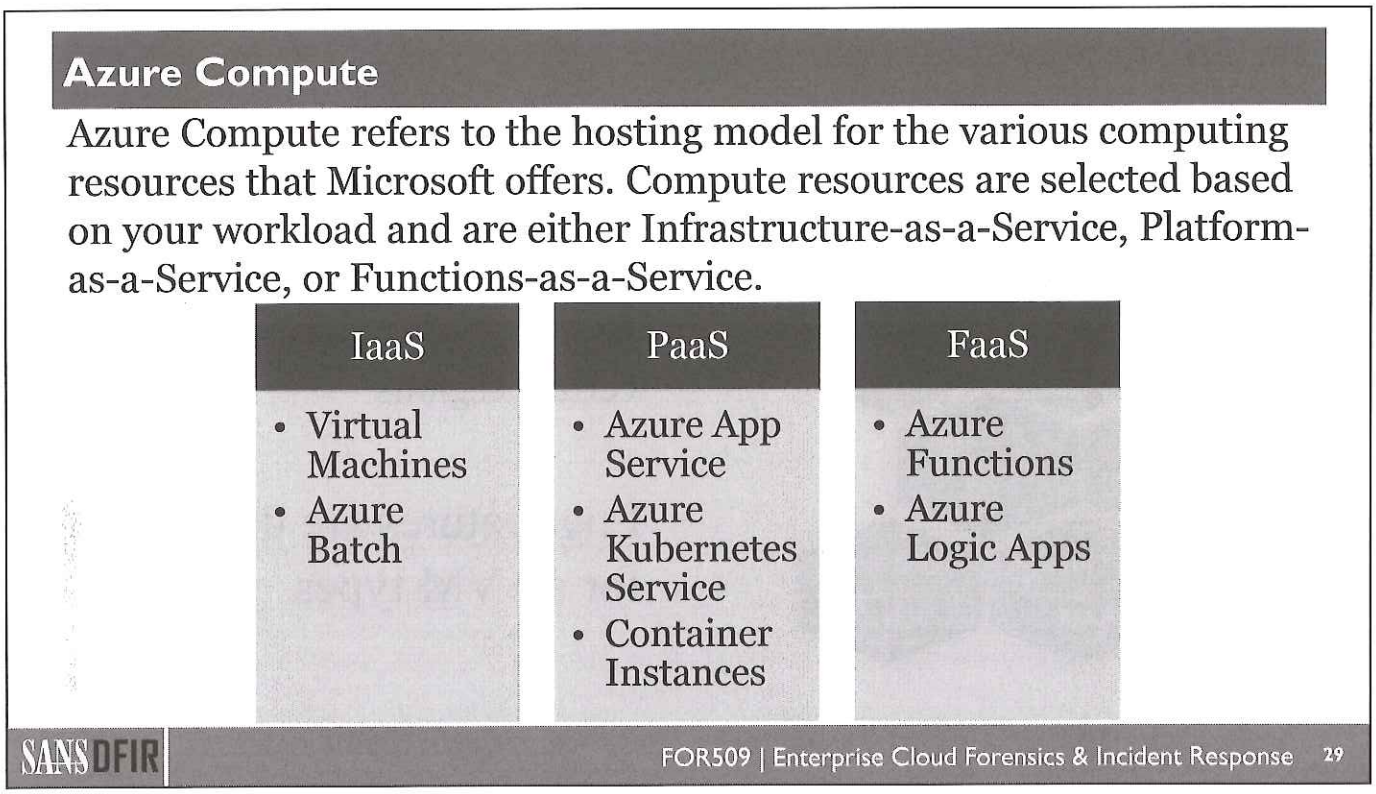
This page intentionally left blank.

Microsoft Azure Roadmap

- 3.1: Understanding Azure
- 3.2: VMs, Network and Storage
- 3.3: Log Sources for IR
- 3.4: Virtual Machine Logs
- 3.5: In-cloud IR

- Azure Compute
- Virtual Machine Types
- Case Study: Crypto Mining VM
- Azure Virtual Networks
- Network Security Groups
- Virtual Appliances
- Storage
- Accessing Azure
- **Lab 3.1**: Using SOF-ELK with Azure Logs

This page intentionally left blank.



The most visible resource of any cloud computing are the virtual machines (VM). However, there are many other kind of resources available in Azure. Microsoft calls this category Azure Compute and offers several options based on your workload and application.[1]

Azure Compute resources are categorized based on their service model.

Infrastructure-as-a-Service	Description
Virtual Machine	Typical server where you are responsible for everything
Azure Batch	Enables large-scale parallel and high-performance batch jobs with the ability to scale to tens, hundreds, or thousands of VMs

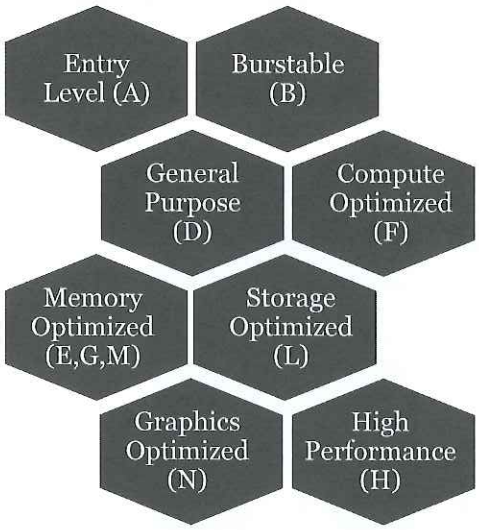
Platform-as-a-Service	Description
Azure App Service	Managed service to hosts web apps, mobile app back ends, RESTful APIs
Azure Kubernetes Service	Managed Kubernetes service for running containerized applications
Container Instances	Simple way to run a container in Azure without provisioning a VM

Functions-as-a-Service	Description
Azure Functions	Code executed in response to an event without concerns for the underlying platform or infrastructure
Azure Logic Apps	Logic apps are similar to functions for execute workflows instead of code

References:

[1] https://for509.com/azurecompute

Virtual Machine Types



- Many VM types to meet workload and application requirements
- Some are only available in certain regions

Log features are the same for all VM types

Since you will spend most of your time analyzing VMs, let's examine the different classes of VMs offered by Azure. Azure offers a wide variety of different types of virtual machines. Some are only available in certain regions and the Virtual Machines Pricing webpages will show which ones are available in which region: Windows[1] and Linux.[2]

Series A: Entry level

- Series A VMs are entry level machines suitable for development workloads, low-traffic website, micro services, etc.
- Naming examples: A1 v2, A2 v2, A4 v2, A8 v2

Series B: Burstable

- Series B VMs are a low-cost option that have the ability to burst to significantly higher CPU performance when the demand rises.
- Naming examples: B1S, B2S, B4MS, B12MB, B16MS, B20MS

Series D: General Purpose

- Series D VMs are optimized to meet the requirements of most production workloads. There are many variants in the D family, some of which emphasize certain features such as fast CPUs or fast disks.
- Naming examples: D2a v4, D2as v4, D2d v4, D2ds v4, D2s v4

Series F: Compute Optimized

- Series F VMs are optimized for compute intensive workloads. They have a high CPU-to-memory ratio.
- Naming examples: F1, F1s, F2s v2

Series E, G, and M: Memory Optimized

- Series E, G, and M VMs are ideal for memory intensive enterprise applications, such as database servers.
- Naming examples: E2a v4, E2as v4, E2ds v4, G1, G1s, M8ms, M208s

Series L: Storage Optimized

- Series L VMs feature high throughput, low latency, and directly mapped local NVMe storage.
- Naming examples: L8s v2, L4s

Series NC, NV, ND: Graphics Optimized

- Series NC, NV, and ND VMS have high end GPUs and target applications such as visualization, deep learning, and predictive analytics.
- Naming examples: NC6, NC6s v2, NC4as T4, NV6, NV12s, NV4as v4, ND6s, ND40rs v2, ND96asr

Series H: High Performance Computing

- Series H VMs are designed for high performance computing in applications such as financial risk modeling, seismic and reservoir simulation, and genomic research.
- Naming examples: H8, HB60rs, HB120rs v2, HC44rs

While you will encounter any combination of these VMs, logging and forensics analysis is performed the same way irrespective of the model.

References:

[1] https://for509.com/windowspricing

[2] https://for509.com/linuxpricing

Case Study: Detecting a Crypto Mining VM

- Cloud crypto mining is a favorite action-on-objective for bad actors
- Only **N**-series VMs have GPUs
- Monitor logs for creation of **N**-series VMs
- Create workflow to notify subscription owner
- PowerShell script provided in the notes

```
PS> .\azlogs.ps1 (script provided in the notes & class GitHub)

VmID                 VmID
----                 ----
Standard_NV4as_v4    f57b8807-a5bb-4e87-8f7d-d7ad0a3f0f80

Standard_D2s_v3      08bc[illegible]-49a4-b1ea-e927c338f82a
```

VM to validate with subscription owner

The next section will have an in-depth discussion of the various Azure log sources. However, while we are on the topic of virtual machines, let's quickly discuss an idea to potentially detect the creation of crypto mining VMs. A bad actor whose action-on-objective might be to run a crypto miner will first need to access your Azure subscription. Perhaps they get extremely lucky and are able to compromise a VM which features a GPU. But most likely that won't be the case and they will need to create an appropriate VM (or many such VMs).

As we saw in the last slide, GPU enabled VMs belong to the N-series of VMs (NC, ND, and NV). The idea is to monitor the VM creation logs for this series of VMs. In the log, they will show up in a field called "vmSize" as "Standard_N*" where N* stands for the specific model name. An example would be "Standard_NV4as_v4".

While the log may reflect a legitimate creation of such a VM, a workflow could be designed to notify the subscription owner of this new specialized VM. It would then be up to the subscription owner to decide if this is a legitimate VM or not.

If your organization doesn't normally use GPU-enabled VMs, this is a very simple early warning system. On the other hand, for an organization that dynamically builds and tears down a large number of these kind of VMs, additional filters will need to be implemented.

This workflow can be implemented in two different manners depending on the organization. If your organization imports Azure logs in a SIEM, writing rules for that condition would be the simplest method.[1]

Here is an example rule for Splunk:

```
Index=ms_azure properties.hardwareProfile.vmSize=Standard_N* | dedup
properties.vmId | stats count by properties.hardwareProfile.vmSize
```

Example 30-day snapshot from a large corporation:

Properties.harwareProfile.vmSize	count
Standard_NV12s_v3	3867
Standard_NV6	474
Standard_NV12_Promo	22
Standard_NV12	7
Standard_NV6_Promo	7
Standard_NC24s_v3	6
Standard_NV32as_v4	4

Over 4,000 GPU-enabled VM creation in a 30-day period may seem suspicious. However, this is an example of a company that processes large amounts of data over short periods of time and therefore dynamically creates and tears down these VMs. The next step would be to add a filter and narrow down either by subscription or resource group to remove those that are authorized to create these GPU-enable VMs.

For organizations that don't have a SIEM, here is some PowerShell code that will extract the information from the log.[2]

```
$results = get-azlog -ResourceProvider "Microsoft.Compute" -DetailedOutput
$results.Properties | foreach {$_} | foreach {
   $contents = $_.content
   if ($contents -and $contents.ContainsKey("responseBody")) {
      $fromjson=($contents.responseBody | ConvertFrom-Json)
      $newobj = New-Object psobject
      $newobj | Add-Member NoteProperty VmId $fromjson.properties.vmId
      $newobj | Add-Member NoteProperty Vmsize
$fromjson.properties.hardwareprofile.vmsize
      $newobj
      }
}
```

There are many other parameters that you can extract from the logs. This is a simple example to illustrate the possibilities. One downside of using PowerShell rather than a SIEM is that the script must be executed for each subscription.

References:

[1] https://for509.com/splunkautomation

[2] Shoutout to Arjun Bhardwaj and Michael Getachew for their assistance with PowerShell scripting.

Azure Virtual Network (VNet)

- Required for communications between VMs and/or the internet.
  - Address space
    - Range of IP address that are available for the resources in that VNet
  - Subnet
    - Smaller network to facilitate resource grouping and security
  - Regions
    - Belongs to a single region and single subscription
    - Every resource on the VNet must be in the same region

Subscription	Azure subscription 1
Resource group	Research
Name	MetalVNet
Region	East US
IP addresses	
Address space	10.1.0.0/16
Subnet	Gold (10.1.10.0/24),Silver (10.1.20.0/24),Copper (10.1.30.0/24)

Azure Virtual Network (VNet) is the glue that allows other Azure resources to communicate with each other and with the internet.[1]

When creating a VNet, Azure will assign a set of RFC 1918 IP addresses, typically 10.0.0.0/24 for the first VNet you create. This private address space allows your VMs to communicate with each other. You can assign any public or private address range you want to your VNet. Creating a VNet doesn't incur any charges.

To communicate with the internet, you will need to assign your VM a public IP address. There is a recurring charge to use Azure's public IP addresses.

There are 3 ways to communicate with on-premises resources:

1. Point-to-site virtual private network (VPN): connection between a VNet and a single computer.
2. Site-to-site VPN: connection between your on-premise VPN device and an Azure VPN gateway deployed in the VNet.
3. Azure ExpressRoute: connection between your premises and Azure through an ExpressRoute partner. This connection is private and doesn't go over the internet.

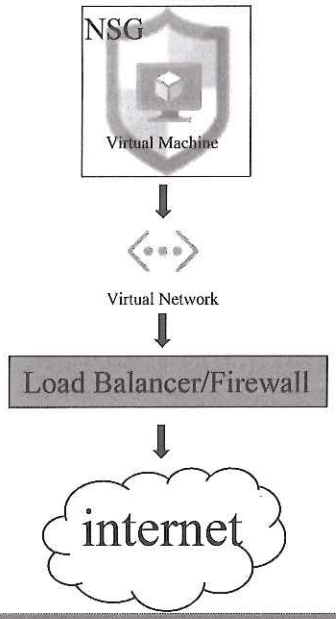
From an incident response point of view, it's very important to understand network topology and communication channels between Azure resources. We will discuss network security groups in the next slide.

References:

[1] https://for509.com/virtualnetwork

Network Security Group

- Stateful packet filtering based on:
  1. Source IP
  2. Source Port
  3. Destination IP
  4. Destination Port
  5. Protocol
- For each rule, you must specify:
  1. Priority
  2. Action: Allow or Deny
- SSH and RDP are opened with source: Any & destination: Any (must change ASAP)



To protect your VM, Azure will automatically create a network security group (NSG). The NSG allows you to control traffic in and out of the subnet. The NSG conducts a stateful inspection of the traffic based on the Source IP, Source Port, Destination IP, Destination Port, and Protocol.[1] This is to be considered a very basic firewall. Additional offerings are available from Azure and third parties for full-fledged firewalls that include application layer inspection.

NSG rules are read in order based on the priority number assigned to each rule from 100 to 4096 (excluding Azure defined rules in the 65000 range). 100 is the highest priority rule and 4096, the lowest.

From an incident response point of view, it's important to understand the network topology and associated control. For example, you can imagine the scenario where you are told that a firewall rule was in place to block nefarious traffic, but your investigation reveals that the priority of that rule was too low to be effective and that a higher priority rule was in fact allowing that traffic to the VM.

References:

[1] https://for509.com/nsg

Network Virtual Appliance

Many devices can shape network traffic and can produce their own set of logs. Review the network topology before you start your investigation and look for the presence of network-shaping devices. These network virtual appliances are offered by both Microsoft Azure and third parties:

- Load Balancer
- Firewall
- Application Gateway
- VPN gateway
- WAN optimization appliance
- Virtual router

There are many other pieces of network infrastructure that can shape the traffic. These may have various impacts to your incident response or forensic investigation. You need to be aware of their presence so that you may investigate their logging capabilities as they may provide further insight to your investigation.

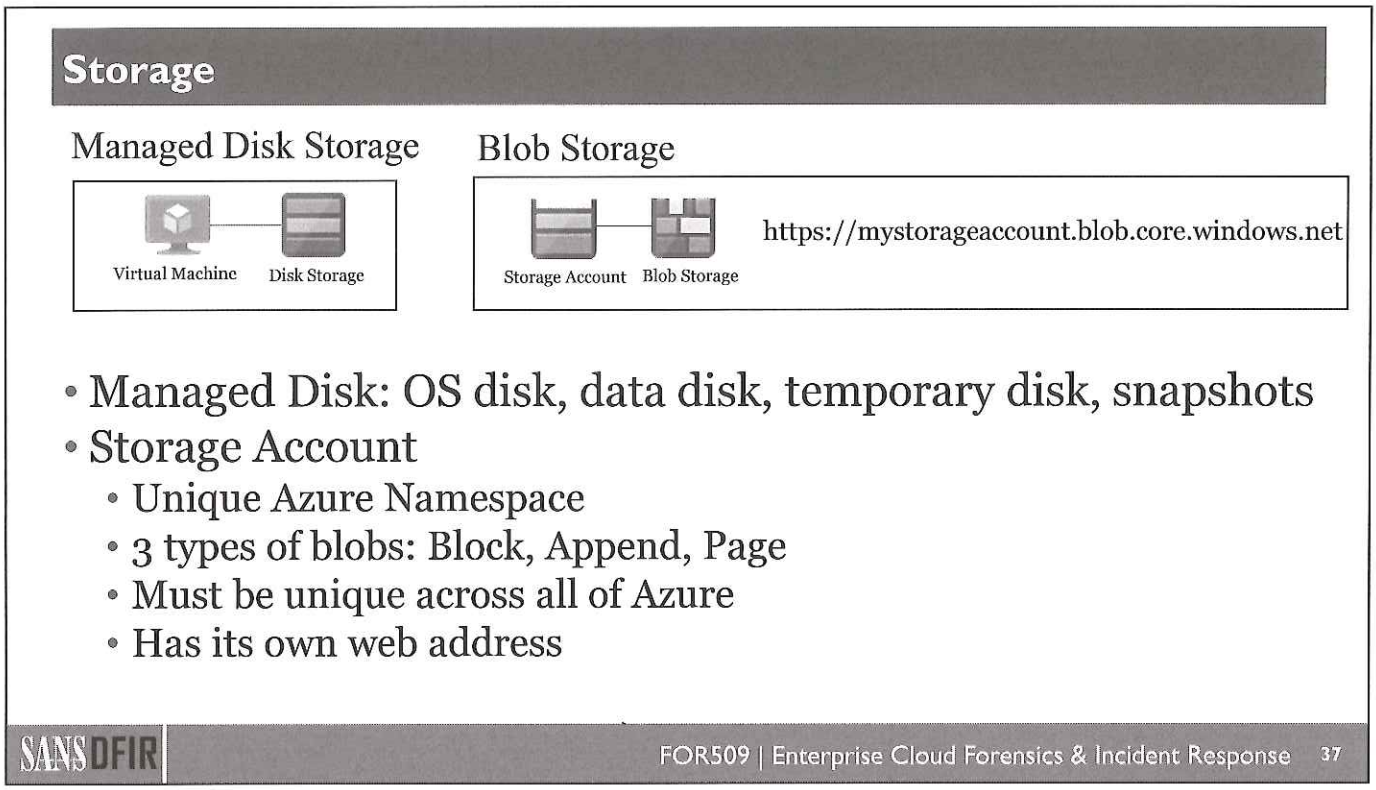
Azure offers the following options:

- Azure load balancer which will evenly distribute incoming network traffic across a groups of resources.
- Azure firewall which offers both stateful network and application-level filtering.
- Application Gateway which will protect your application from common web vulnerabilities.
- VPN gateway which connects your on-premise network to Azure.

In addition, third party vendors offer many network virtual appliances such as firewalls, WAN optimization, VPN access server, virtual router, etc. These offerings can be found on the Azure marketplace.[1]

References:

[1] https://for509.com/marketplace



Storage will be a key component of any investigation you conduct. In addition, many logs will only be retained if you create dedicated storage for them. There are 6 types of storage on Azure:[1]

1. Disk: persistent disk storage for every workload.
2. Blob: massively-scalable object storage for unstructured data.
3. File: simple, distributed, cross-platform file system.
4. Data Lake Storage: Limitless storage for analytics data.
5. Archive: storage for rarely accessed data.
6. HPC Cache: file caching for high-performance computing.

We will focus on blob storage[2] and disk storage.[3]

Storage accounts provide a unique namespace for your data. As we discussed in the previous section, every resource in Azure has a unique resource id. When you create a storage account, you are able to access your data directly using the URL:

```
https://mystorageaccount.blob.core.windows.net
```

Where `mystorageaccount` is the name you selected for your storage account.

This will be a very useful feature when you need to download logs for analysis or safekeeping.

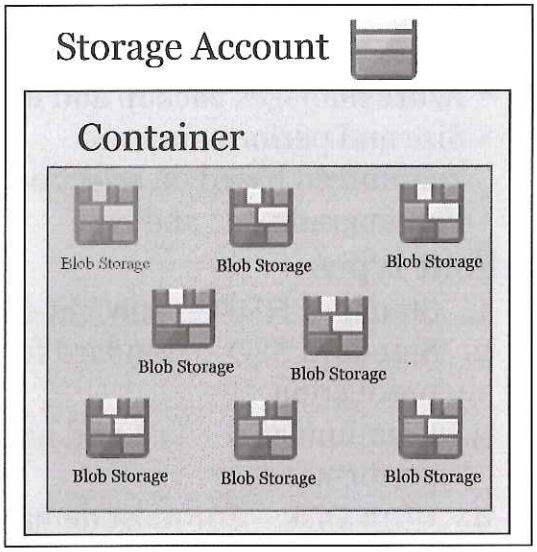
We will discuss the characteristics of blob storage and disk storage in the next slides.

References:

[1] https://for509.com/storageintro
[2] https://for509.com/blobintro
[3] https://for509.com/diskintro

Storage - Blob

- Three levels:
  1. Storage account
  2. Container in the storage account
  3. Blob in a container
- Blob = Binary Large Object
  - Block Blob
    - Any file up to 4.75TB
  - Append Blob
    - Works well for logging where data is constantly appended
  - Page Blob
    - Used for virtual hard drives up to 8TB



Blob stands for Binary Large Object. It's a way for Azure to store an arbitrarily large amount of unstructured data. Any type of data can be stored in blobs which is very convenient for videos, images, and particularly text. As you can imagine, for our purposes it's very convenient to store logs; large amounts of logs.

There are three types of blobs:

1. **Block blobs** to store text and binary data. Since block blobs are made up of blocks of data, they can be managed individually and can store up to 4.75TB of data. Microsoft is currently previewing block blobs up to 190.7TB.
2. **Append blobs** are optimized for append operations and are ideal for logging data.
3. **Page blobs** store random access files up to 8TB. They are usually used to store virtual hard drive files (VHD).

There are two characteristics that are very convenient with blobs:

1. With the correct permission, they can be accessed directly over the internet via HTTP or HTTPS using the url: `http://mystorageaccount.blob.core.windows.net`
2. There are half a dozen methods to transfer data in or out of blobs. The two easiest ones are: AzCopy[1] and Azure Storage Explorer[2] (which leverages AzCopy).

References:

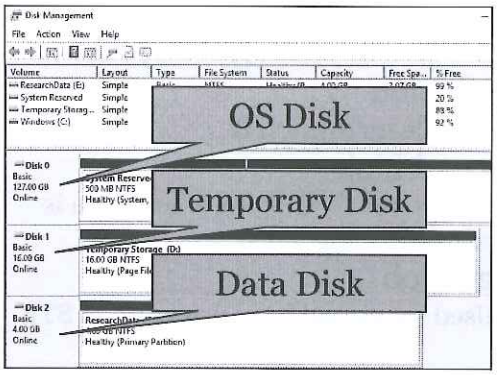
[1] https://for509.com/azcopy

[2] https://for509.com/storageexplorer

Storage – Managed Disk

- Managed Disk
  - 99.999% availability
  - Azure manages backup and uptime
  - Size and performance are guaranteed based on selection
  - Can upgrade size and type
- Four types
  1. Standard HDD – slow but cheap
  2. Standard SSD – standard for production
  3. Premium SSD – fast and high performance.
  4. Ultra Disk – For most demanding and data intensive workloads

- Monthly cost based on:
  - Disk type and size
  - Snapshots
  - Outbound data transfers
  - Transactions



A managed disk is very similar to the physical disk you have in your laptop or desktop. When configuring a managed disk, you select the size and the type[1]: Ultra disk, Premium SSD, Standard SSD, Standard HDD.

However, there is a big difference that is very relevant to our investigation: **cost**.[2] Unlike the disk in your laptop or desktop which has a one-time cost, Azure managed disks have a recurring cost. There are 5 components to the cost of a managed disk:

1. Disk type: SSD (various levels as previously indicated) or HDD.
2. Disk size: as expected the larger the disk, the more it costs.
3. Snapshots: billed based on the size of the snapshot – these are very important for our investigations.
4. Outbound data transfers: transferring data out of Azure will incur billing for bandwidth usage.
5. Transactions: billed for each I/O operation.

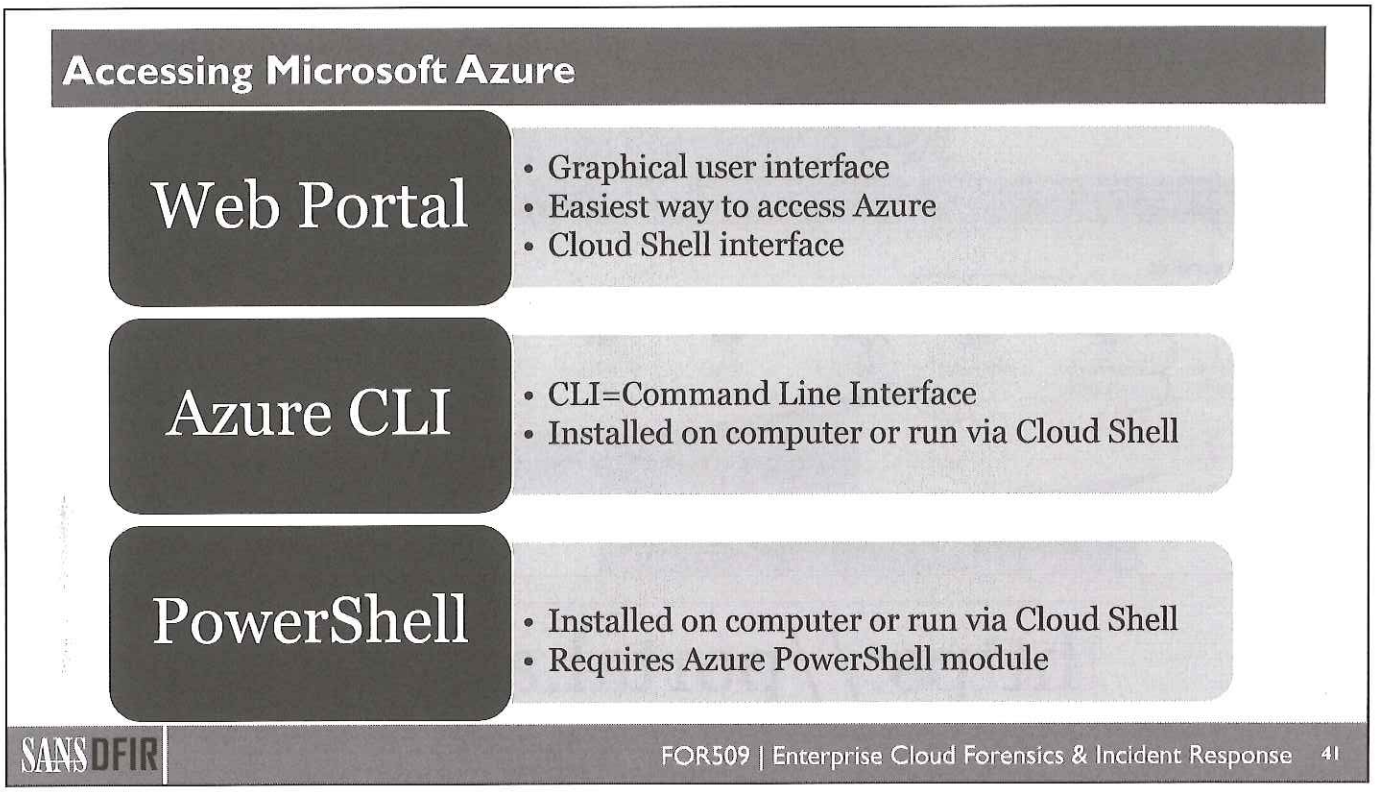
Most VMs will have 2 or more managed disks:

- OS disk which is selected when the VM is created.
- Temporary disk used for short term storage (example: page or swap files). Not all VM types have temporary disk.
- One or more data disk which is user created.

As a preview of the last section of the class, one method we use to perform our investigations is to snapshot the OS disk of the compromised machine, apply that snapshot to a new disk and mount that new disk as a data disk on a fresh VM. This is why understanding managed disks is so important.

References:

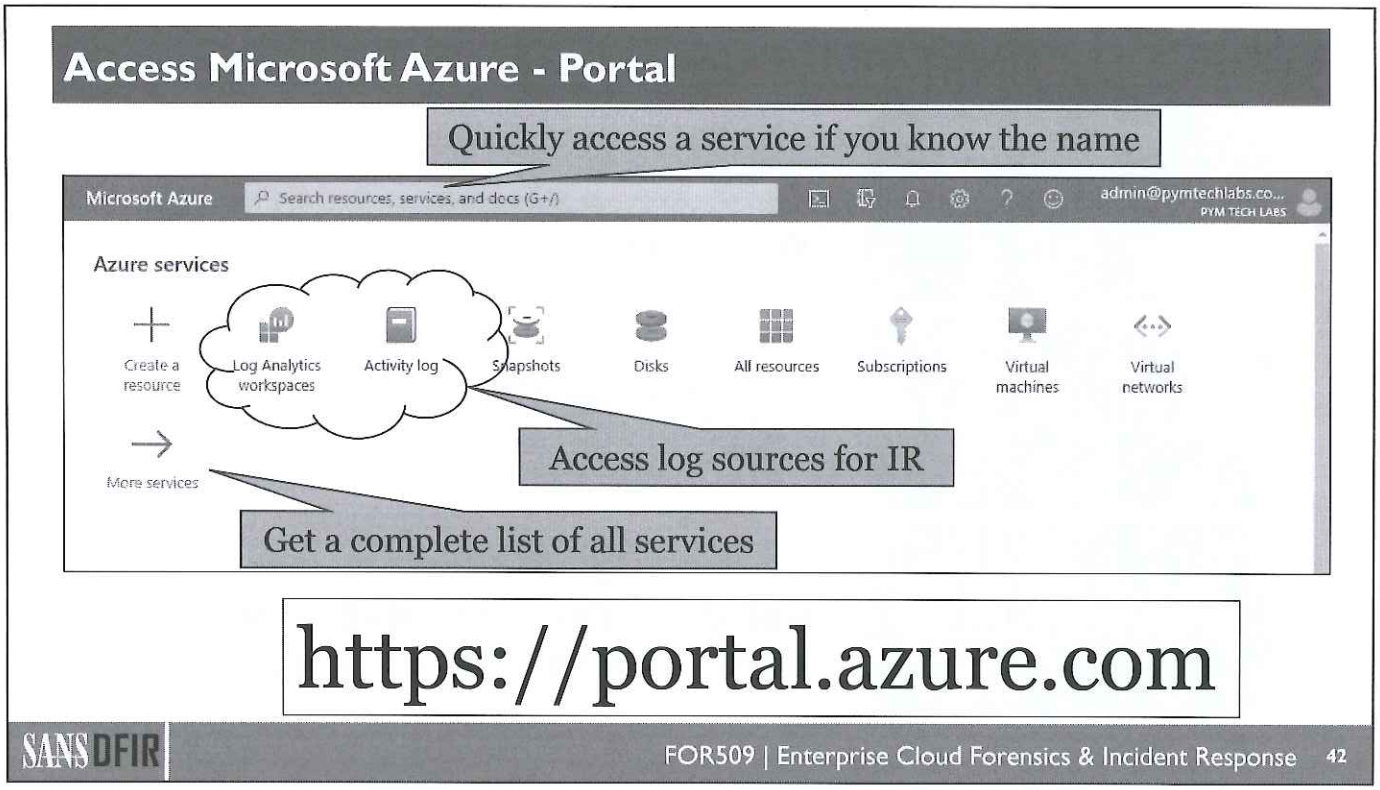
[1] https://for509.com/disktypes
[2] https://for509.com/diskprice



Before we move on to the next section and discuss the various log sources in Azure, we need to quickly review the three ways to access Azure:

1. Azure portal
2. Azure CLI
3. PowerShell

Azure CLI and PowerShell can be installed on your computer or used via Cloud Shell. Cloud Shell is a command line interface available in the Azure portal.



The Azure portal is a graphical user interface and is the most common and easiest way to access Microsoft Azure.[1]

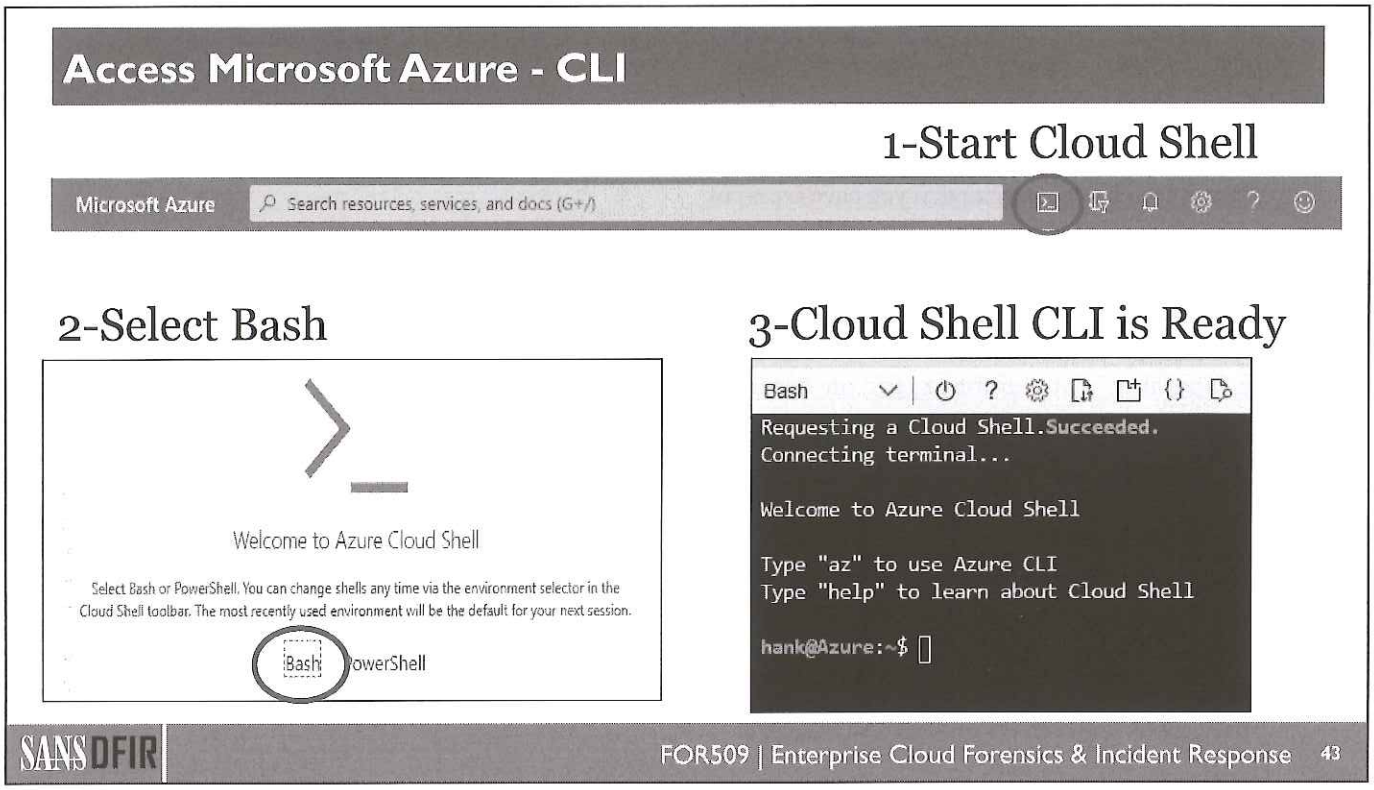
For the purposes of this class, the most important services will be `Activity log` and `Log Analytics workspaces`. Other important services are `Resource groups`, `Virtual machines`, `Virtual networks`, `Disks`, `Snapshots`, and `Subscriptions`.

The service `All resources` is also very important as it gives you a quick overall view of every resource provisioned in your subscription.

The top row will only show you the icons for the most recent services used. If you don't see what you are looking for, you can type it in the search box or select "More services" to get a complete list.

References:

[1] `https://for509.com/portaloverview`



The Azure command-line interface (CLI) is a set of commands used to create and manage Azure resources[1]. It can be installed on your computer[2] or run via Cloud Shell[3]. This slide shows Azure CLI running from cloud shell.

Microsoft provides the Azure CLI for Windows, macOS, and Linux. If you would rather not install software on your computer, you can use the same CLI commands in the Azure Cloud Shell Bash environment. The Azure Cloud Shell is an interactive shell for managing Azure resources. It provides a terminal window inside your browser. The terminal window is based on the Linux bash shell so in addition to the Azure CLI commands, you also have access to a wealth of Linux commands.

When selecting the Cloud Shell in the Azure Portal, you will select the Bash environment if you wish to use CLI commands.[4] Another option is PowerShell which we will discuss in the next slide.

Whether you choose to run the Azure CLI on your own computer or via the Cloud Shell, you now have access to the set of "az commands".

The first command you will need to issue is to authenticate yourself (not needed if you are using Cloud Shell, since you already authenticated to the Azure Portal)

```
az logon
```

You can then list the subscription you have access to

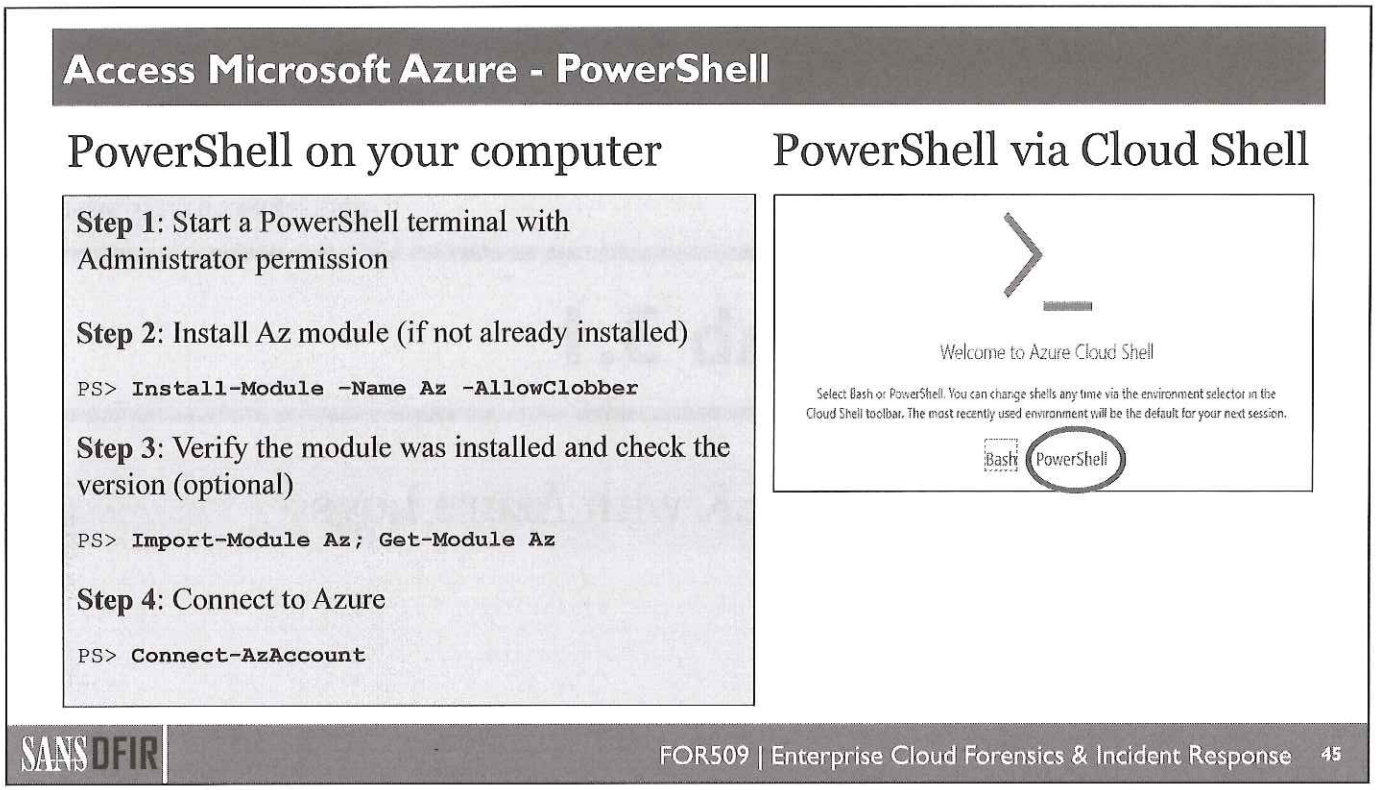
```
az account list
```

You now need to select the subscription that's appropriate for your investigation

```
az account set –subscription "name of subscription"
```

References:

[1] https://for509.com/cli-intro
[2] https://for509.com/cli-install
[3] https://for509.com/cloudshell-intro
[4] https://for509.com/bash



Another way to interface with Azure is through PowerShell. Just like Azure CLI, you have the option to run PowerShell from your computer or run via Cloud Shell.[1]

To use PowerShell on your computer, you will need to install the Azure PowerShell module. Be sure to install the Az module and not the older AzureRM module which is now deprecated (unfortunately, many online articles still refer to older Cmdlets from the AzureRM module).

Step 1: Start a PowerShell terminal with Administrator permission

Step 2: Install Az module (if not already installed)

```
Install-Module -Name AZ -AllowClobber
```

Step 3: Verify the module was installed and check the version (optional)

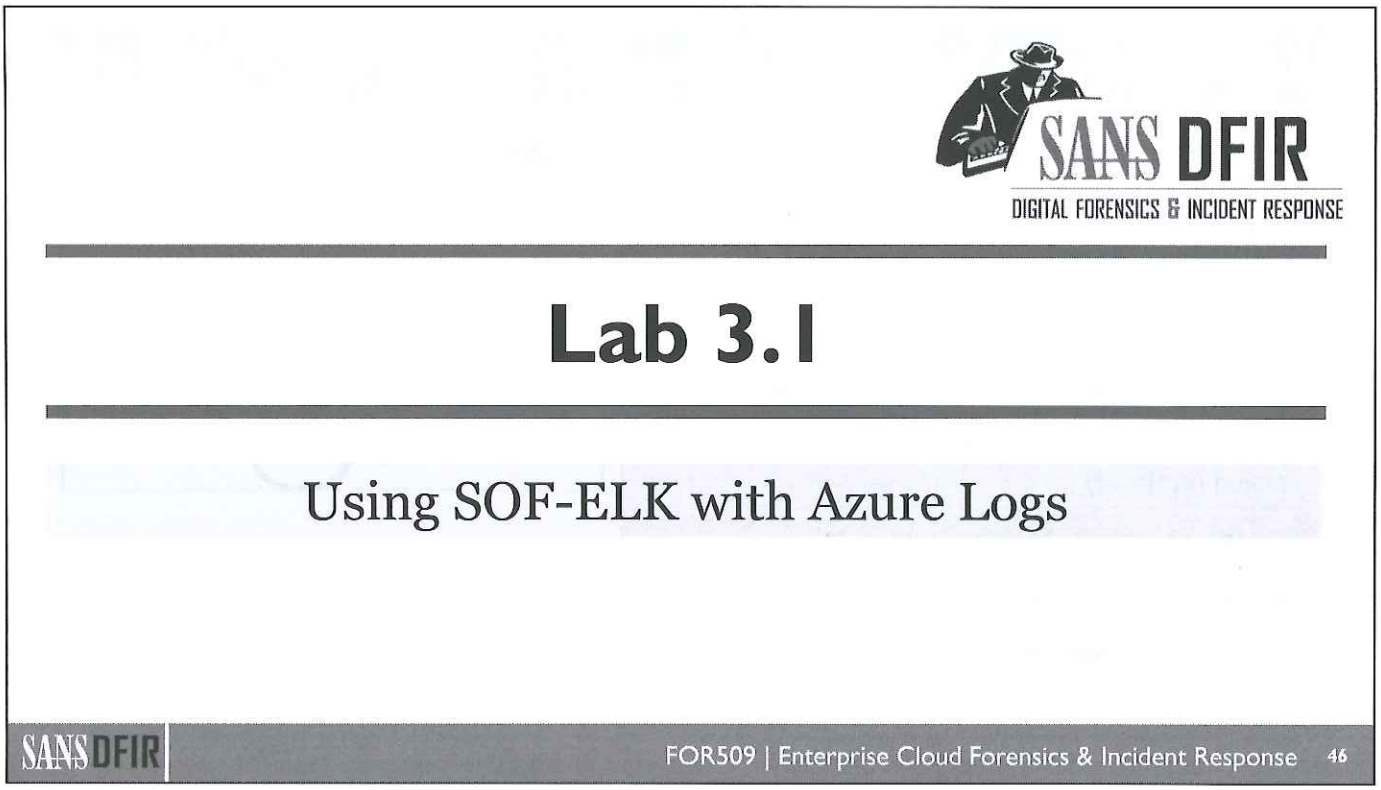
```
Import-Module Az; Get-Module Az
```

Step 4: Connect to Azure

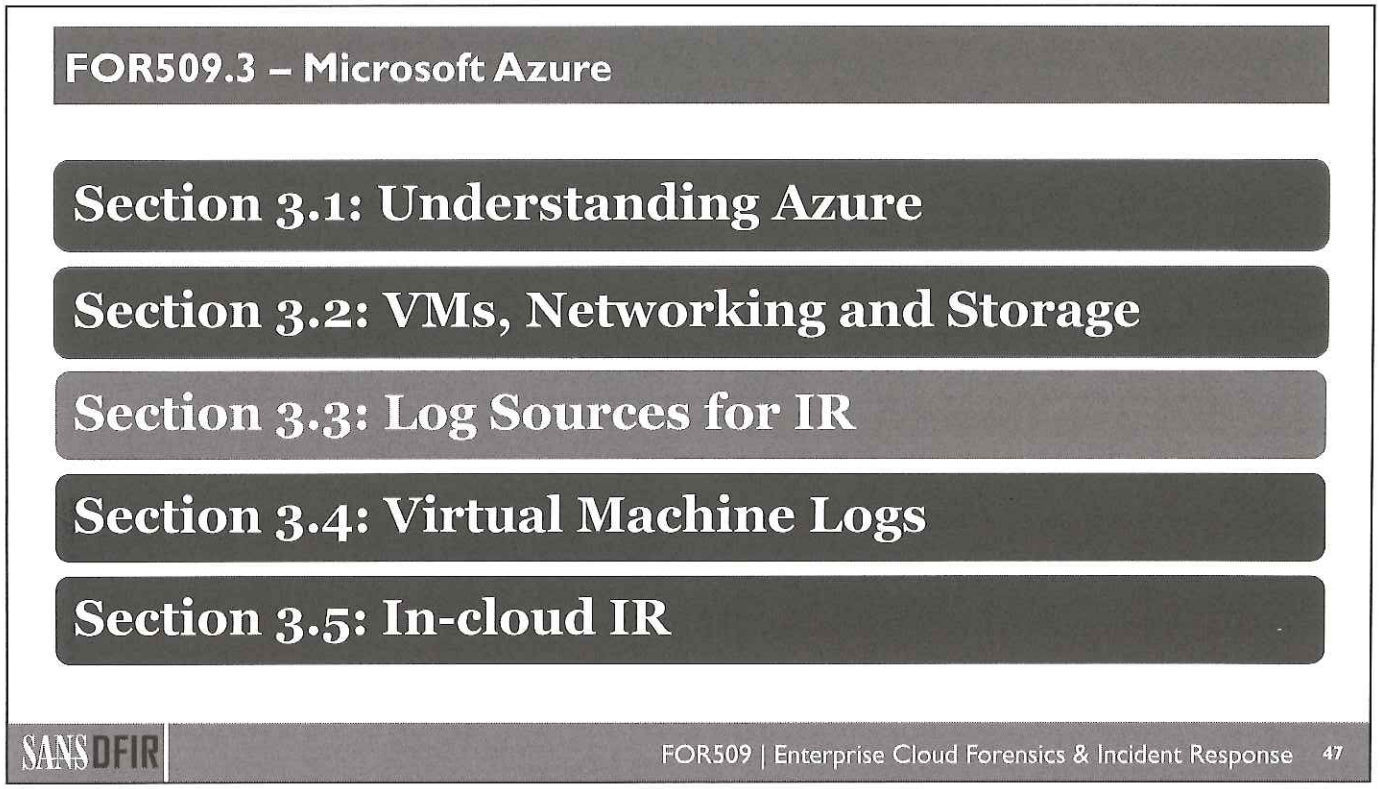
```
Connect-AzAccount
```

References:

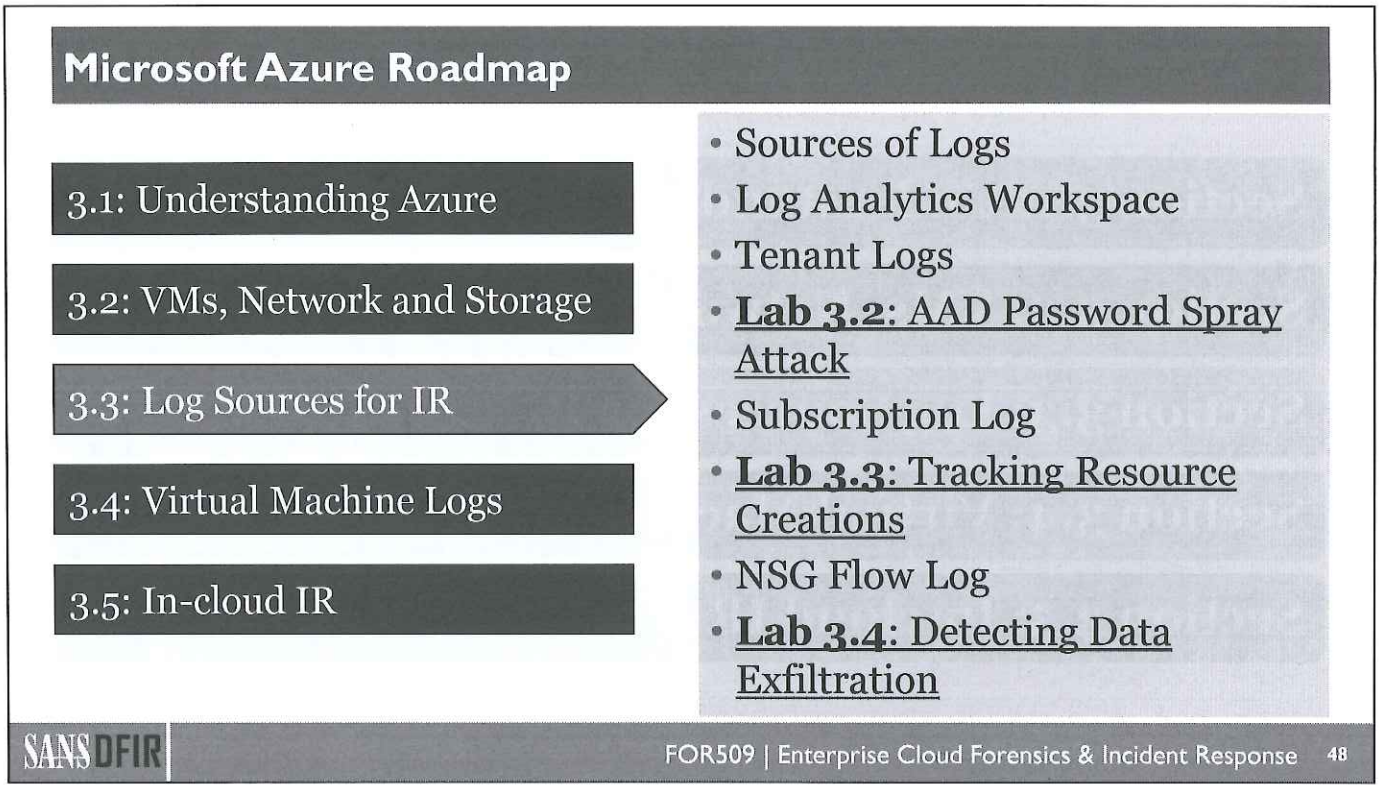
[1] https://for509.com/powershell



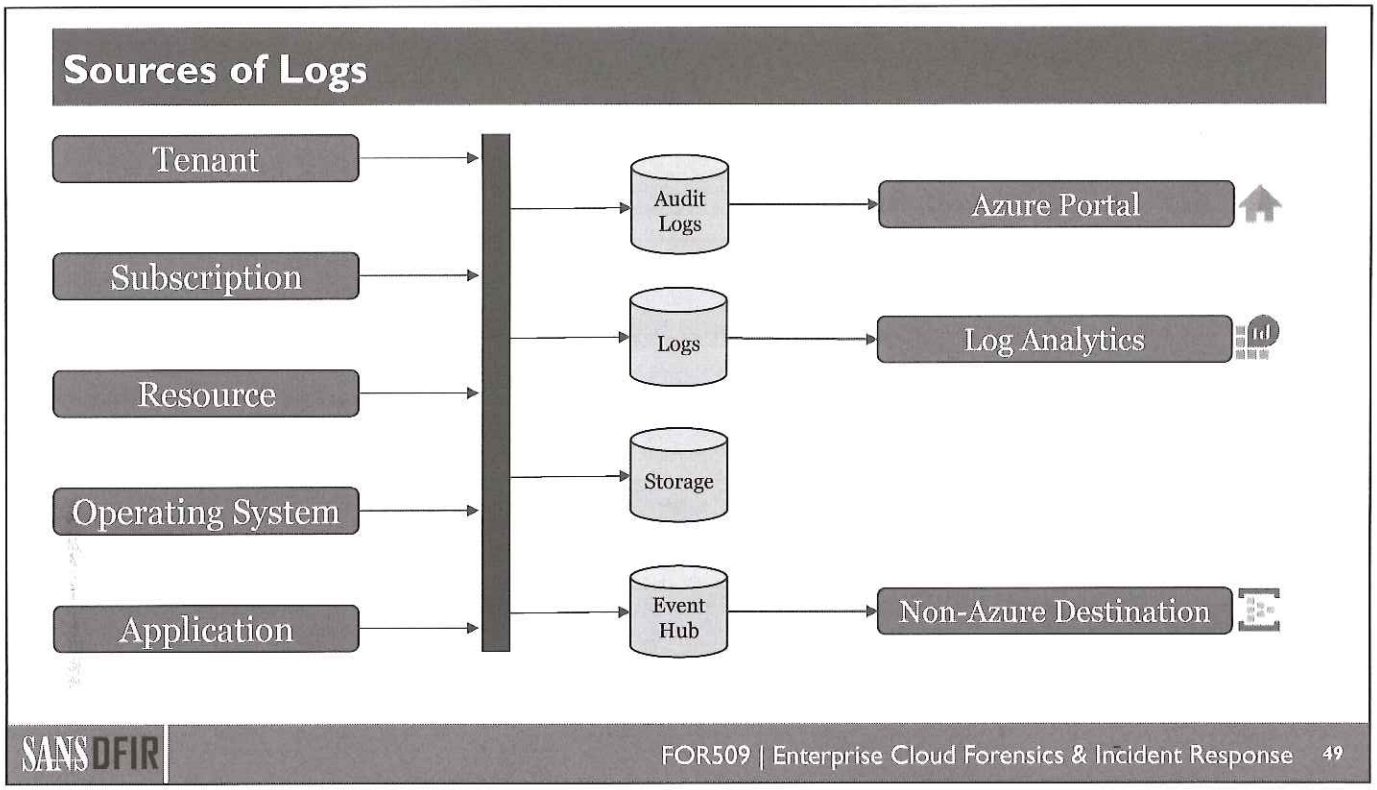
This page intentionally left blank.



This page intentionally left blank.



This page intentionally left blank.



To support your investigation, it's crucial to understand all the log sources in Azure. There are 5 sources of logs we will discuss in this section:[1]

1. Tenant logs
2. Subscription logs
3. Resource logs
4. Operating system logs
5. Application logs

Where to find these logs and how to access them is the next piece of the puzzle. Logs can be written to multiple locations at the same time which means that there may be more than one way to consume the information.

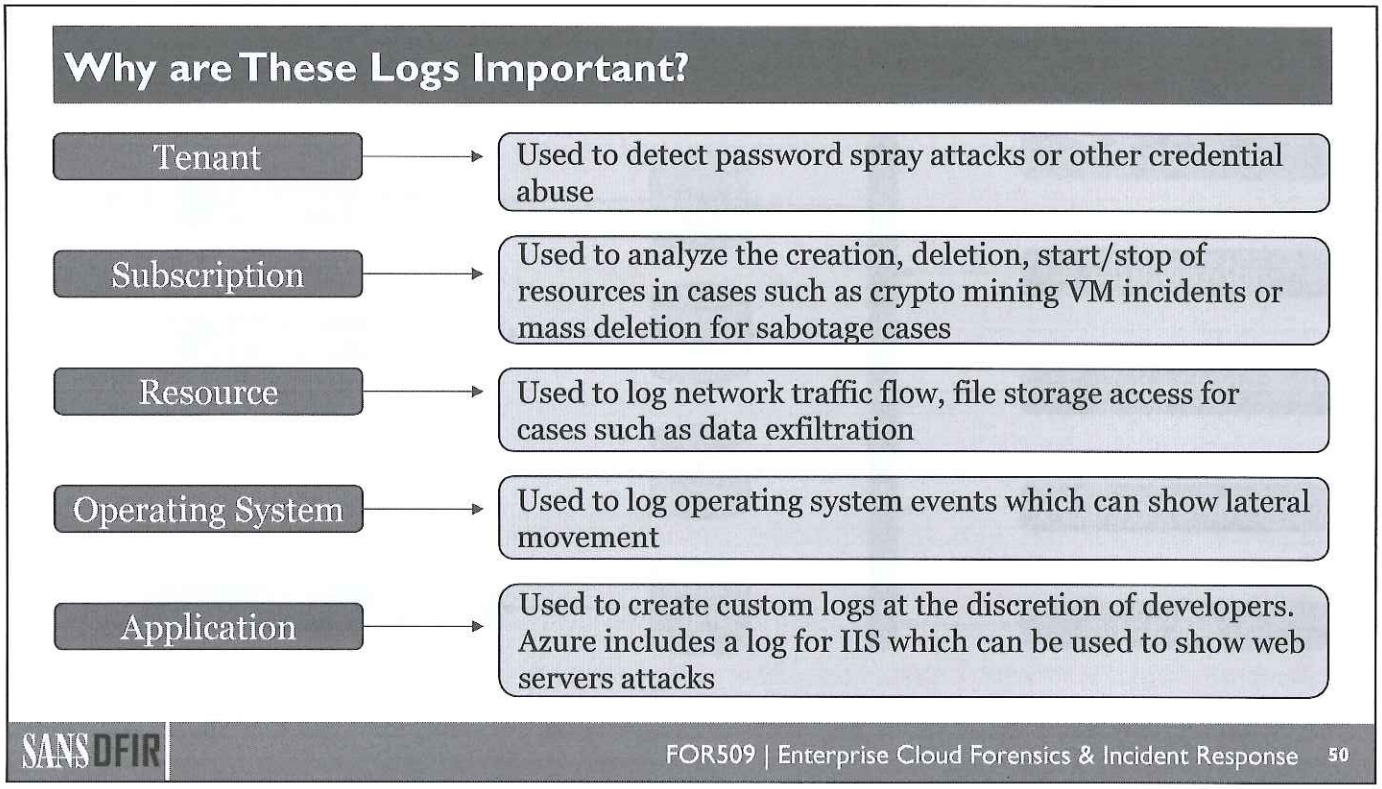
While some log sources are automatically configured by Azure, that's not the case for most of them. In order to store logs, storage must be allocated which implies a recurring cost as we previously discussed. Unfortunately, when called to perform incident response, you may find that the subscription owner didn't configure any of the optional log sources leaving you with very little information to analyze.

Educating your team on the importance of configuring these log sources is really the first step in incident response. The ideal configuration is to continuously export these logs to a SIEM so that they are not stored where a bad actor may be able to delete them.

As we examine the five log sources mentioned above, we will discuss where to find them, how to configure them and how to export them for analysis.

References:

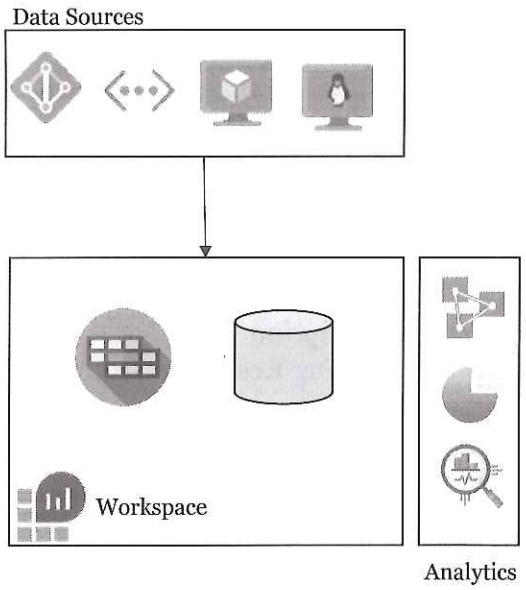
[1] https://for509.com/logsources



With so many logs available in Azure, it's easy to become overwhelmed. Investigations will normally make use of multiple logs, and these are just a few examples of the logs you could use in various DFIR situations.

Log Analytics Workspace Overview

- Workspace
  - Similar to a data lake
- End-to-end analytics: combines multiple data sources
  - Azure resources
  - Subscription logs
  - Azure Active Directory
  - Non-Azure logs
- Data organized in a table
  - Each log source has its own table
- Designed for up to 6GB/min and 4TB/day



Microsoft will store some logs by default and make them available on the Azure portal. Others will only be retained if specialized storage has been created. Microsoft calls this specialized storage the Log Analytics workspace. The Log Analytics workspace collects and aggregates logs from various data sources, both Azure-based and non-Azure. The workspace is organized into tables and each data source will create their own tables.[1]

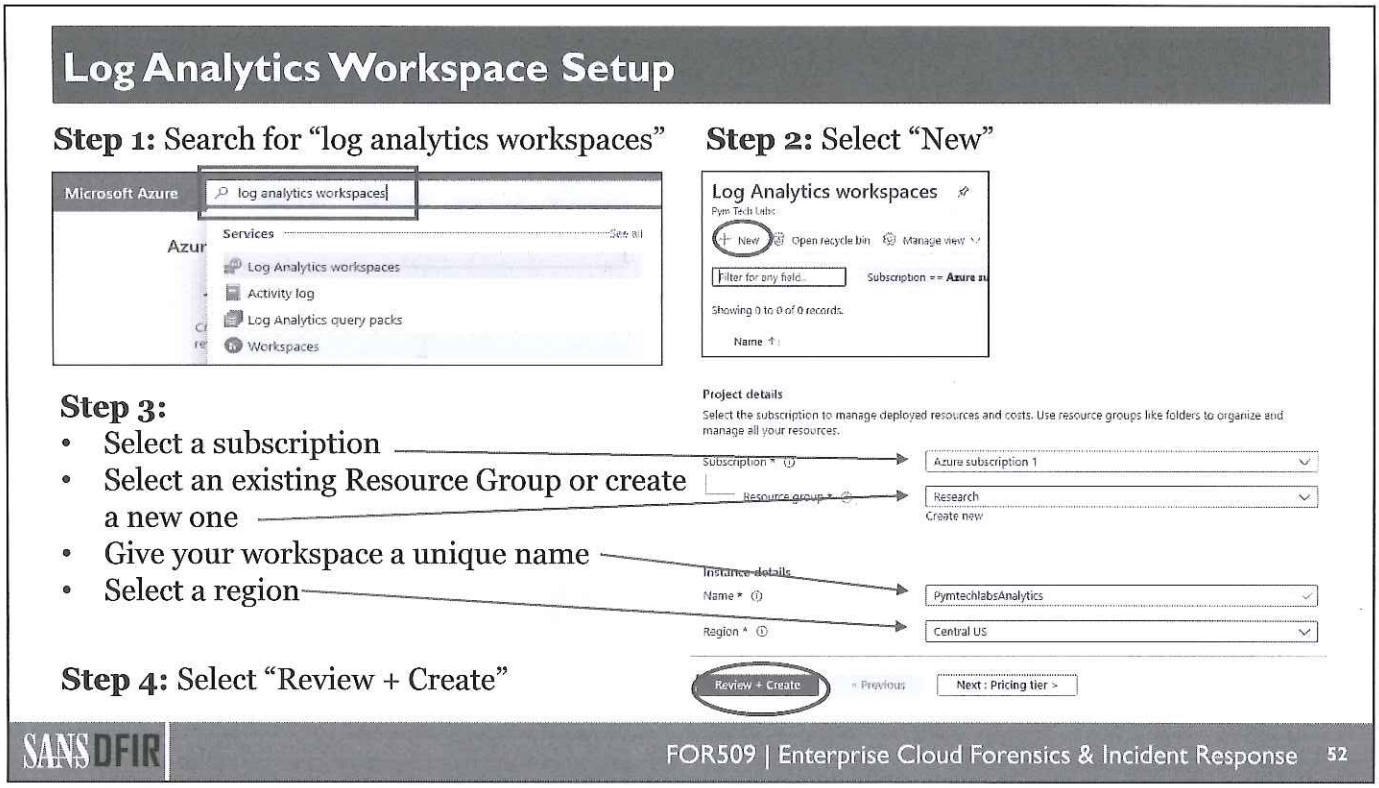
While you can create multiple workspaces to segregate logs, it's generally not needed as a default workspace can accept logs at a rate up to 6GB/min with a maximum of 4TB/day. Customized workspaces can be created with greater limits if needed.

Access control is a key concern and can be customized based on your company's security policy. Azure offers many options which are well documented in references [1] and [2].

References:

[1] https://for509.com/logdeployment

[2] https://for509.com/logaccess



Before we dig into the logs, we need to create a Log Analytics workspace. The Log Analytics workspace can be created via the Azure portal[1], Azure CLI[2], or PowerShell[3]. You can reference the Microsoft documentation if you wish to create the workspace using the CLI or PowerShell. In this slide, we will use the Azure portal.

Step 1: Sign into the Azure portal & Search for the "log analytics workspaces" service in the search bar

Step 2: Select "New"

Step 3: Enter the information requested: Subscription, Resource Group, Name, and Region

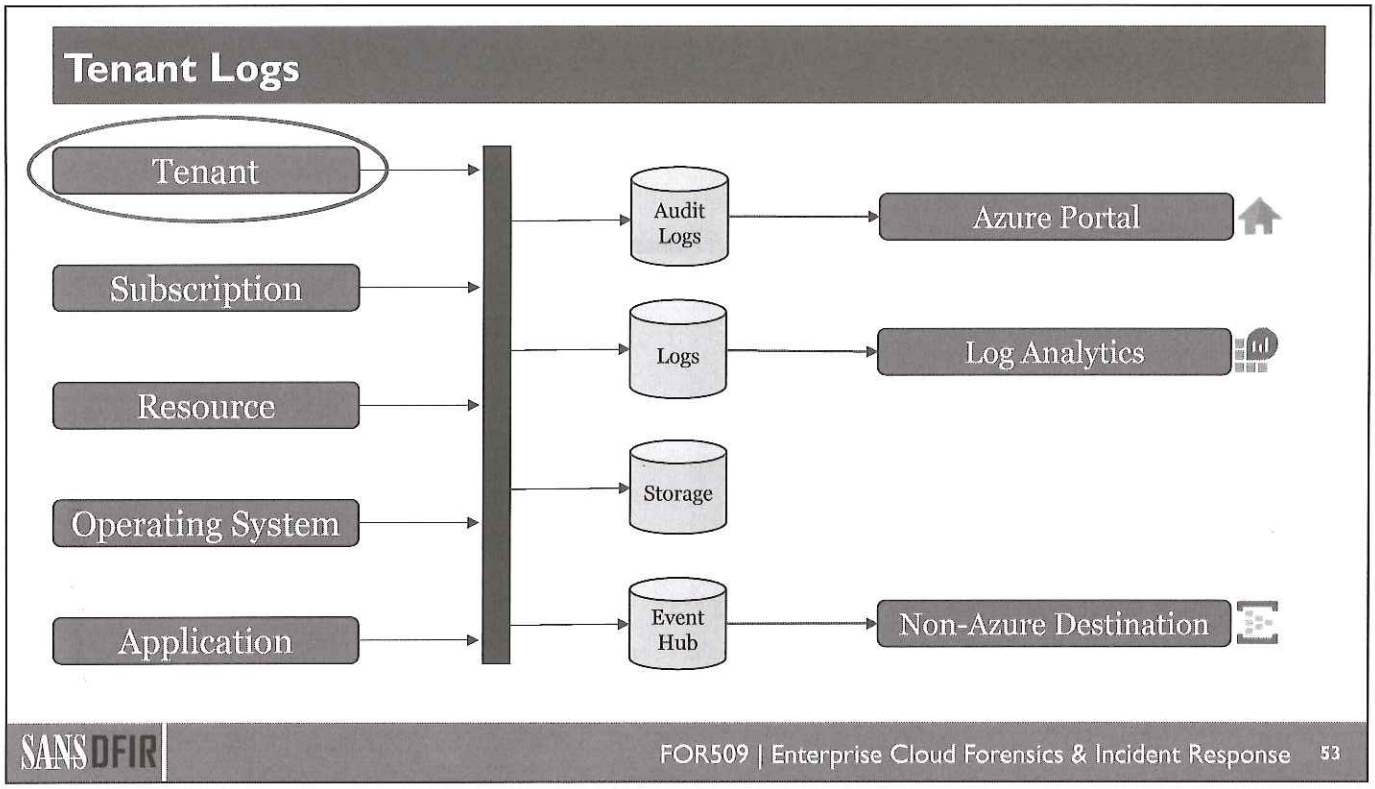
Step 4: Select "Review + Create"

References:

[1] https://for509.com/law-portal

[2] https://for509.com/law-cli

[3] https://for509.com/law-powershell



We will start our exploration of the Azure log sources with the tenant logs.

The tenant log contains information about operations conducted by tenant-wide services. This is where you will find the Azure Active Directory (AAD) log.[1] The AAD log contains audit logs, sign-in logs, and provisioning logs. The provisioning logs are still in preview.[2] The sign-in logs require an Azure AD P1 or P2 license.

> In order to export Sign-in data, your organization needs Azure AD P1 or P2 license. If you don't have a P1 or P2, start a free trial.

References:

[1] https://for509.com/logs-aad

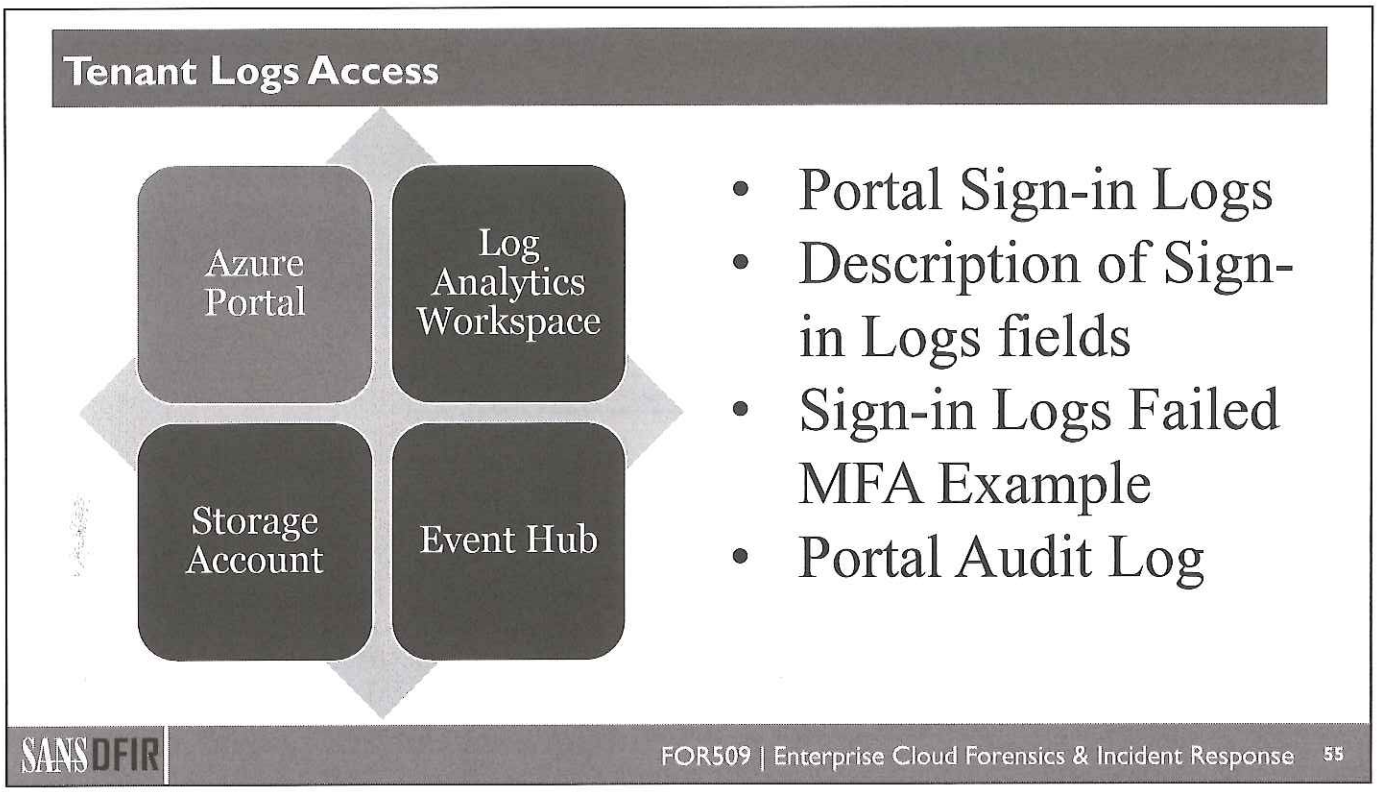
[2] https://for509.com/logs-provisioning

Tenant Logs Agenda

- Sources of Logs
- Log Analytics Workspace
- Tenant Logs
- Lab 3.2: AAD Password Spray Attack
- Subscription Log
- Lab 3.3: Tracking Resource Creations
- NSG Flow Log
- Lab 3.4: Detection Data Exfiltration

- Azure Portal
  - Sign-in logs
  - Audit log
- Log Analytics Workspace
- Storage Account
- Azure Storage Explorer
- Import into SOF-ELK
- Event hubs
- Graph API

This page intentionally left blank.

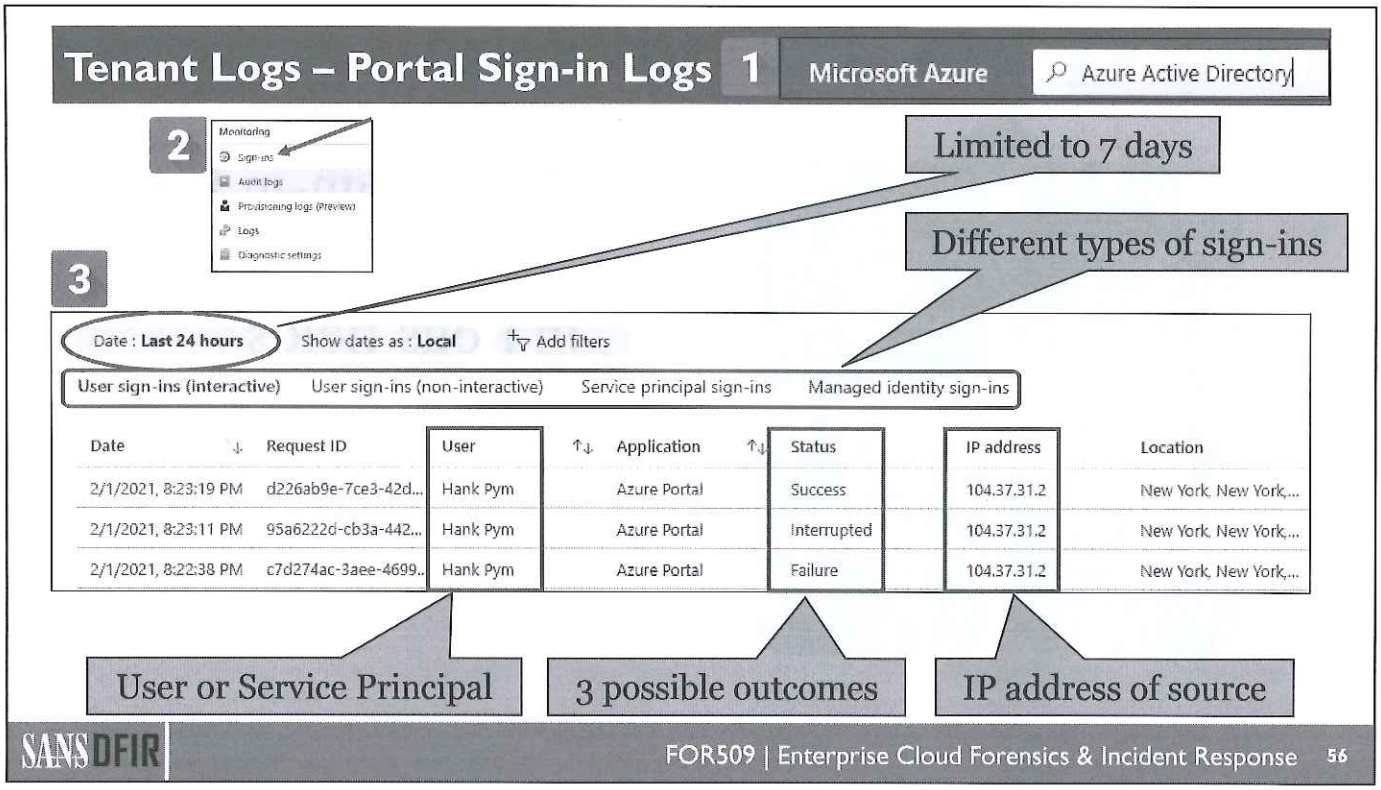


There are four actions you can take with these logs:

1. View them directly on the Azure portal.
2. Store them in a log analytics workspace.
3. Send them to a storage account for archival.
4. Send them to a SIEM by using the event hub.

These four options will be the similar for the other types of logs: subscription, resource, operating system, and application.

Note: Since there are multiple logs that contains sign-in information, we will refer to **logs** rather than log in this section.



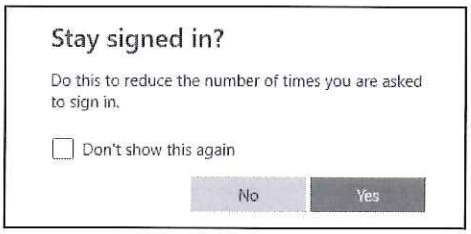
The portal is a quick and easy way to check the sign-in log. Unfortunately, it's limited to the last 7 days.

The key fields are the date, user, status, and IP address. All the way to the right (not shown on the slide), there is a column that will tell you if multi-factor authentication was used.

Notice at the top the 4 tabs for the different kinds of sign-ins: user, user non-interactive, service principal, managed identity.

You will see 3 possible status:

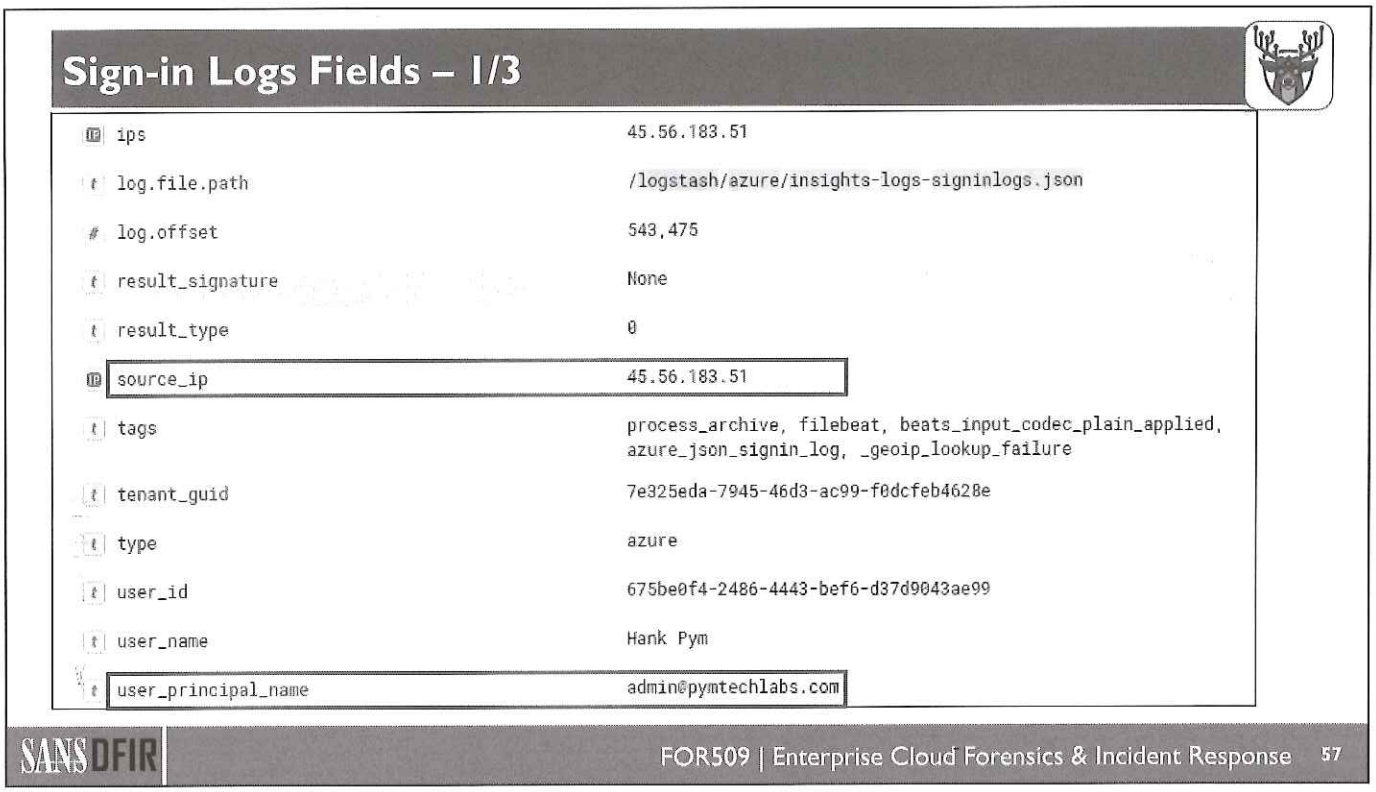
- Success
- Failure
- Interrupted



Success and failure are self-explanatory. Interrupted is due to the "Stay signed in?" window.[1] The documentation states that if a user "abandons the sign-in attempt" then the status of "Interrupted" will be recorded. Testing indicates that replying "No" may also generate the same status.

References:

[1] https://for509.com/staysignedin

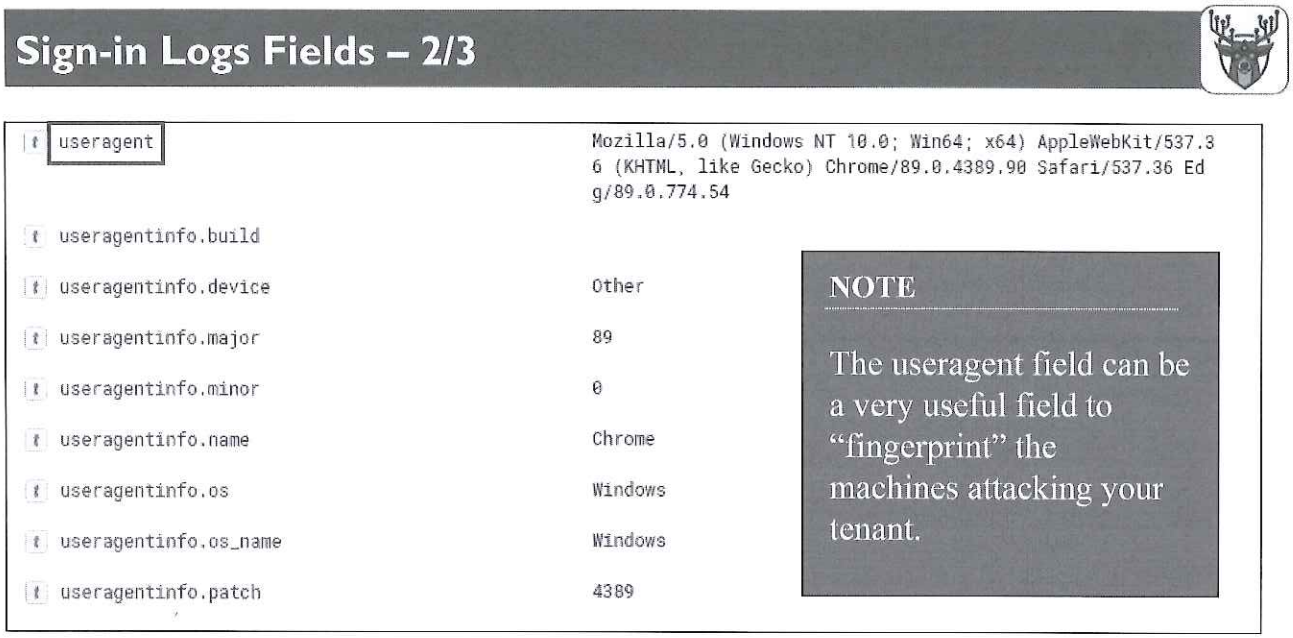


The event as recorded by Microsoft contains a large number of fields. When we import the data to SOF-ELK via the Logstash ingestion script, we filter the fields to import the most important ones only. This is an important step in order to maintain a high speed of ingestion and performance of SOF-ELK.

The next few slides will show the key fields. In SOF-ELK, you will see every one of these fields for each event.

Field	Description
ips/source_ip	IP address of the system accessing Azure
log.file.path	Name of the file ingested in SOF-ELK
tenant.guid	Unique identifier for your tenant
user_id	Guid of the login user
user_name	Display name of the login user
user_principal_name	Azure Active Directory name of the login user

Sign-in Logs Fields – 2/3

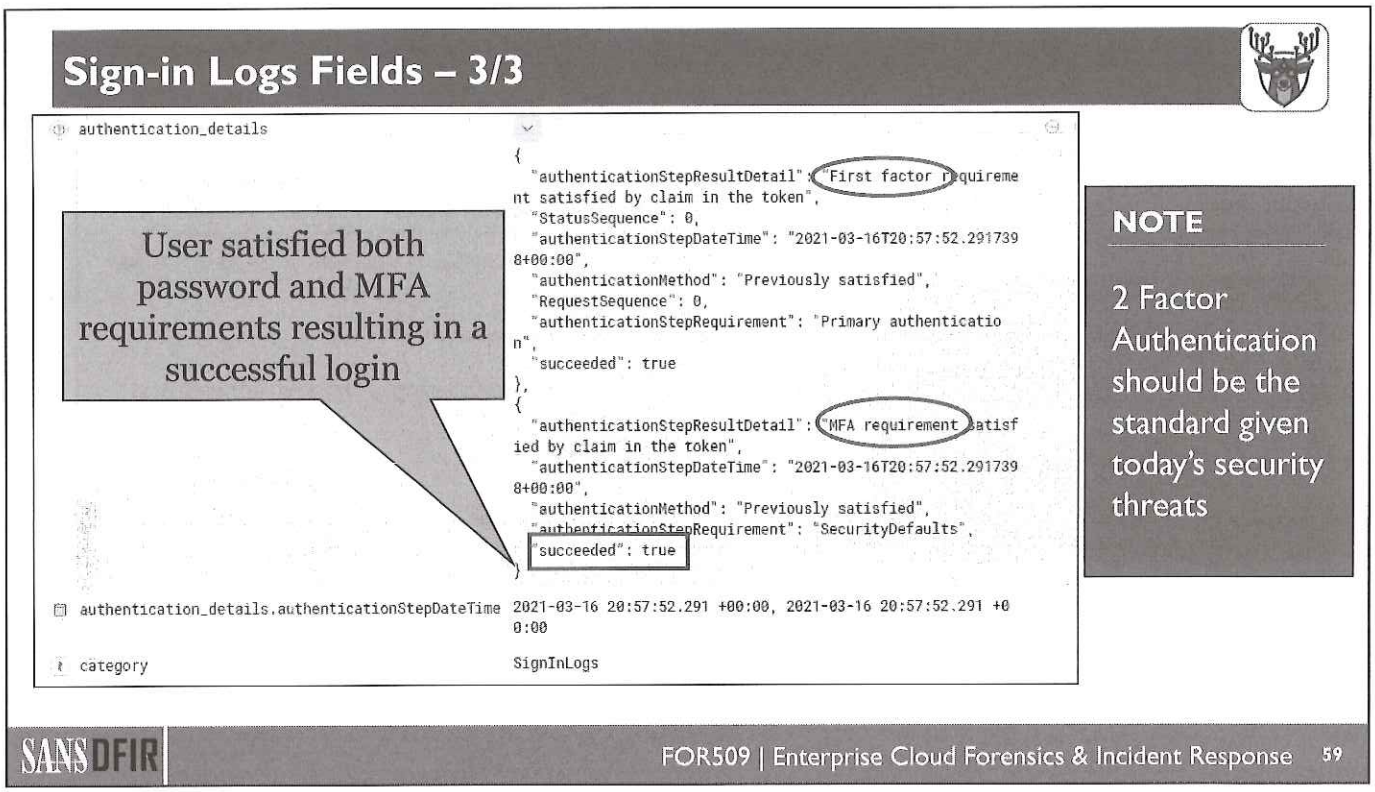


useragent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90 Safari/537.36 Edg/89.0.774.54
useragentinfo.build	
useragentinfo.device	Other
useragentinfo.major	89
useragentinfo.minor	0
useragentinfo.name	Chrome
useragentinfo.os	Windows
useragentinfo.os_name	Windows
useragentinfo.patch	4389

NOTE

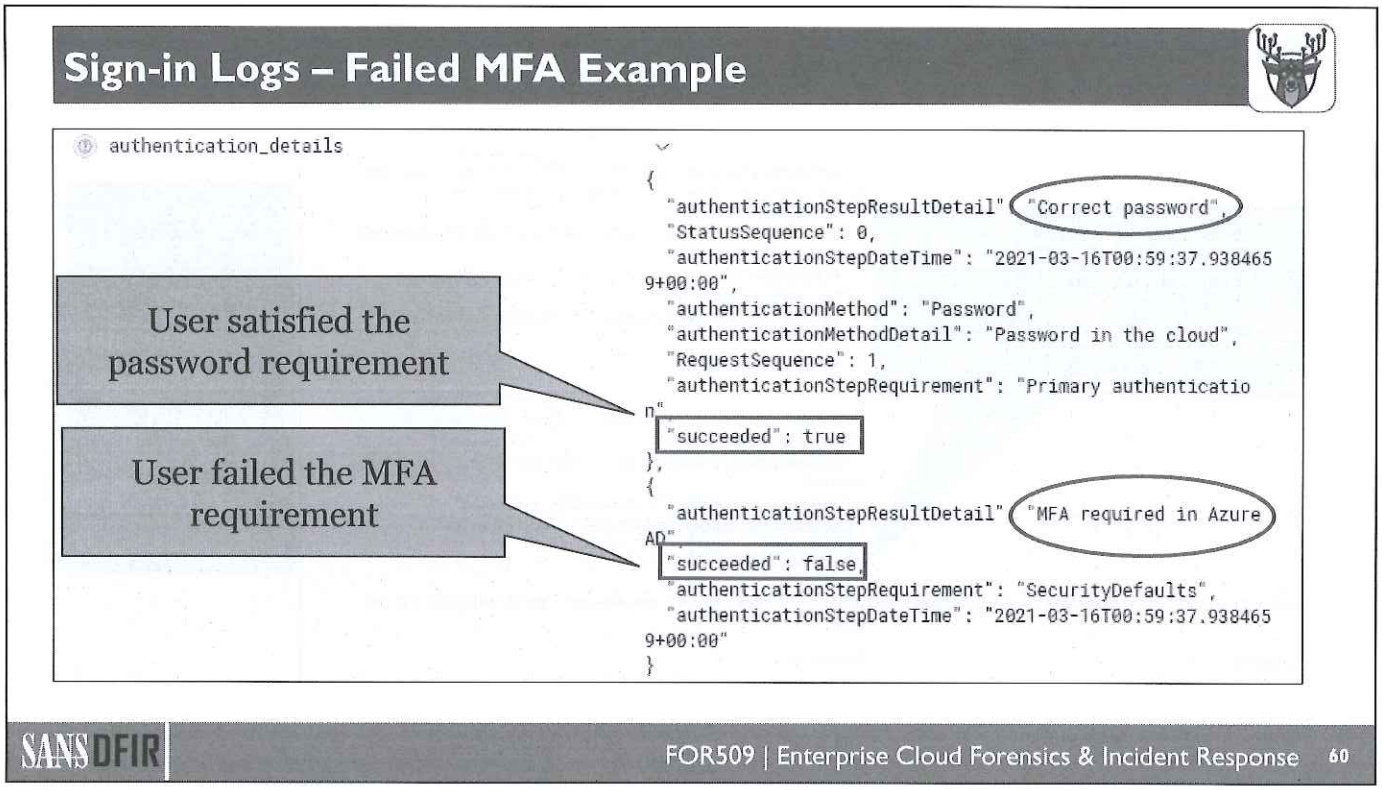
The useragent field can be a very useful field to "fingerprint" the machines attacking your tenant.

This set of fields breaks down the browser user agent. This can be a very useful field in your investigation in order to "fingerprint" the machines trying to access your tenant.

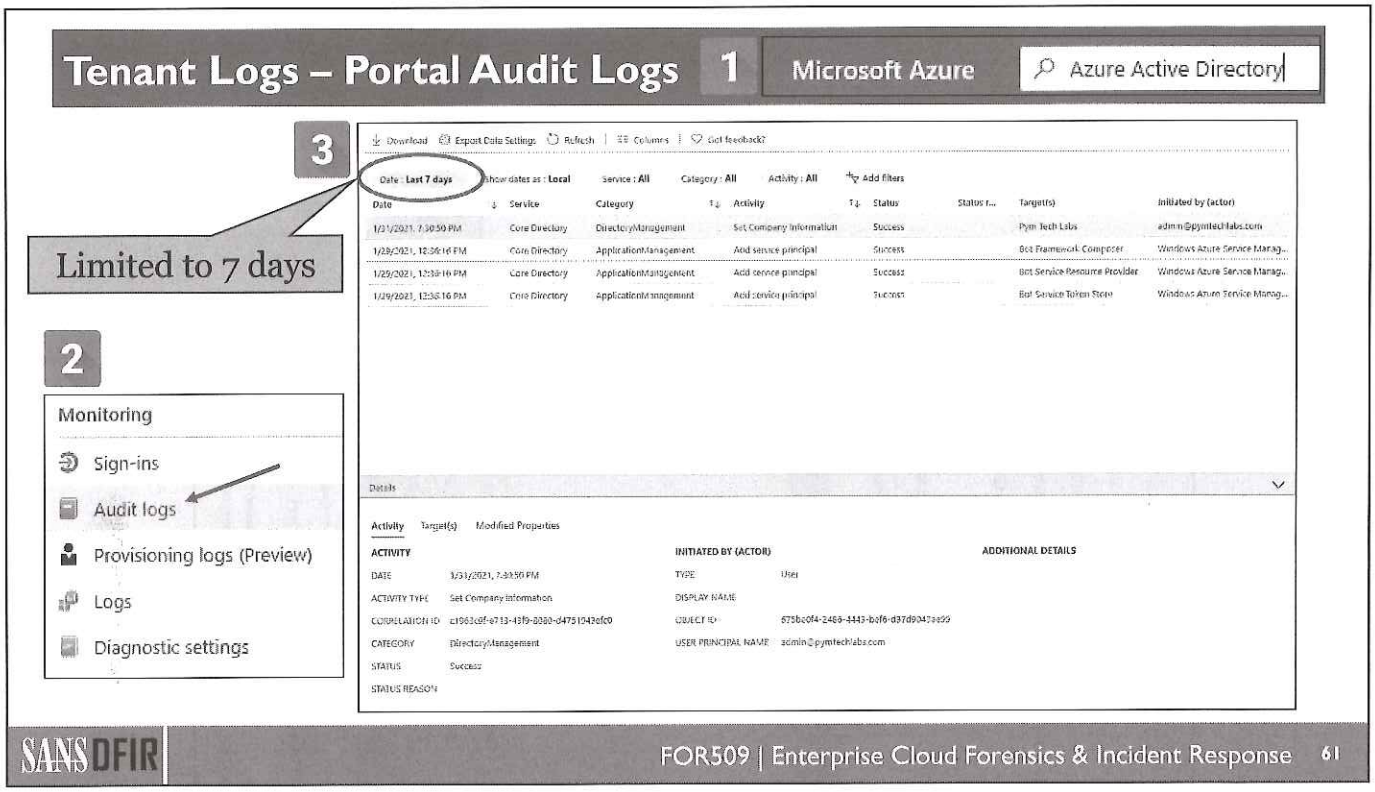


The authentication details field is quite interesting as it provides information about the authentication method used for this specific login. In a tenant where 2-factor authentication has been implemented you will see if the MFA token was accepted.

An interesting search is for successful login with failed MFA. This may indicate that the user's credentials have been compromised, but the bad actor isn't able to fulfill the MFA requirements.



This is an example where the user satisfied the password requirement but failed the MFA requirement. While these will happen frequently due to user error, repeated failures on a large number of accounts coming from the same IP address should be cause for concern.

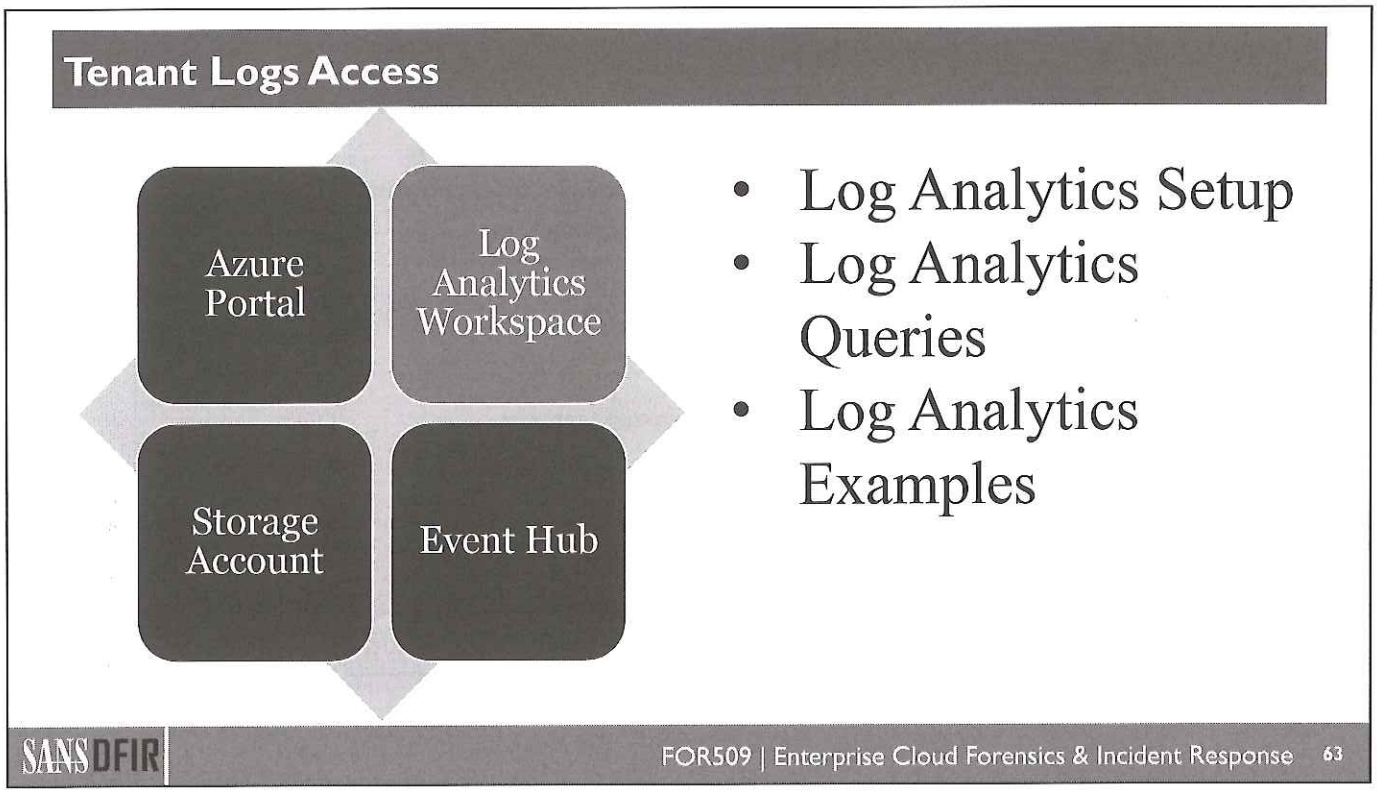


The other log available in the portal is the audit log. It's also limited to the last 7 days.

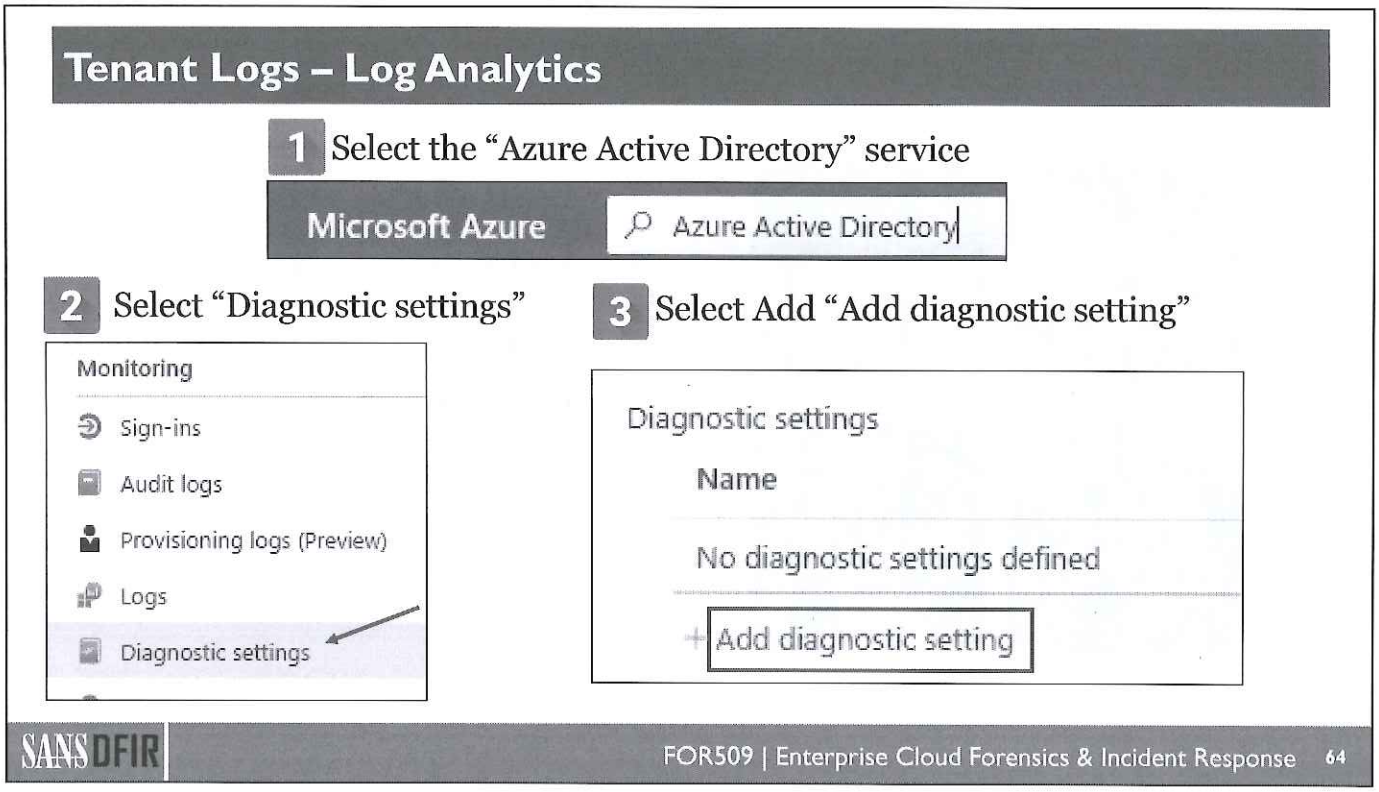
The audit log will show tenant-wide actions such as configuration changes to AAD.

Other than a quick check of recent events, the portal is far from the ideal place to check the tenant logs. A much better place is the log analytics workspace which we will configure in the next slides.

Limited to 7 days



This page intentionally left blank.



While the portal is convenient for a quick search, you have seen that it's very limited. The real power is in the log analytics workspace.

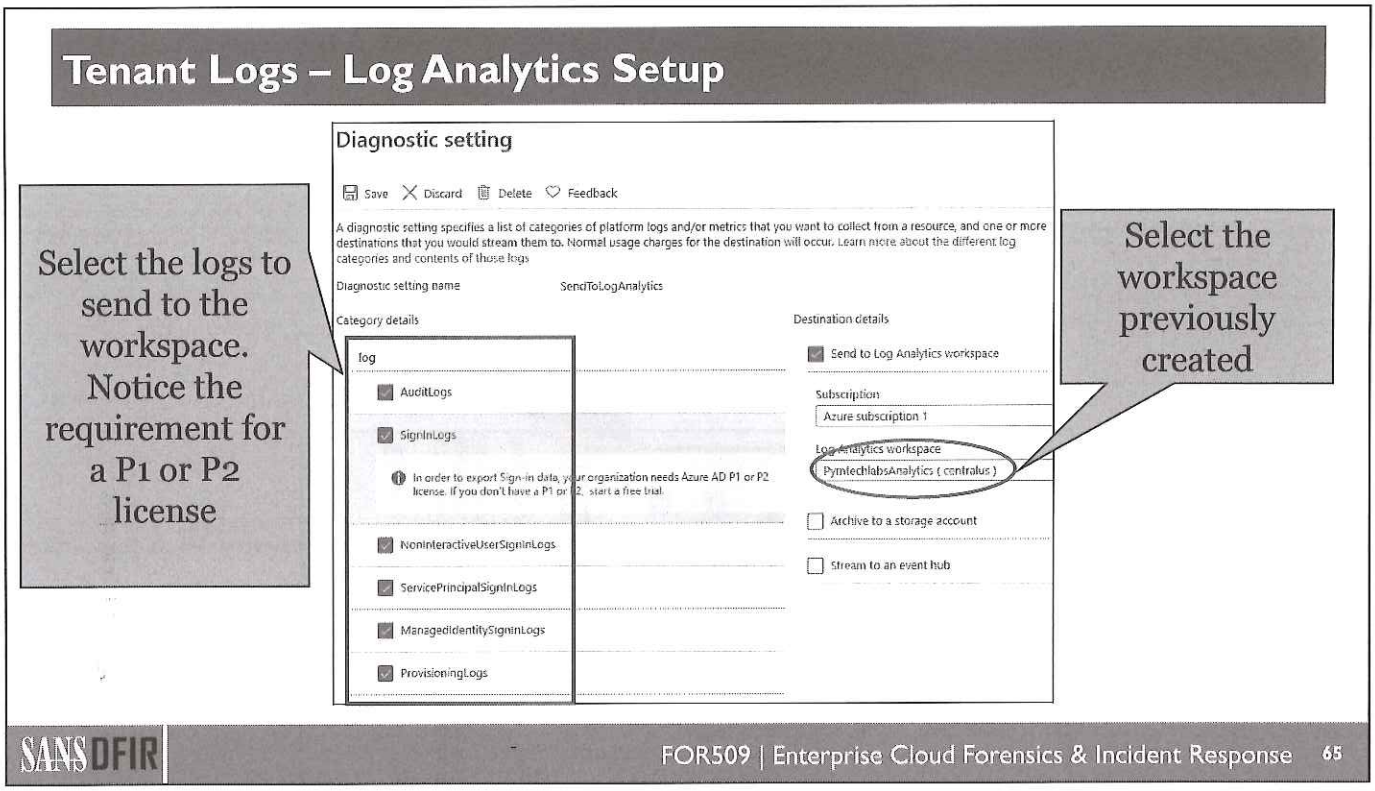
In this section about the tenant logs, we will describe how to send the AAD logs to the log analytics workspace. In later slides, we will send other logs (such as subscription, resources, etc.) to the same log analytics workspace. This provides you a single location to see all your logs which is very convenient.

For the AAD logs, you will need to complete the following steps in the Azure portal:

- **Step 1**: Search for and select the "Azure Active Directory" service
- **Step 2**: On the left menu select "Diagnostic settings"
- **Step 3**: Select "Add diagnostic setting"

The next screen will allow us to select our log analytics workspace.

These steps may also be completed via the Azure CLI or PowerShell which you may find unnecessarily complicated compared to the Azure portal.



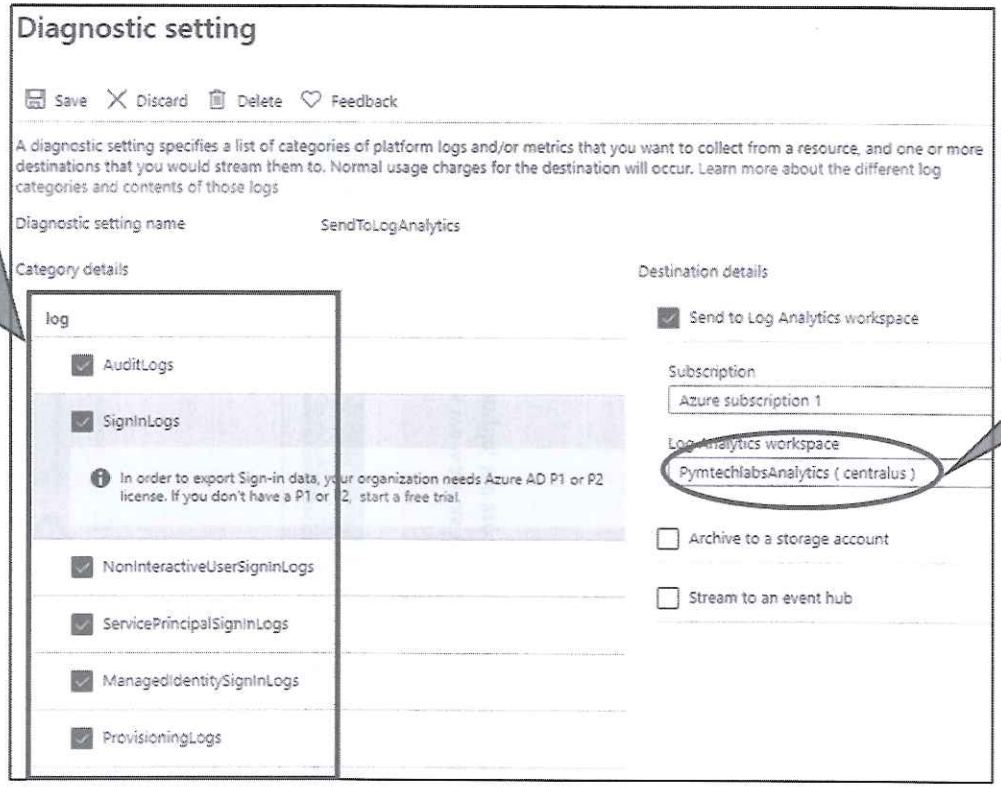
In the second step, we need to select both the AAD logs we want as well as the log analytics workspace where we want to send the selected logs to. For this example, we are using the log analytics workspace we created in prior slides: PymtechlabsAnalytics.

AAD stores information in 3 different logs:

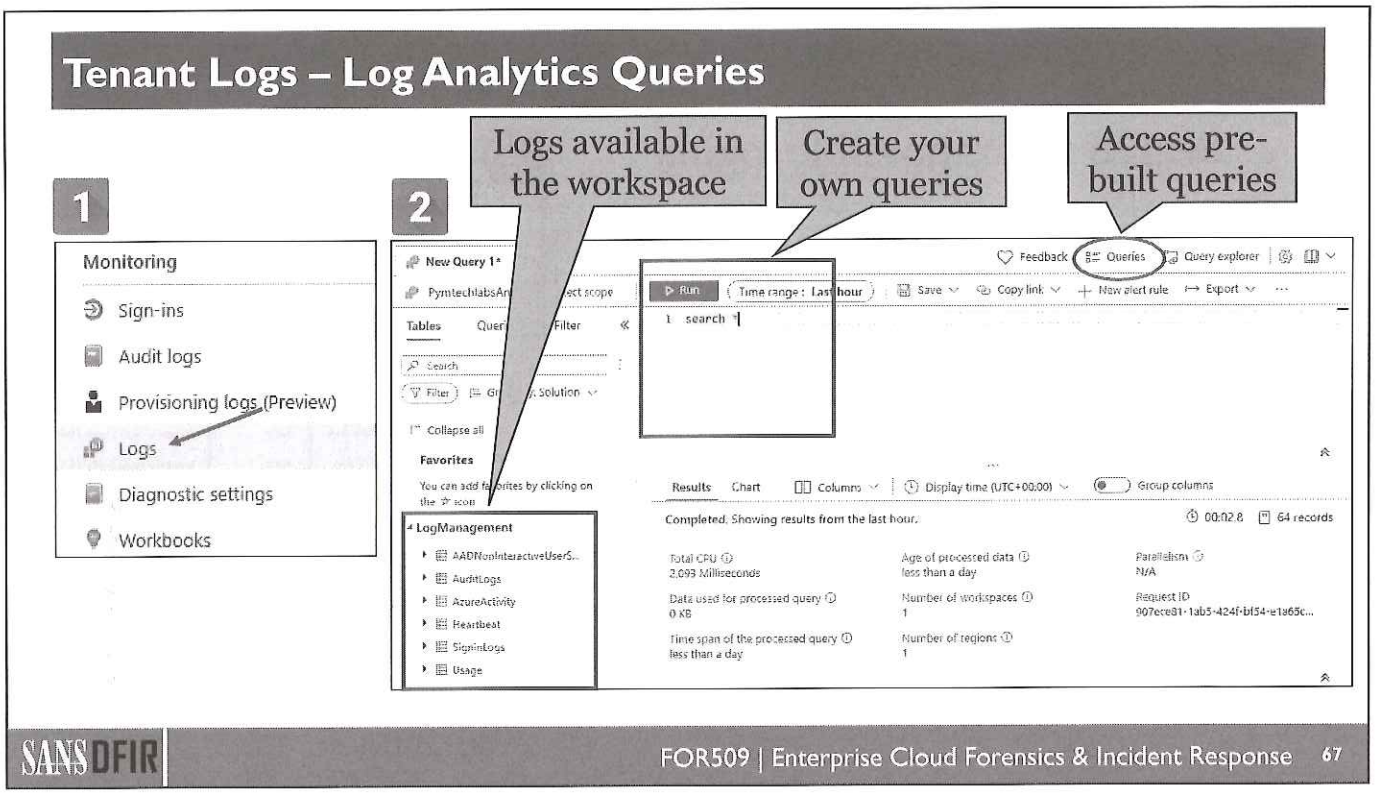
- **Audit logs** include adding or removing users, apps, groups, roles and policies.
- **Sign-in logs** include usage of managed applications and user sign-in activities. These are further sub-divided into:
    - Non-interactive user sign-in
    - Service principal sign-in
    - Managed identity sign-in
- **Provisioning logs** include activity about users, groups, and roles that are provisioned by the Azure AD provisioning service. This log is still in preview.

It might be tempting to save everything, but in a large organization the storage requirements may get very costly. An alternative solution is to store the logs in a storage account as we will see on the next slide.

Select the logs to send to the workspace. Notice the requirement for a P1 or P2 license



Select the workspace previously created



Now that we configured our AAD logs to be sent to the log analytics workspace. We can now query these logs using the Kusto Query Language (KQL) which is very similar to SQL.[1] Learning KQL is beyond the scope of this class, but here is a simple example:

```
SigninLogs
| where TimeGenerated > ago(1d)
| where ResultType == 0
```

The first line of the query specifies the log we want to search.

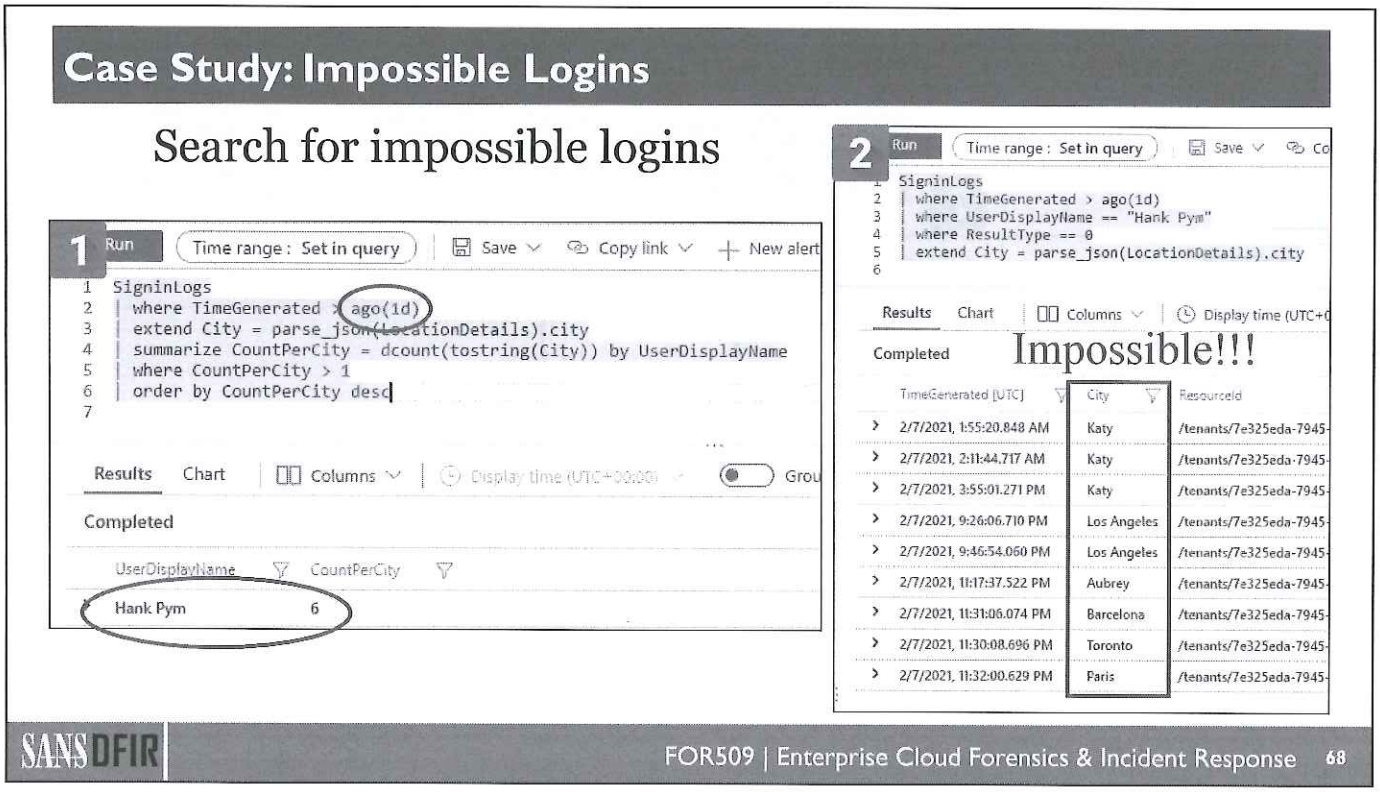
The second line limits the query to the last 24 hours.

The third line limits the query to successful login

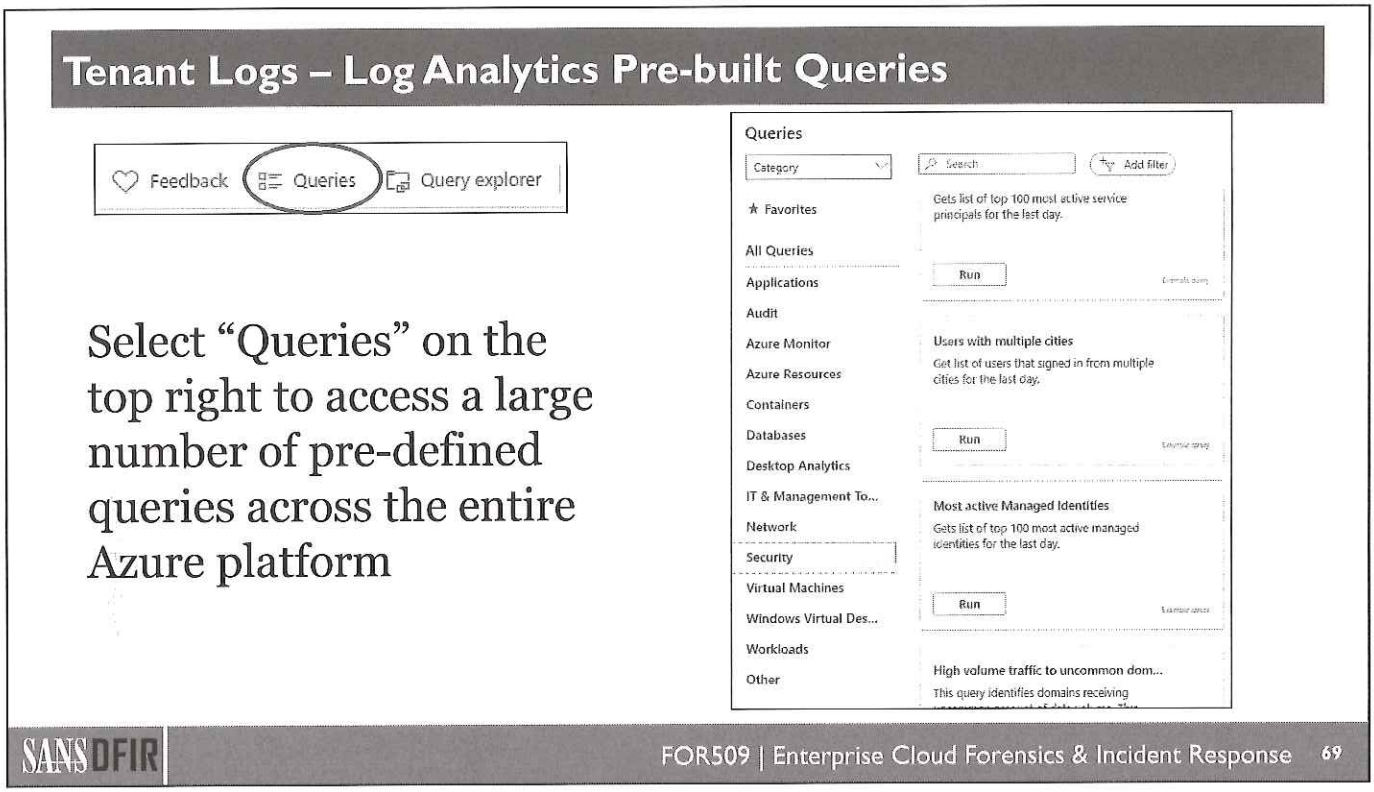
KQL is a powerful language which can help you analyze and visualize your data easily.

References:

[1] https://for509.com/kql-overview



Let's try a KQL search to see if anyone has logged in from multiple cities too quickly (also known as an impossible login). As you can see from the first query, Hank Pym logged on from 6 different cities in the last 24 hours. That seems abnormal. The second query searches for the name of these cities and shows us the login times as well. Clearly, we may have an issue with Hank Pym's account.

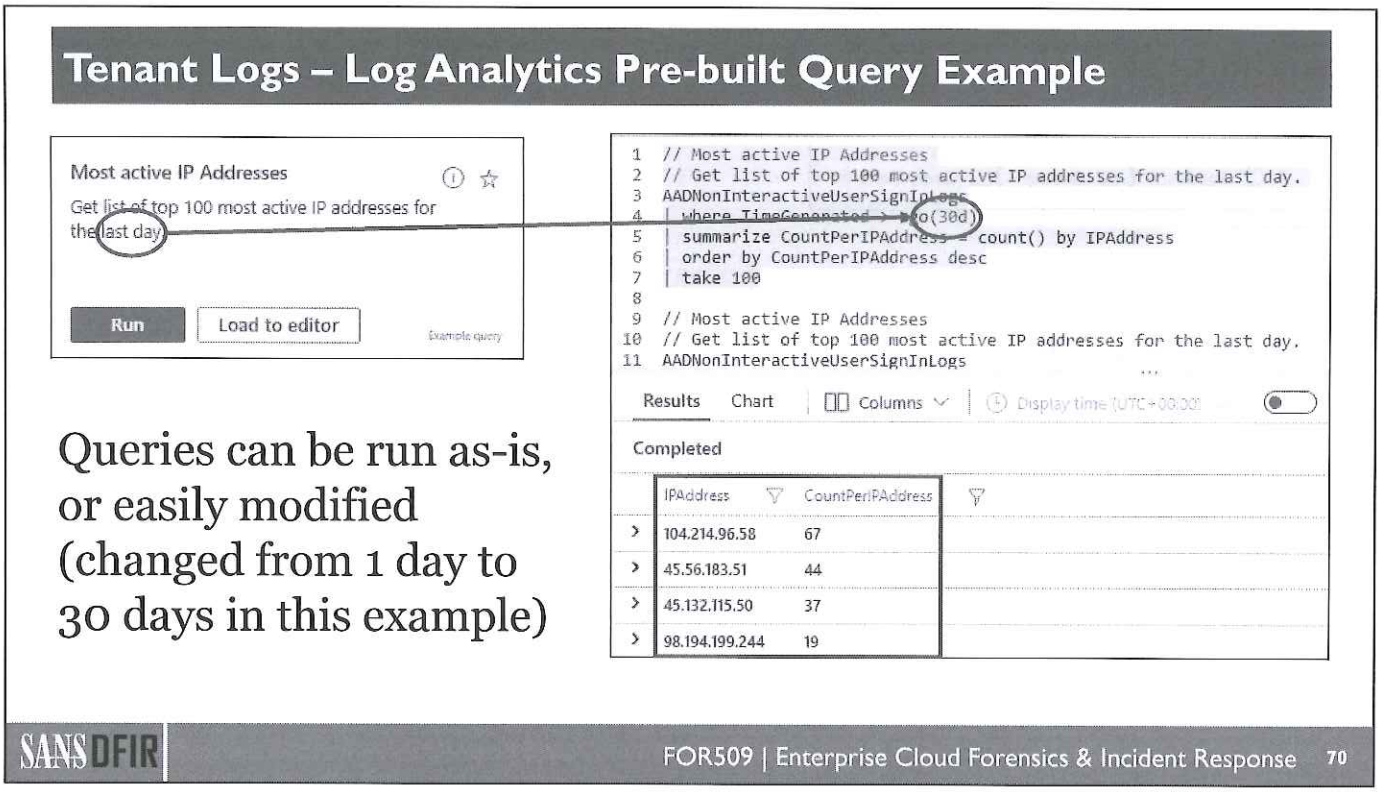


Microsoft provides a number of pre-defined queries for the various Azure resources. These serve as a great basis for developing your own queries. Obviously, you must first configure the data source to send its logs to the log analytics workspace for any of these queries to work.

More information about Log Analytics is available in reference [1].

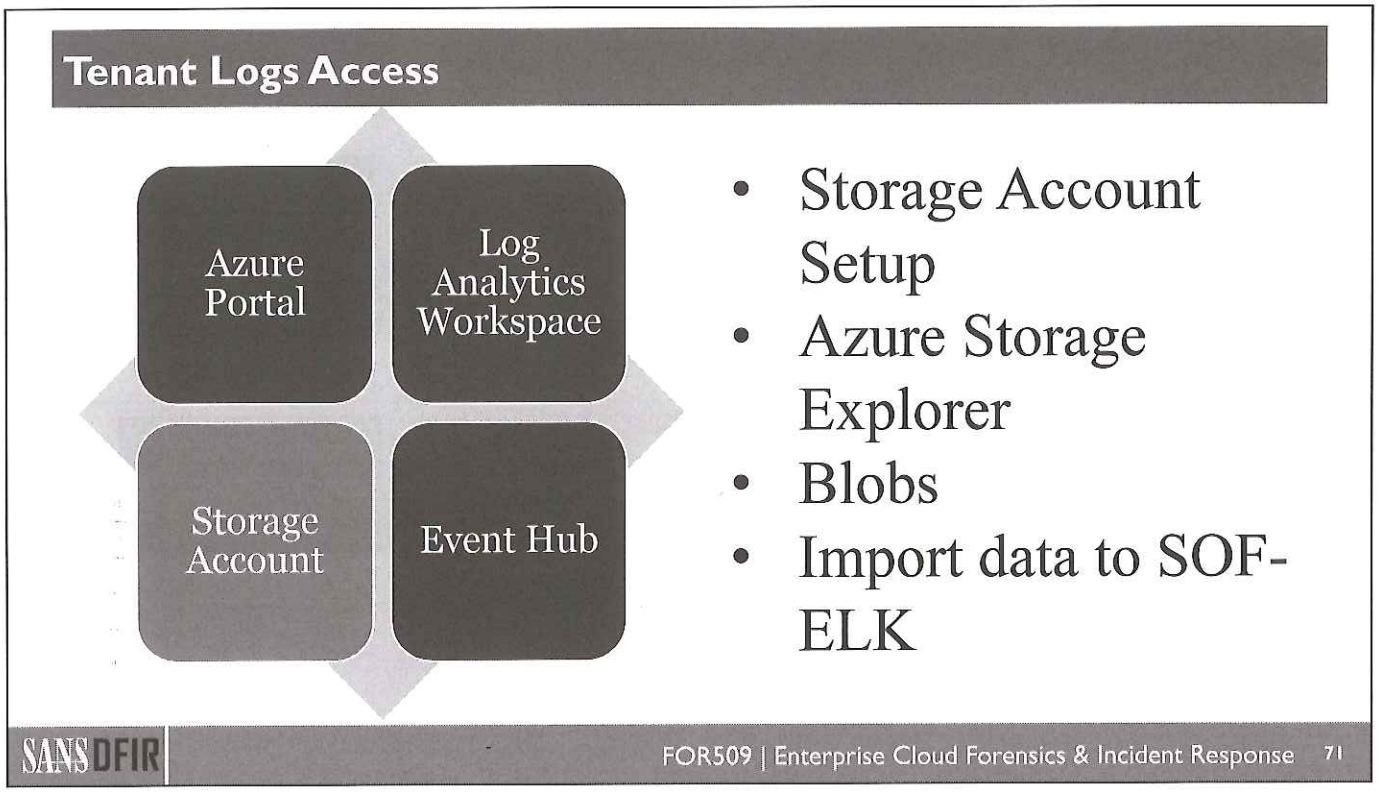
References:

[1] https://for509.com/loganalyticsoverview

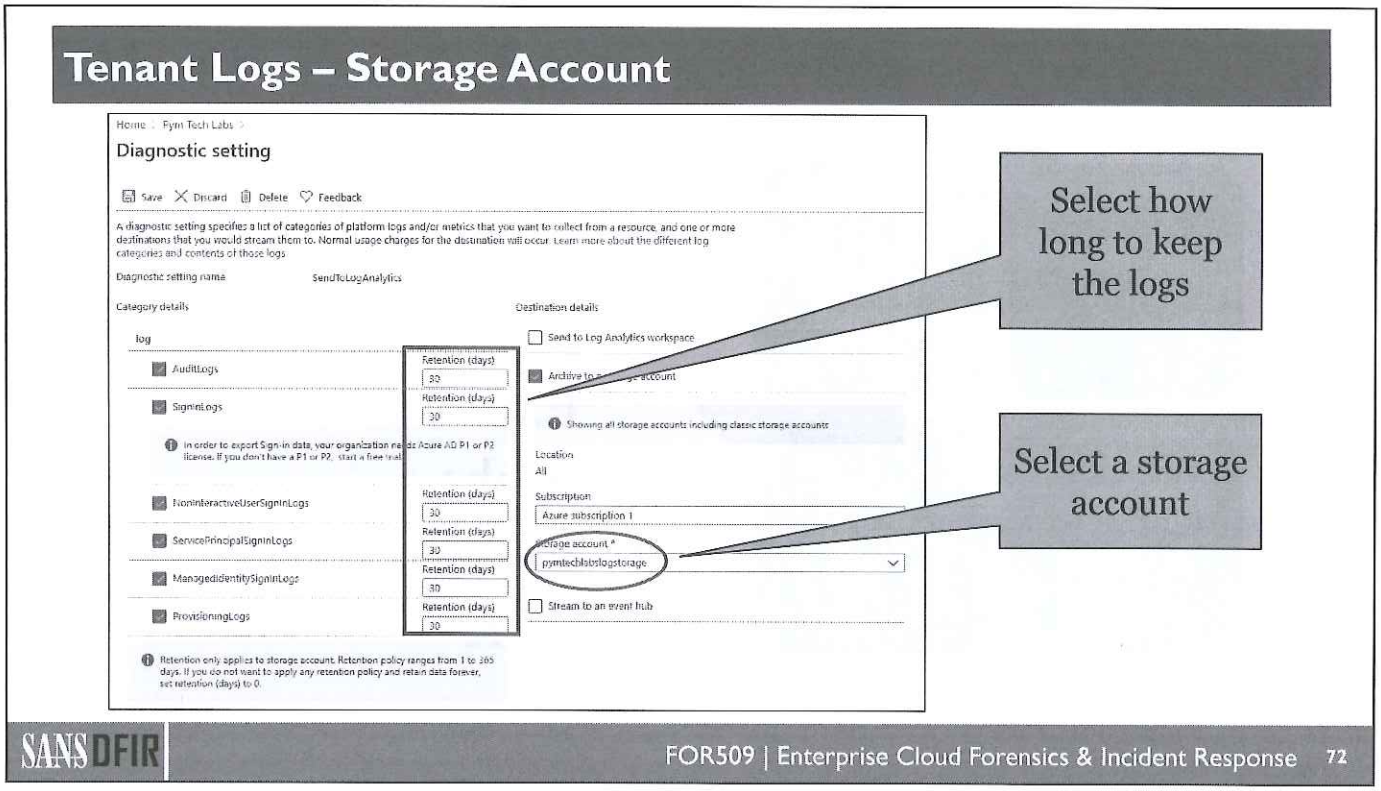


In this example, we selected a pre-defined query to search for the most active IP addresses. The query can be easily modified to meet your parameters. As an example, we changed this query from only looking at the last 24 hours to searching for the last 30 days.

Pre-defined queries are a great basis for creating your own queries.



This page intentionally left blank.

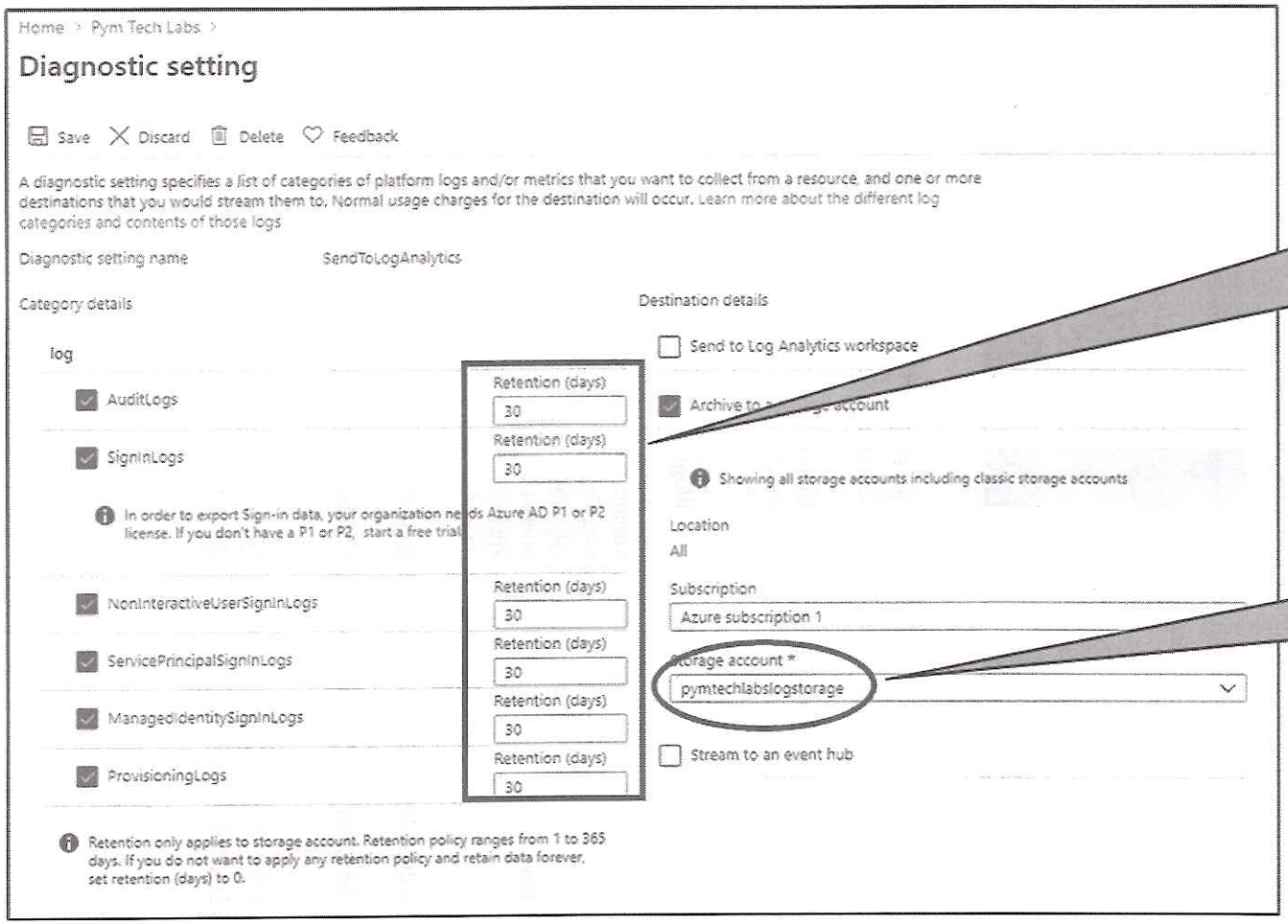


The log analytics workspace is a great way to store and view the logs. However, you may want to save the logs for an extended period of time and export them to other tools. To achieve that goal, you will need to export the logs to a storage account.

Going back to the diagnostic setting, we configure the tenant logs to be send to a storage account called *pymtechlabslogstorage*.

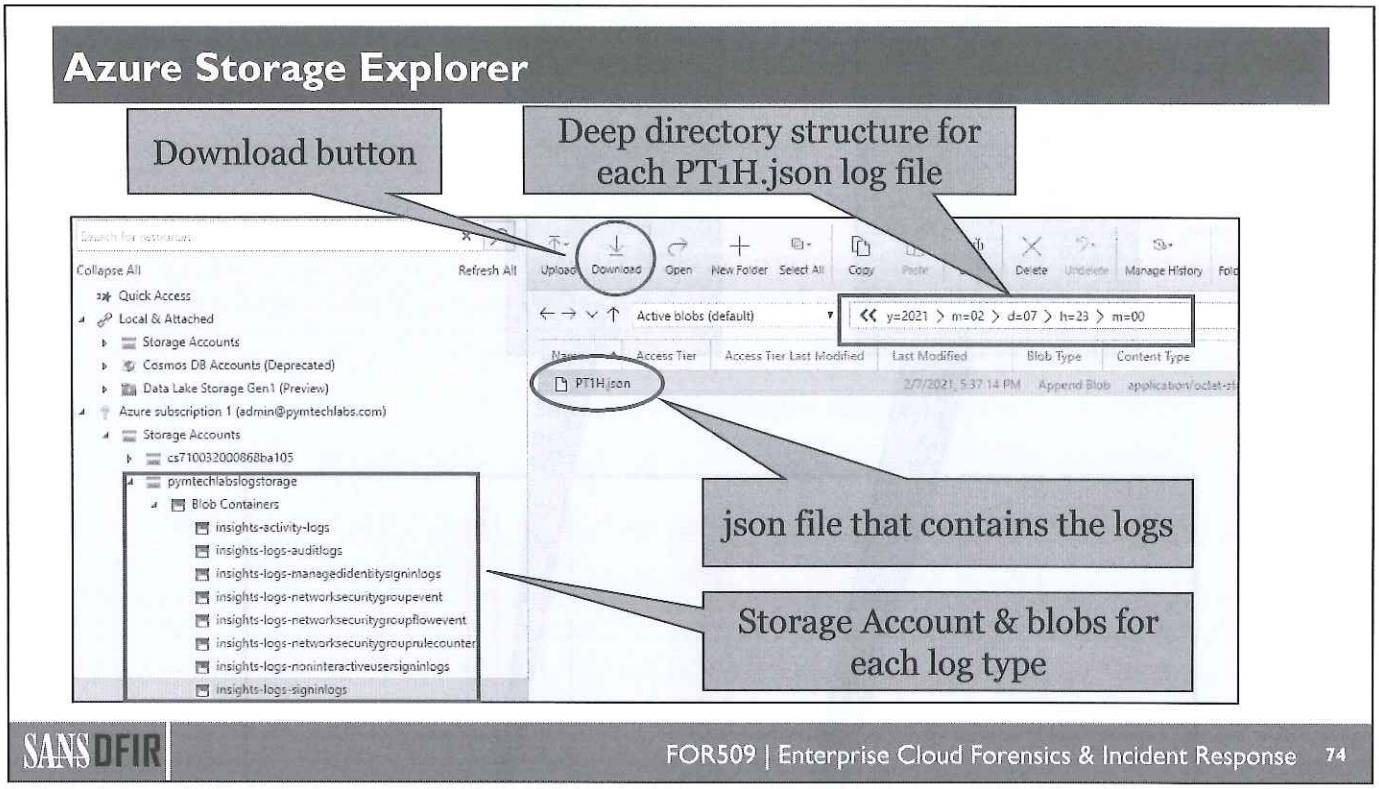
One of the great features of storage accounts it the ability to specify a retention period. As expected, you will be billed based on the quantity of data stored in the storage account. You will need to balance which logs you wish to retain, how long to retain them for, and the cost of the storage. Each organization will have a different answer.

Once the logs are in the storage account, you may access them in different ways. Programmers will use their favorite programming language to access the data via API. Another easier way is to use a tool such as Azure Storage Explorer which provides a GUI to see the storage accounts and the blobs that store the logs.



Select how long to keep the logs

Select a storage account



Azure Storage Explorer is the simplest way to access the blobs within the storage account and download the data. Azure Storage Explorer is a free Microsoft application that you will need to install on your computer.[1]

We will use Azure Storage Explorer to export many kinds of logs. As a preview, here are some of the names of some of the logs you may encounter. There are so many, we are only listing a few here:

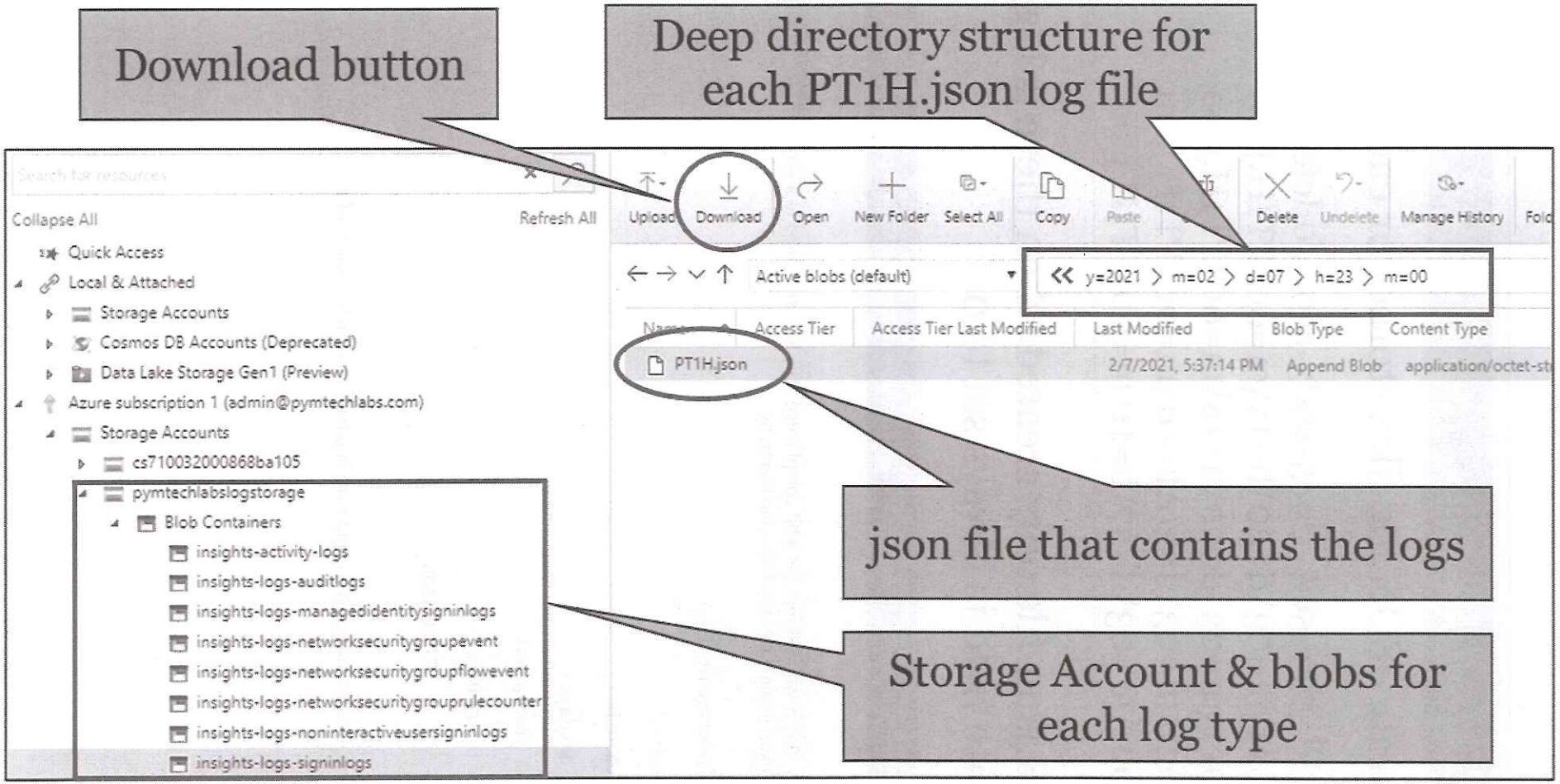
Log	Name
Tenant	insights-logs-auditlogs
	insights-logs-managedidentitysigninlogs
	insights-logs-noninteractiveusersigninlogs
	insights-logs-signinlogs
Subscription	insights-activity-logs
Network Watcher	insights-logs-networksecuritygroupevent
	insights-logs-networksecuritygrouprulecounter
NSG Flow	insights-logs-networksecuritygroupflowevent

The schema for these logs is documented in reference.[2]

Most logs will be stored in the storage account as blobs. Operating system logs will be stored as tables.

References:

[1] https://for509.com/storageexplorer
[2] https://for509.com/schema-activitylog



Tenant Logs – Storage Blobs

- Separate JSON log files stored for each hour

```
tenantId=7e325eda-7945-46d3-ac99-f0dcfeb4628e\
  y=2021\m=03\d=20\h=15\m=00\PT1H.json
  y=2021\m=03\d=20\h=18\m=00\PT1H.json
  y=2021\m=03\d=26\h=01\m=00\PT1H.json
  y=2021\m=03\d=27\h=11\m=00\PT1H.json
```

- PT1H.json file only created when there is log data
- Minute field is always set to 00

Unfortunately, this is where things get a bit complicated. The logs are stored in a series of files called PT1H.json broken down under the following hierarchy:

```
tenantId=<tenant id>\
  y=<year>\
    m=<month>\
      d=<day>\
        h=<hour>\
          m=00\
              PT1H.json
```

As you can imagine, this means you could have hundreds or even thousands of files.

Tenant Logs – Import into SOF-ELK

Steps to import logs in SOF-ELK

1. Download the blob containers
   a) Azure Storage Explorer then transfer to SOF-ELK VM
   b) Python script directly on SOF-ELK VM (see notes)
2. Combine all PT1H.json files into a single file

```
[elk_user@sof-elk]> find ./insights-logs-auditlogs/ -type f -name PT1H.json ⏎
                    -exec cat {} + |tee insights-logs-auditlogs.json
```

3. Copy file to Azure Logstash folder

```
[elk_user@sof-elk]> cp insights-logs-auditlogs.json /logstash/azure
```

4. Repeat for the different sign-in logs

Import these logs into SOF-ELK by following these steps:

1. Download the blob containers with option a) or b)
   a) Use Azure Storage Explorer and then transfer the logs to the SOF-ELK VM
   b) Use the download_blobs.py script[1] directly in your SOF-ELK VM
2. In your SOF-ELK VM, you will now have a large number of PT1H.json files located in a deeply nested directory structure. You will need to combine them into a single. Assuming that the top directory is called `insights-logs-auditlogs`, run the command:

```
find ./insights-logs-auditlogs/ -type f -name PT1H.json ⏎
          -exec cat {} + |tee insights-logs-auditlogs.json
```

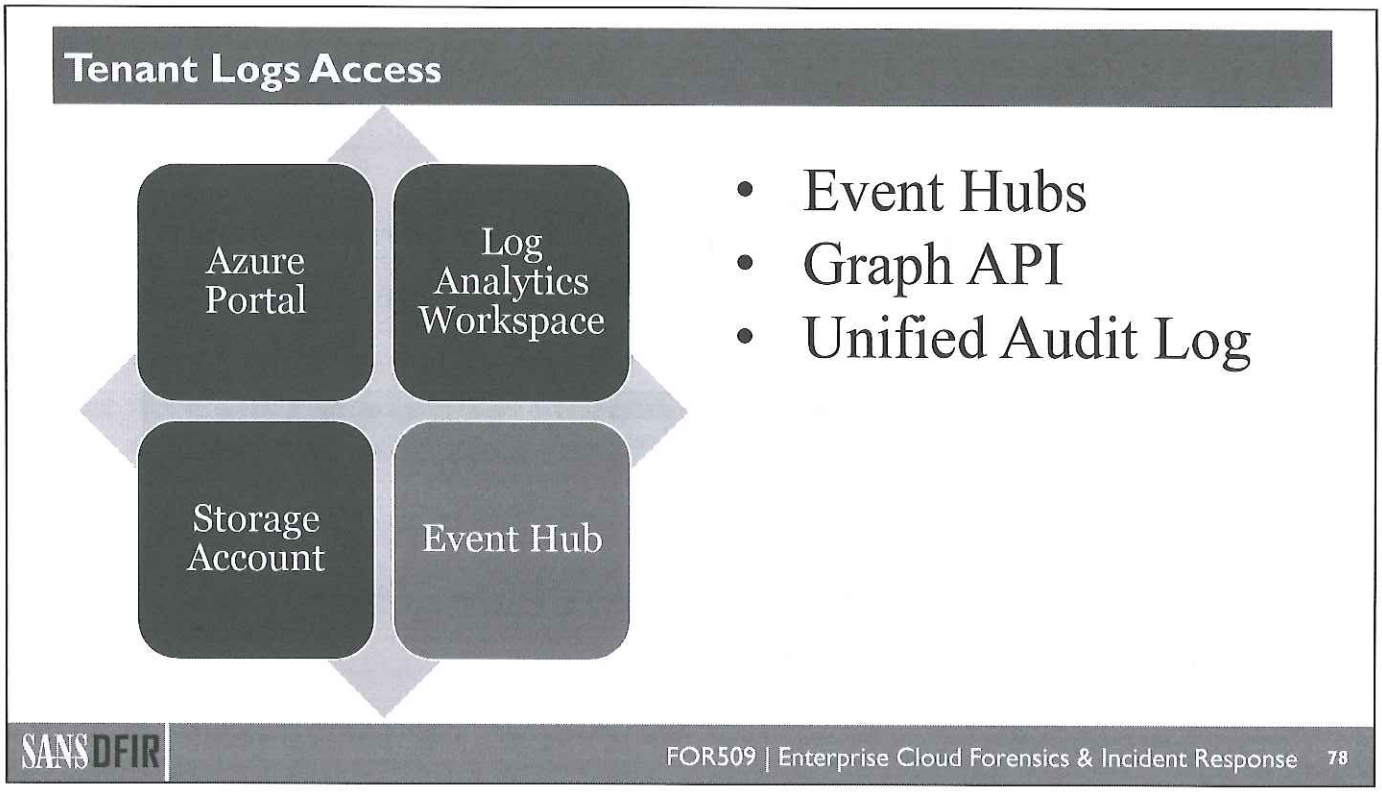
3. You will now have a single file called `insights-logs-auditlogs.json` that combines every PT1H.json files. Copy it to the appropriate Logstash folder:

```
cp insights-logs-auditlogs.json /logstash/azure
```

4. ELK will now process the JSON file and within a few minutes you will be able to see the logs in Kibana.

References:

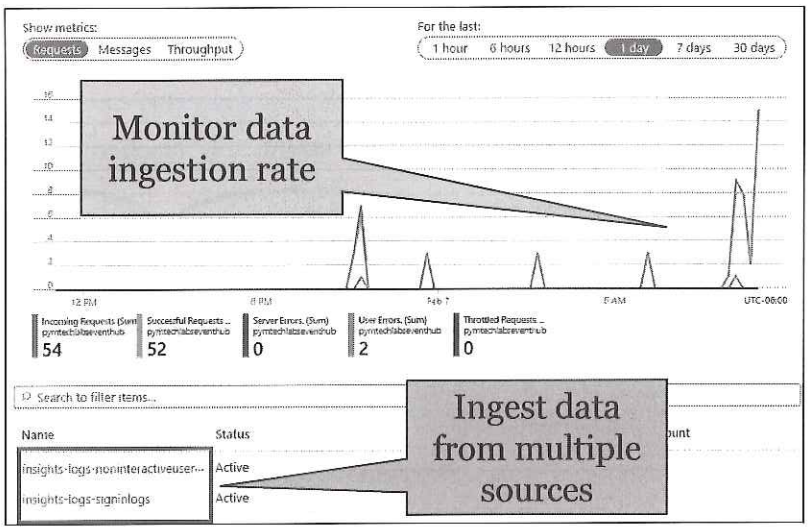
[1] The python script is courtesy of Quick Programming Tips
https://for509.com/blobpythonscript



This page intentionally left blank.

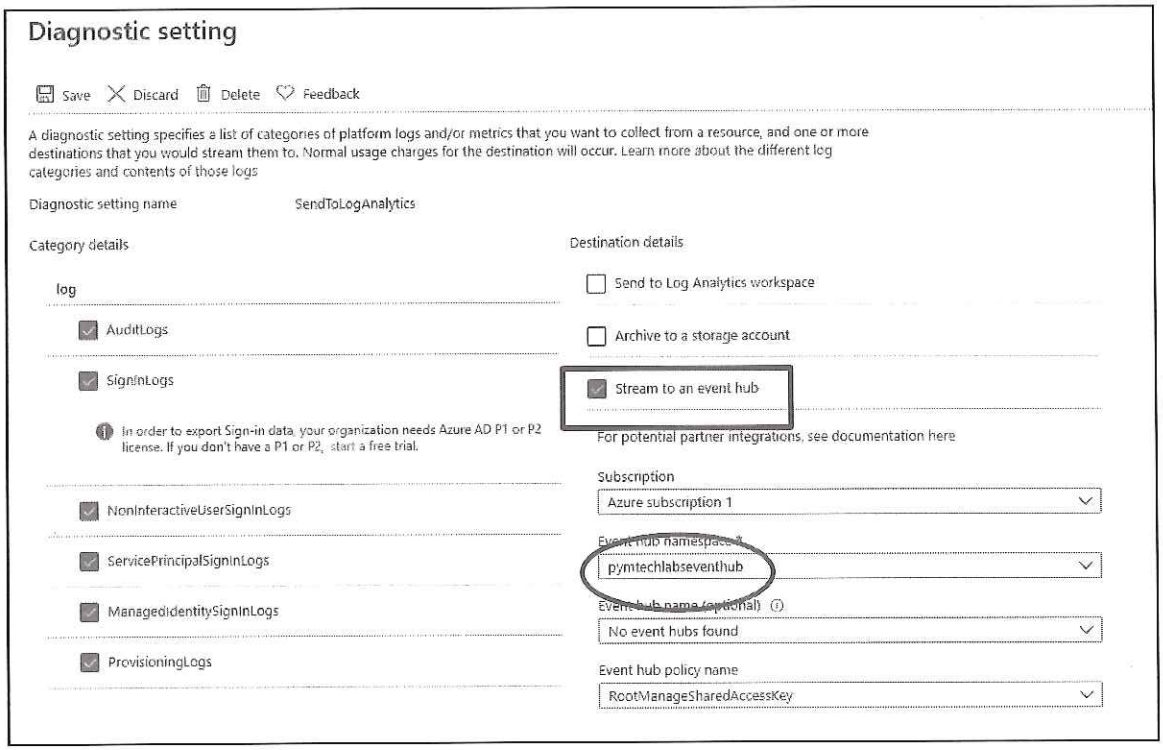
Event Hubs Stream

- Event hubs are real-time data ingestion services
- Capable of ingesting millions of events per second
- Support for multiple sources of data
- Check for their existence

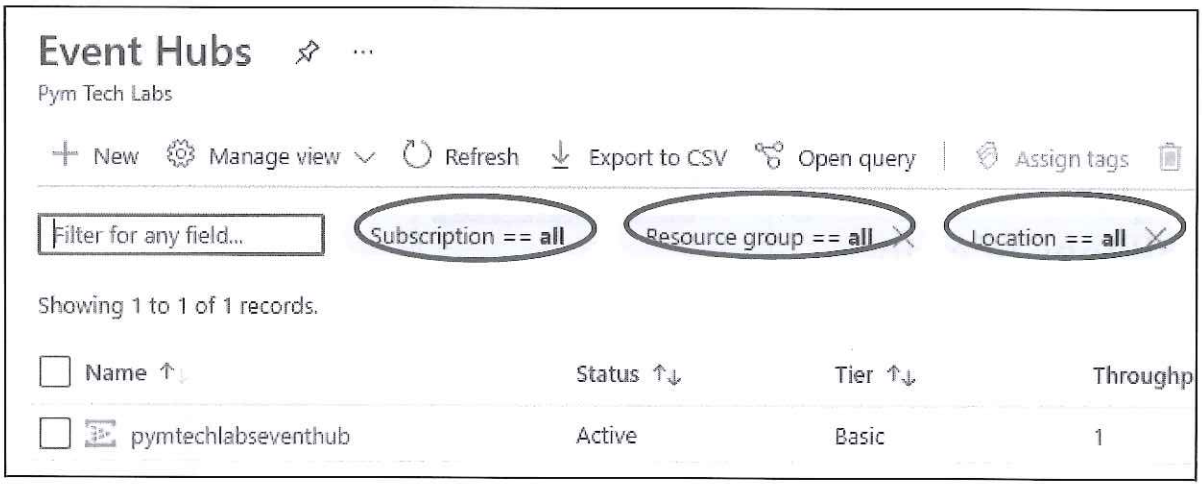


If you want to send your logs to a non-Azure destination, the event hub is a real-time data ingestion service.[1] It's capable of ingesting millions of events per second and create a dynamic data pipeline to feed an application such as a SIEM.

The configuration to send logs to an event hub follows the same process as the one for the storage account. First, you will need to create the event hub. Second, you will specify the event hub namespace in the diagnostic settings as shown below.[2]



Using an event hub is beyond the scope of this course. However, as part of your investigation, it's important to look for such a configuration as it may point to the existence of a log repository that your client may have forgotten to tell you about.



References:

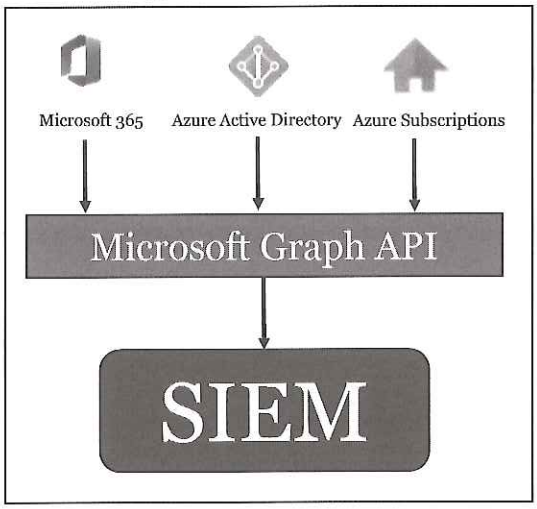
[1] https://for509.com/eventhubs-overview

[2] https://for509.com/eventhubs

Graph API

- An alternative method to send logs to an external SIEM is to use the Graph API
- Highly customizable and granular
- Important to discuss with SIEM team which logs they choose to import
- SIEM team should provide name of indexes for each type of logs

Sample architecture



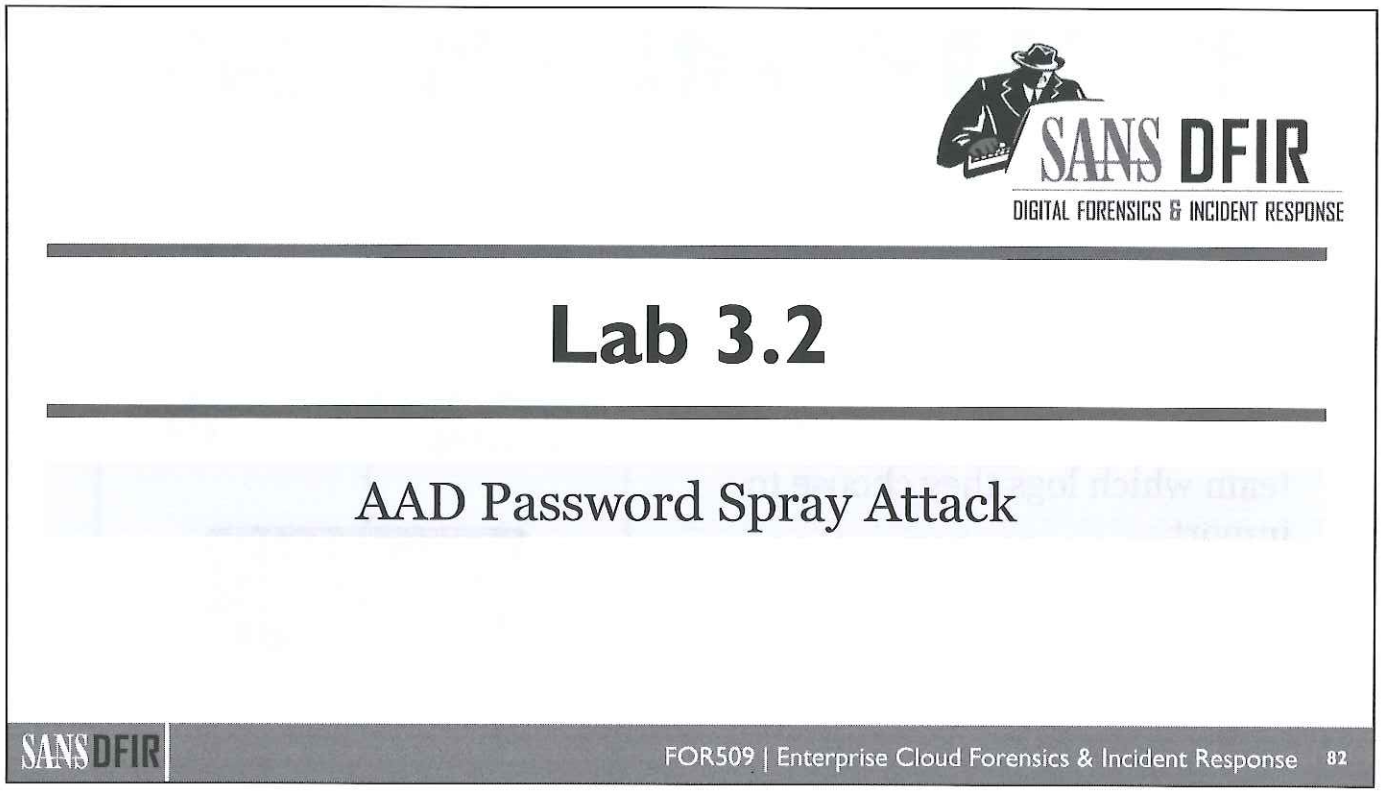
The Microsoft Graph API is a very powerful way to retrieve data from Azure and Microsoft 365. Many programming languages and platform are supported.[1]

Like the event hubs, it's important that you know about this feature so you can communicate with the SIEM team and understand what logs they choose to import.

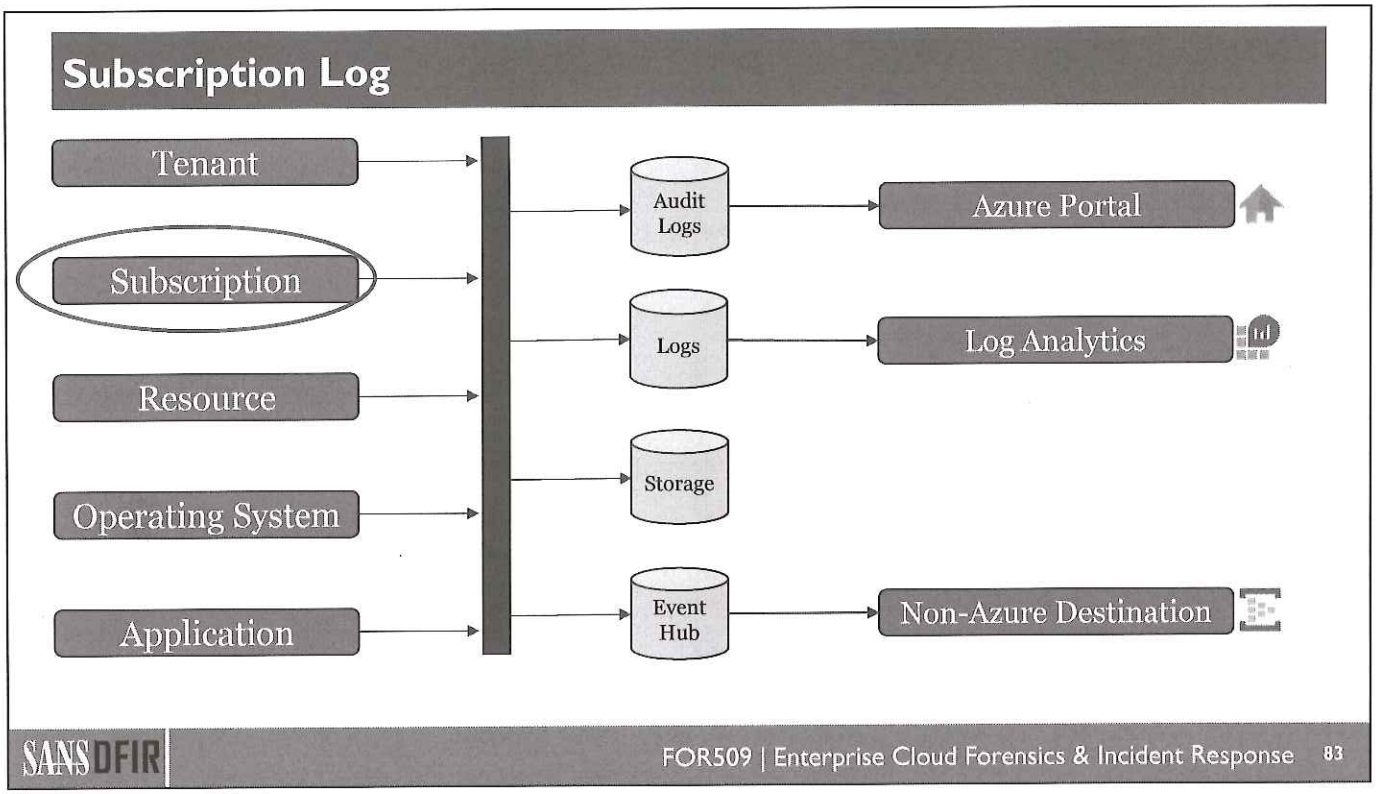
A company may choose to use event hubs and/or the Graph API as many different architectures are possible. Because the Graph API pre-dates event hubs, you are very likely to find that many SIEMs obtain their data from Azure in this manner.

References:

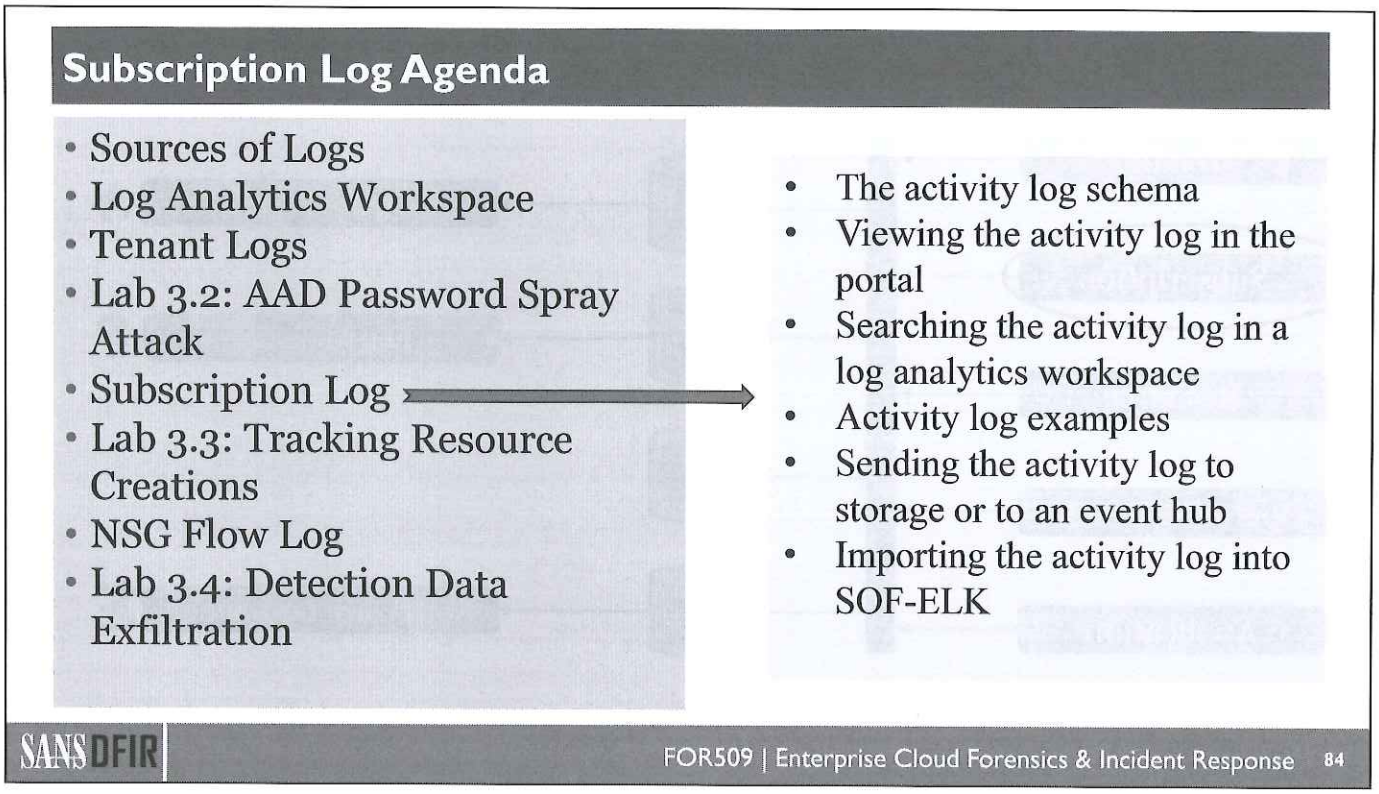
[1] https://for509.com/graph



This page intentionally left blank.



This page intentionally left blank.



The subscription log contains information about operations conducted by tenant-wide services.[1] This is where you will find information about resources being created, modified, or deleted.

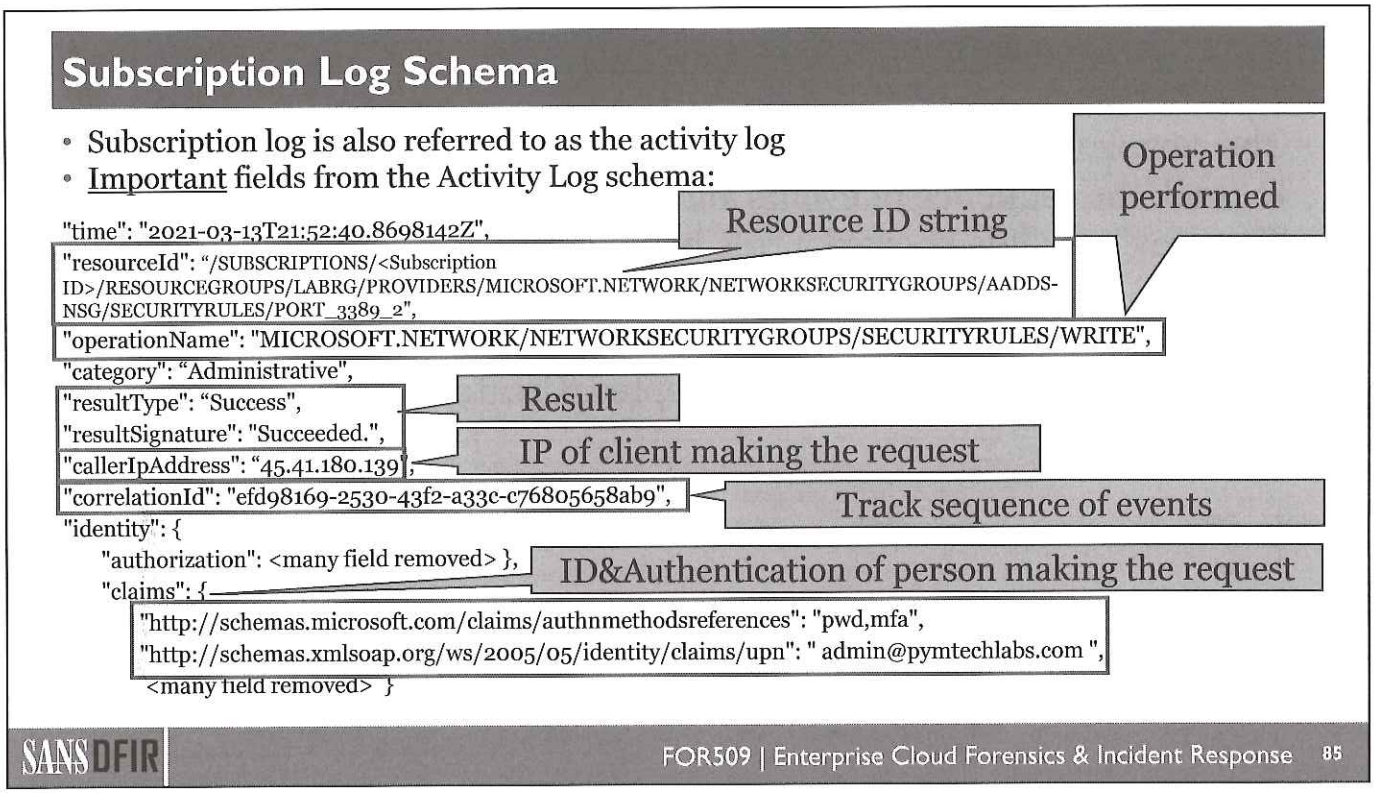
You will find the subscription logs under the "Activity log" service.

Just like the tenant logs, there are four actions you can take with these logs:

1. View them directly on the Azure portal.
2. Store them in a log analytics workspace.
3. Send them to a storage account for archival.
4. Send them to a SIEM by using the event hub.

References:

[1] https://for509.com/activitylog



The subscription log is often referred to as the activity log. The complete schema is quite long, and many fields duplicate the same information. In this slide, we are highlighting a few important fields.

- **resourceId** is the string of the resource being added/modified/deleted
- **operationName** is the name of the provider making the change and the nature of the change. In this example, we are changing a security rule in the Network Security Group
- **resultType** & **resultSignature** contain nearly identical information with the result of the operation
- **callerIpAddress** is the IP address of the client making the request
- **correlationId** is a unique GUID that is used to track the sequence of events that make the operation (example in the next slide)
- **Identity** contains an authorization sub-field and a claims sub-field. In the claims sub-field, you will find information about the person who is making the change

Notice how the schema has multiple levels of nested fields. This will be a challenge to import in SOF-ELK as SOF-ELK uses a flat schema. We will discuss later some of the compromises that have to be made.

Subscription Log - correlationId

- The correlationId field is found in nearly all Azure logs. It's very useful to track the sequence of events and make up an operation
- The previous slide showed a rule being added to a network security group
- Tracking the correlationId shows the three steps that Azure performed to complete that task: Started, Accepted/Created, Succeeded

correlation_guid : "efd98169-2530-43f2-a33c-c76805658ab9"

	Time ▲	action	result_signature
>	2021-03-13 21:52:40.869 +00:00	Microsoft.Network/networkSecurityGroups/securityRules/write	Started.
>	2021-03-13 21:52:41.239 +00:00	Microsoft.Network/networkSecurityGroups/securityRules/write	Accepted.Created
>	2021-03-13 21:52:51.414 +00:00	Microsoft.Network/networkSecurityGroups/securityRules/write	Succeeded.

Using SOF-ELK we can filter our data to only show the events with the same correlationId:

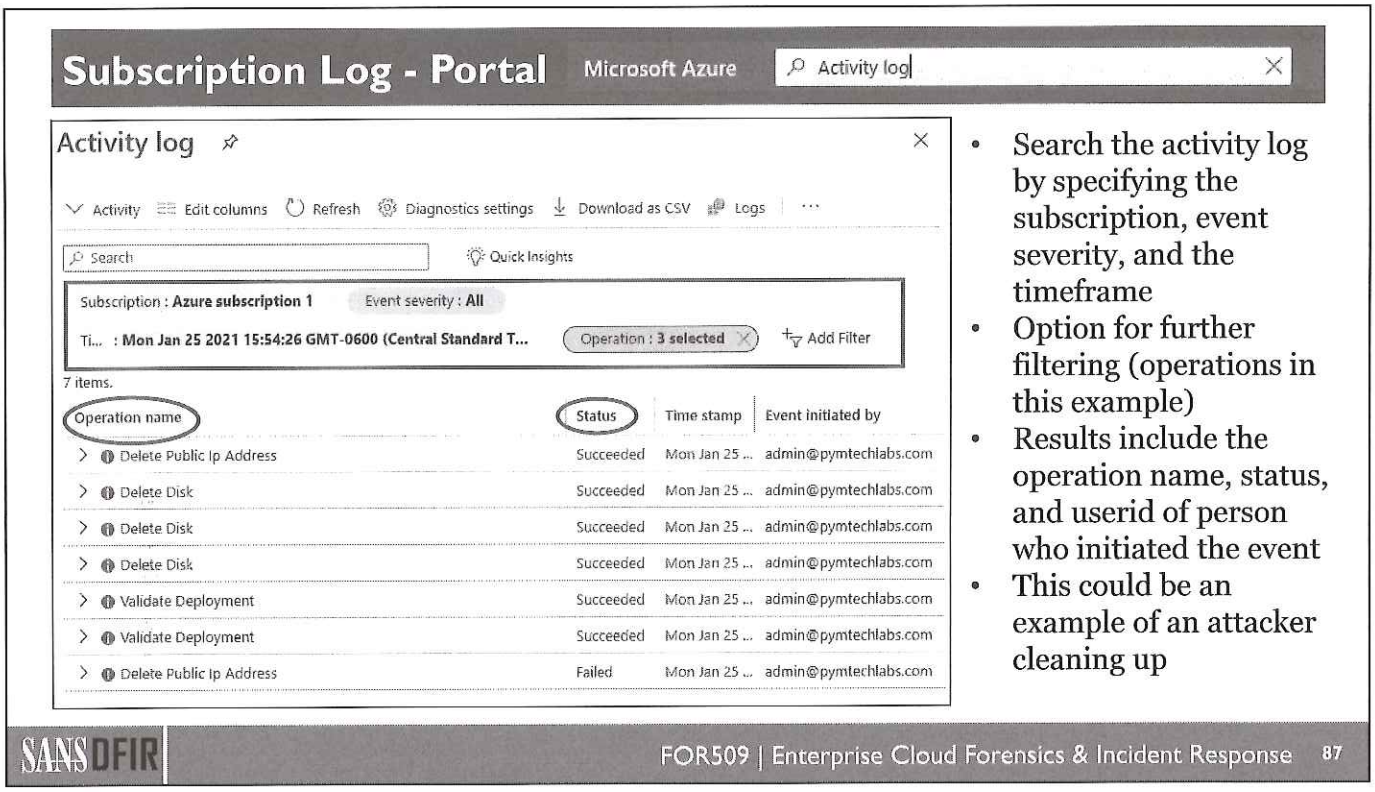
```
Correlation_guid : "efd98169-2530-43f2-a33c-c76805658ab9"
```

This filter will show us the three operations that Azure performed to complete the tasks. We present the information in table format by only showing time, action, result_signature for clarity. SOF-ELK contains more details for each event.

One of these details is the resource_id string:

```
Resource_id: /SUBSCRIPTIONS/BEB0C5FA-418F-4240-AA63-
FF8CAD3F1E1B/RESOURCEGROUPS/LABRG/PROVIDERS/MICROSOFT.NETWORK/NETWORKSECU
RITYGROUPS/AADDS-NSG/SECURITYRULES/PORT_3389_2
```

From this string, you can see that we created a rule called "PORT_3389_2". While the log won't show the details of the rule, it certainly points to someone possibly allowing RDP through the NSG group called "AADDS-NSG".



Just like the AAD log, the portal is a quick and easy way to get an overview. However, as you will see in the next few slides, it's not a practical way to get detailed information.

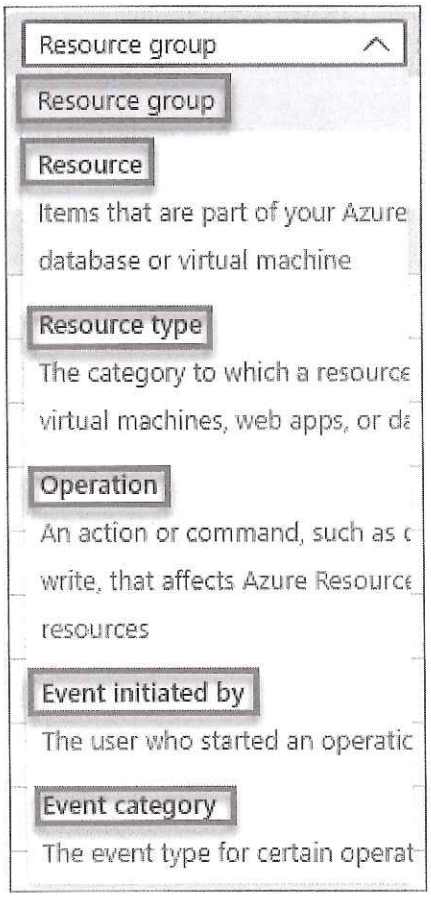
In the portal, you need to select:

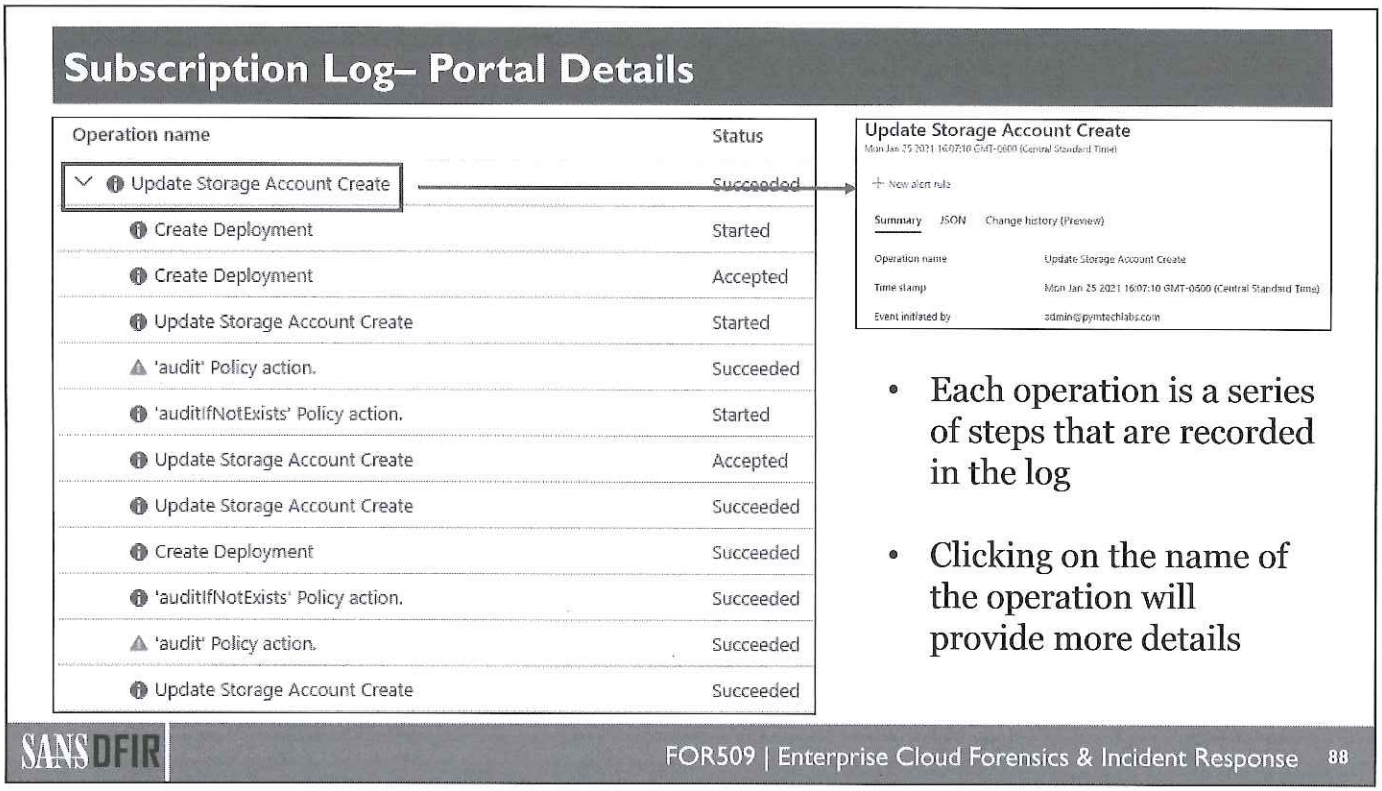
1. The subscription you wish to query
2. The event severity: Critical, Error, Warning, Informational
3. The timeframe
4. Optionally, additional filters are available as shown in the picture on the right.

In the example above, you can see resources being deleted. Specifically, a storage disk and an IP address. The last event shows an IP address deletion that failed because the VM was still running.

Looking for a large number of deletion events, either successful or failed, can be very valuable for your investigation as the attacker may be cleaning up after themselves.

Additional Filters:





In this example, we are creating a new storage account. Notice how this single action is made up of numerous smaller steps. To see the smaller steps, click on the arrow itself rather than the text.

If you click on the text of the operation, you will get details about who initiated it and when.

Further, by selecting the JSON tab, you will see a large number of details regarding the operation. Unfortunately, there is no ability to export the JSON from the portal other than copy/paste which is not practical for a large number of events.

The same data can be queried programmatically using the `Get-AzLog` PowerShell cmdlet.[1] Alternatively, you can use the CLI with the command `az monitor activity-log`.[2]

However, as we saw with the AAD logs, using the log analytics workspace is a much better way to visualize the information in the portal.

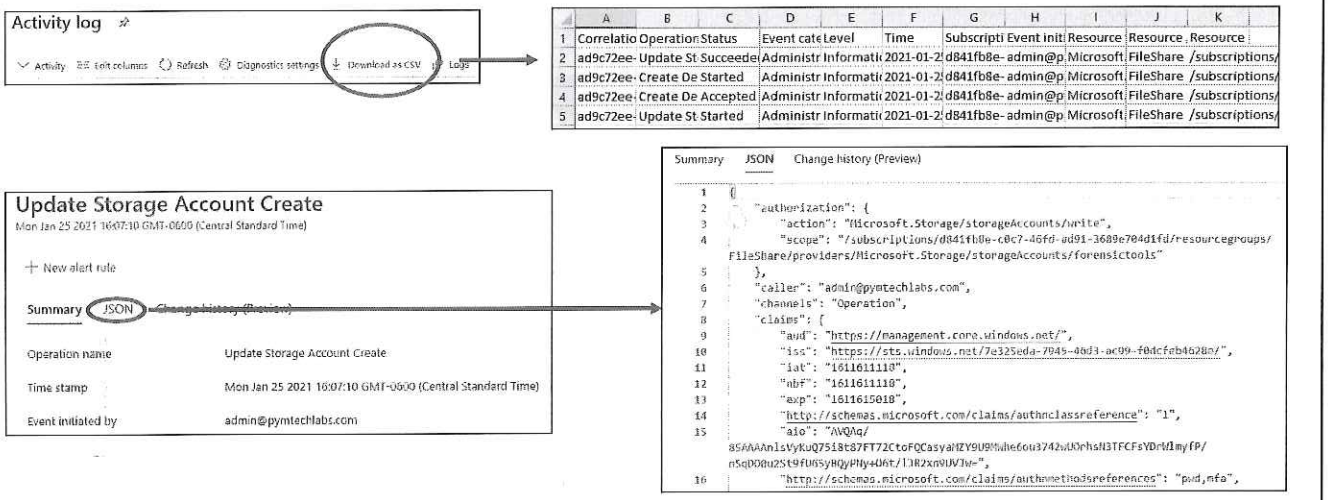
References:

[1] https://for509.com/monitor-powershell

[2] https://for509.com/monitor-cli

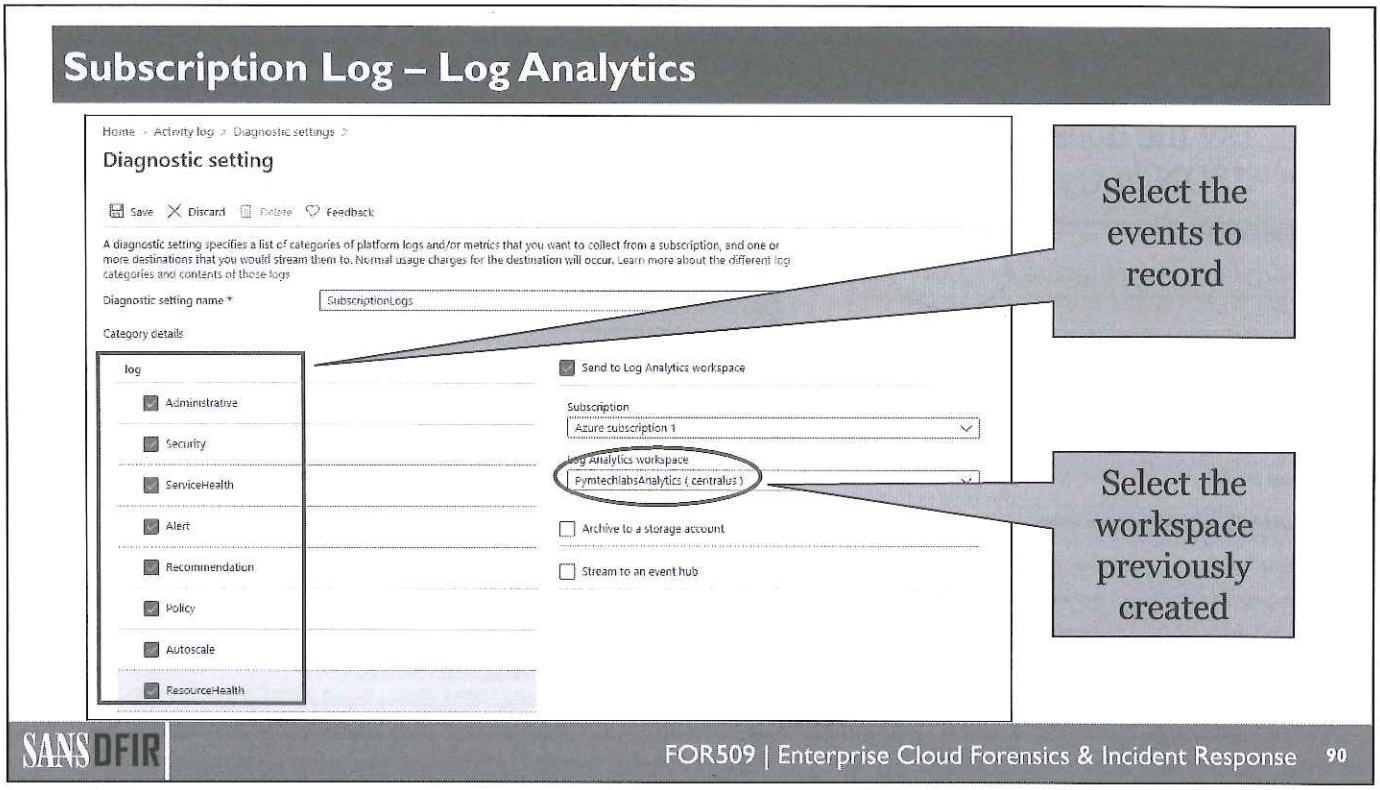
Subscription Log – Log Entry Details

- csv file doesn't include all the details that are part of the schema
- JSON entry has all the details but not useful inside the portal



	A	B	C	D	E	F	G	H	I	J	K
1	Correlatio	Operation	Status	Event cate	Level	Time	Subscripti	Event init	Resource	Resource	Resource
2	ad9c72ee-	Update St	Succeede	Administr	Informati	2021-01-2	d841fb8e-	admin@p	Microsoft	FileShare	/subscriptions/
3	ad9c72ee-	Create De	Started	Administr	Informati	2021-01-2	d841fb8e-	admin@p	Microsoft	FileShare	/subscriptions/
4	ad9c72ee-	Create De	Accepted	Administr	Informati	2021-01-2	d841fb8e-	admin@p	Microsoft	FileShare	/subscriptions/
5	ad9c72ee-	Update St	Started	Administr	Informati	2021-01-2	d841fb8e-	admin@p	Microsoft	FileShare	/subscriptions/

You may have noticed that the portal provides an option to export the data into a CSV file. While this appears to be a good solution, upon closer examination you will find that only the high-level information is exported. When you compare the data in the CSV file to the data in the JSON, you will see that all the detailed information is missing. If you need to export the data to a file, we will show you how to write it to a storage account just like we did for the AAD logs.



While the portal may be fine for a quick look at the subscription logs, it's much better to setup a log analytics workspace. By sending the subscription logs to the same log analytics workspace as the AAD log, we can get a more comprehensive view of the events in our Azure tenant.

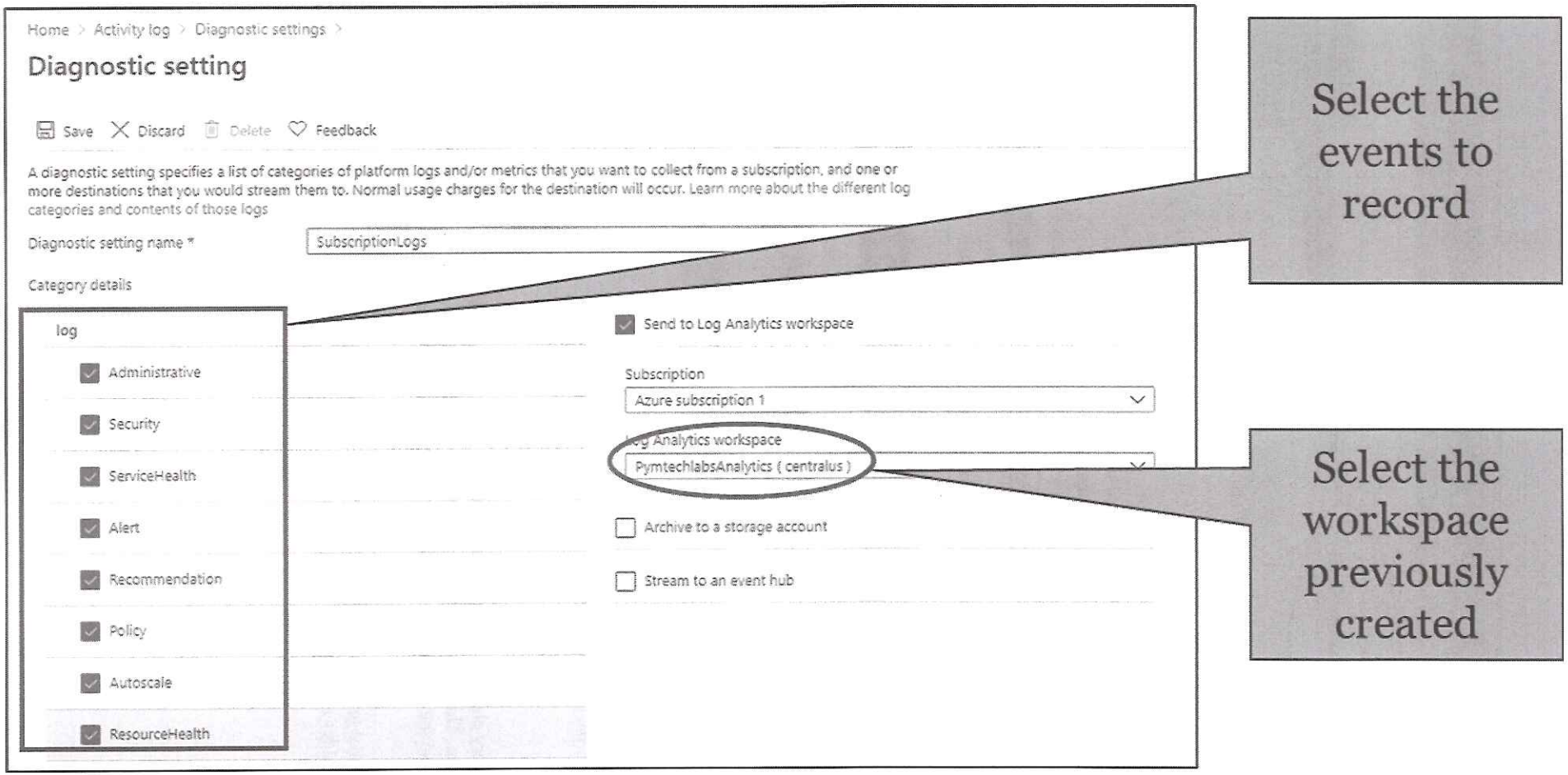
The setup process is identical to the one for the tenant logs: select the log categories you wish to save and the log analytics workspace to send them to.

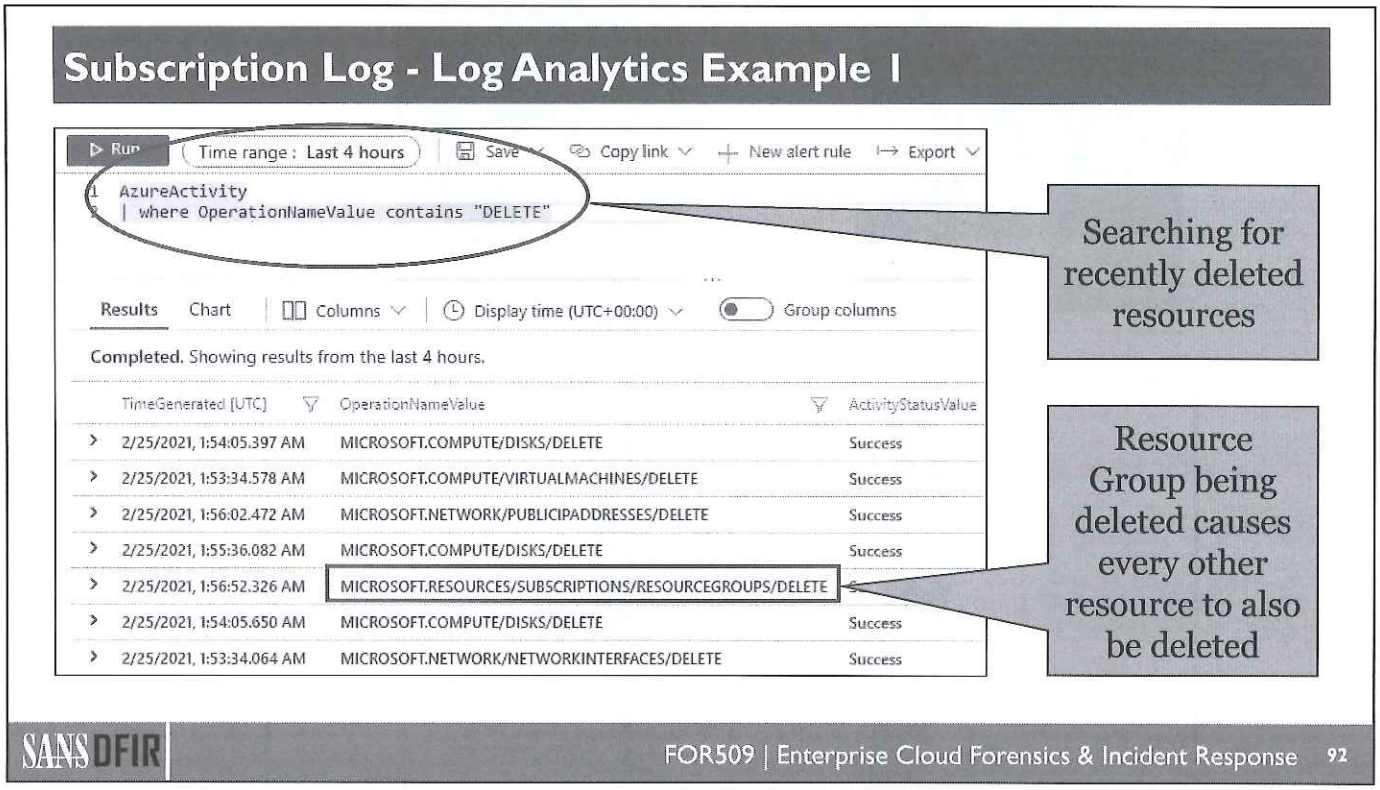
Subscription log categories are:

- **Administrative**: Actions performed through the Resource Manager such as resource creation, update, and deletion.
- **Security**: Alerts generated by Azure Security Center.
- **Service Health**: Service health incidents such as Azure service downtime.
- **Alert**: Triggering of previously configured alerts, such as CPU utilization exceeding a specific threshold.
- **Recommendation**: Recommendations from the Azure Advisor.
- **Policy**: Policy events such as *audit* and *deny* per policies established in the subscription.
- **Autoscale**: Events related to the operation of the autoscale engine based on settings defined in your subscription.
- **Resource Health**: Events regarding the health of your resource with possible status of *Available*, *Unavailable*, *Degraded*, and *Unknown*.

Many of these categories will only show up if the corresponding settings have been enabled in your subscription. From a practical point of view, you will mostly see administrative log entries.

Just like the tenant logs, it might be tempting to save everything, but in a large organization the storage requirements may get very costly.





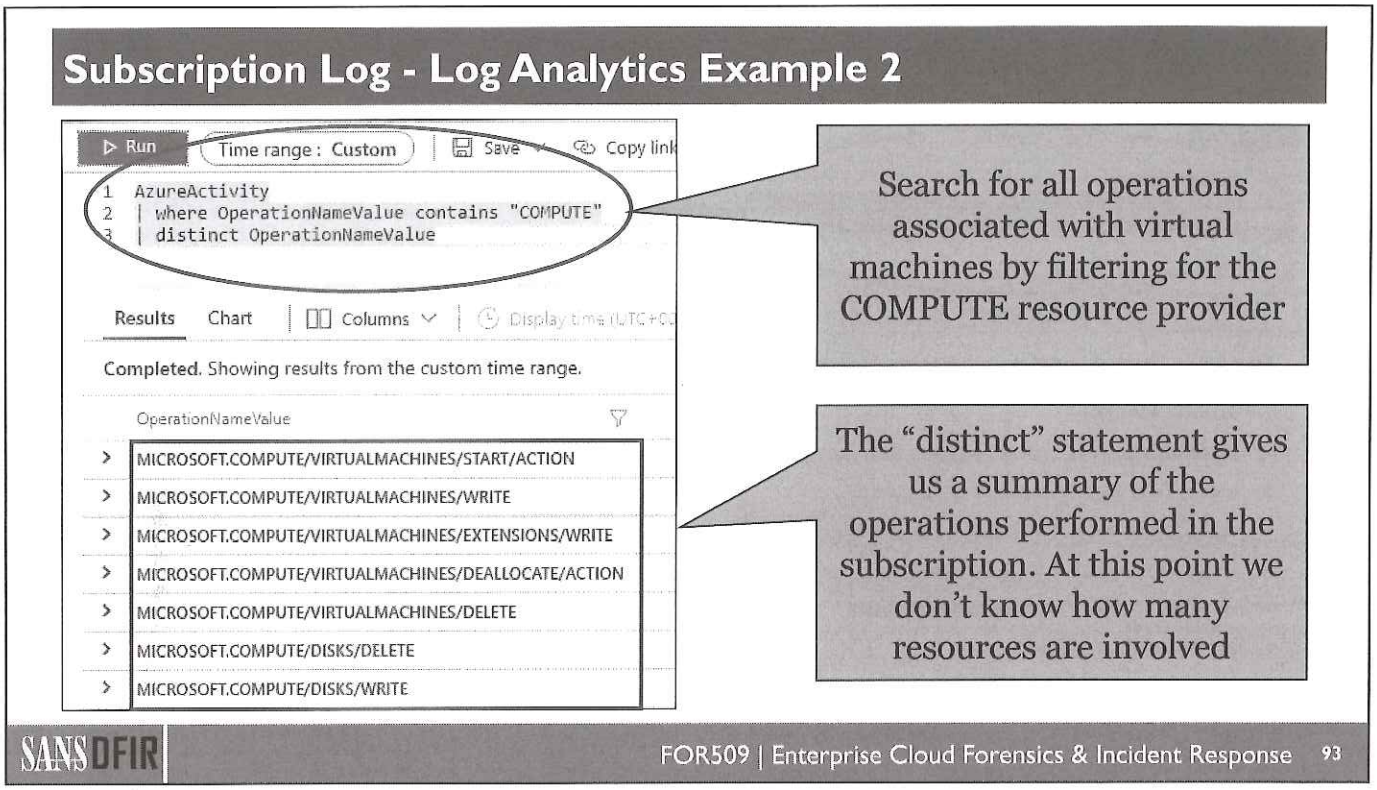
Now that we have configured our Log Analytics workspace, let's look at an example query.

The table that contains the subscription logs is called *AzureActivity*. Consider the scenario where a bad actor deletes a large number of resources to cover their activity. You could write the following query:

```
AzureActivity
| where OperationNameValue contains "DELETE"
```

Observe that every element that constitutes a virtual machine is being deleted: disk, VM itself, IP address, network interface. This actually doesn't happen if you delete just the VM. The reason everything is being deleted is because the resource group that contains the VM is being deleted.

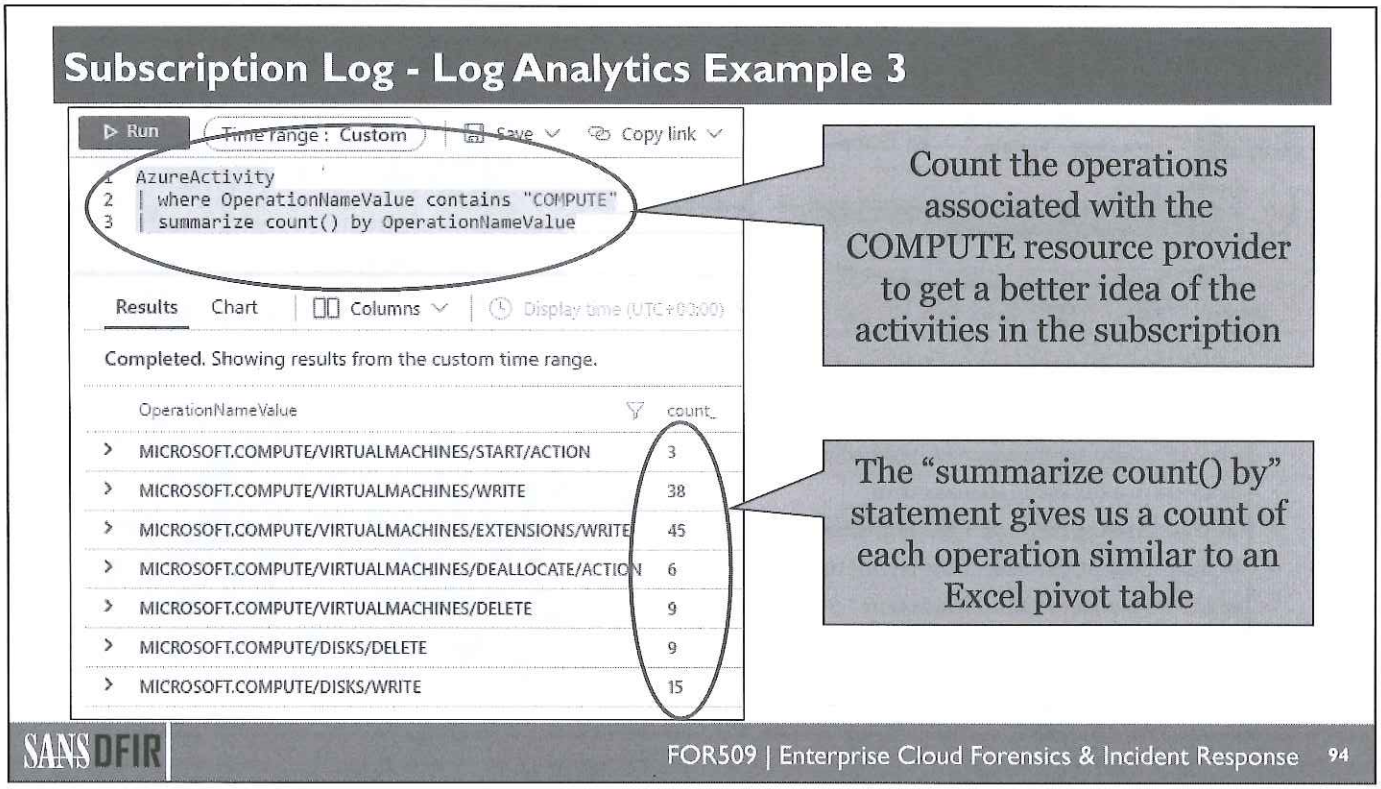
Deleting a resource group is a very effective way to delete a large number of resources, hence it's important to carefully consider who has this level of access.



Another interesting query would be to look for operations associated with virtual machines. You may remember that the resource provider responsible for virtual machines is called COMPUTE. So, to find the unique operations performed on virtual machines, you could write the following query:

```
AzureActivity
| where OperationNameValue contains "COMPUTE"
| distinct OperationNameValue
```

The "distinct" statement gives us the unique values. This is a great way to limit the output to a reasonable number of lines and get an overall view of the activity in the subscription.

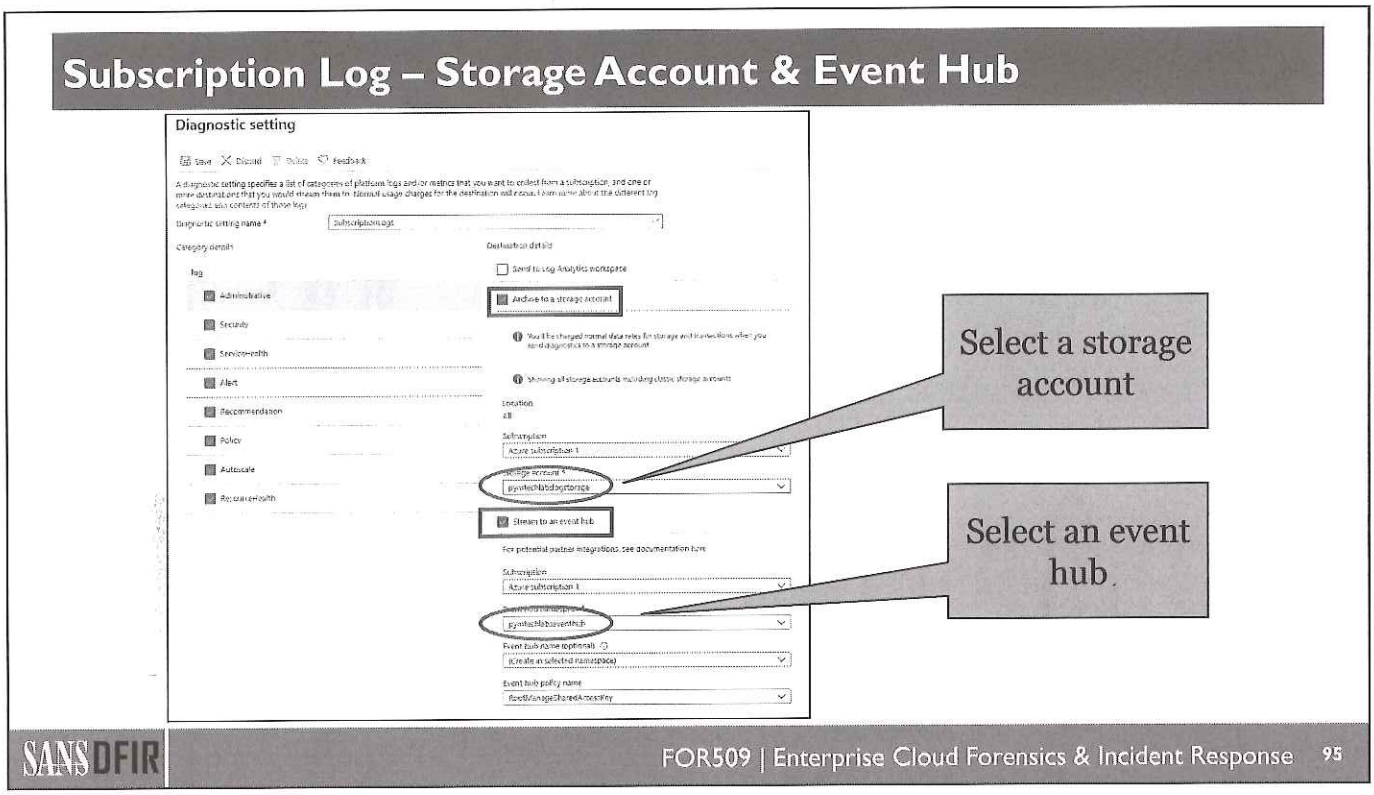


To understand the scope of the resources being impacted, we can count the number of operations like you would do in an Excel pivot table. The following query will do just that:

```
AzureActivity
| where OperationNameValue contains "COMPUTE"
| summarize count() by OperationNameValue
```

This example takes place over a number of days and shows the owner of the subscription creating, starting, stopping, and eventually deleting a number of virtual machines and associated disks. The output is truncated to fit in the slide.

As you can see the log analytics workspace combined with the KQL is a powerful tool to analyze your log data.



The diagnostic settings are configured the exact same way as tenant logs, specifically Azure Active Directory logs. All you have to do is specify the storage account and the event hub.

In this example, we will store the subscription logs to the storage account called pymtechlabslogstorage and stream the logs to the event hub called pymtechlabseventhub.

The log entries will be found in a blob called insights-activity-logs. As you may remember from one of our earlier slides, a series of files called PT1H.json will be created in a very deep directory structure. By using Azure Storage Explorer, you can download that directory structure and use the scripts provided in the SOF-ELK distribution to combine these files in a single one.

Subscription Log – Import into SOF-ELK

- The process is the same as the tenant logs
- Due to the flat nature of the schema in SOF-ELK, some of the fields have to be mapped to different names
- There are too many fields in the activity log schema, so we only mapped some of them. You may choose to create your own mapping

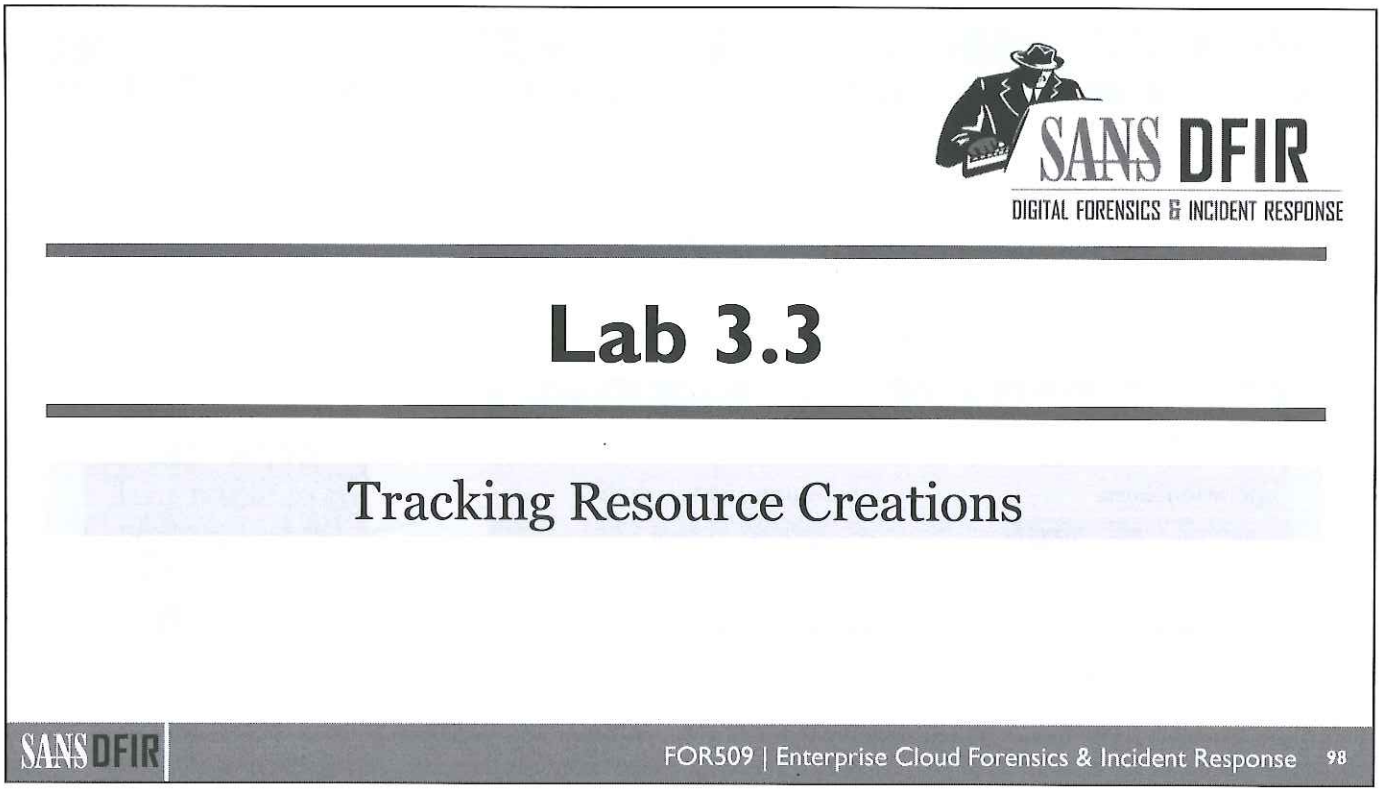
Activity Log Schema	SOF-ELK
resourceId	resource_id = scope
operationName	operation_name = action
resultType / resultSignature	result_type / result_signature
callerIpAddress	ips = source_ip
correlationId	correlation_guid
Identity / claims	Not mapped

Some fields are duplicated in the current version of the import script

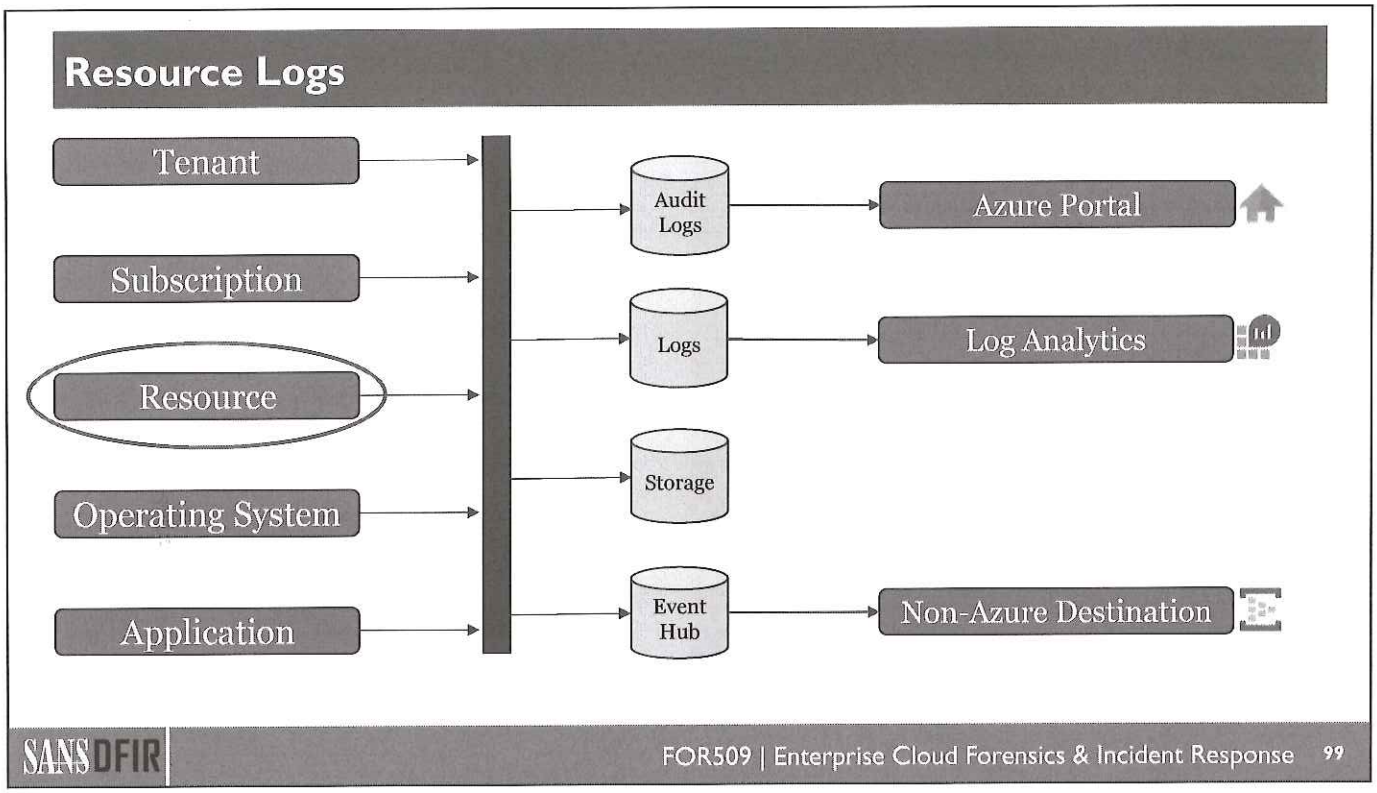
You can import the activity log the same way you imported the tenant logs. You should name your file `insights-activity-logs.json` so it's easily identifiable in SOF-ELK.

The nested nature of the activity log file is a challenge when mapping fields to import into SOF-ELK's flat structure. As a result, some of the data is imported into two separate fields.

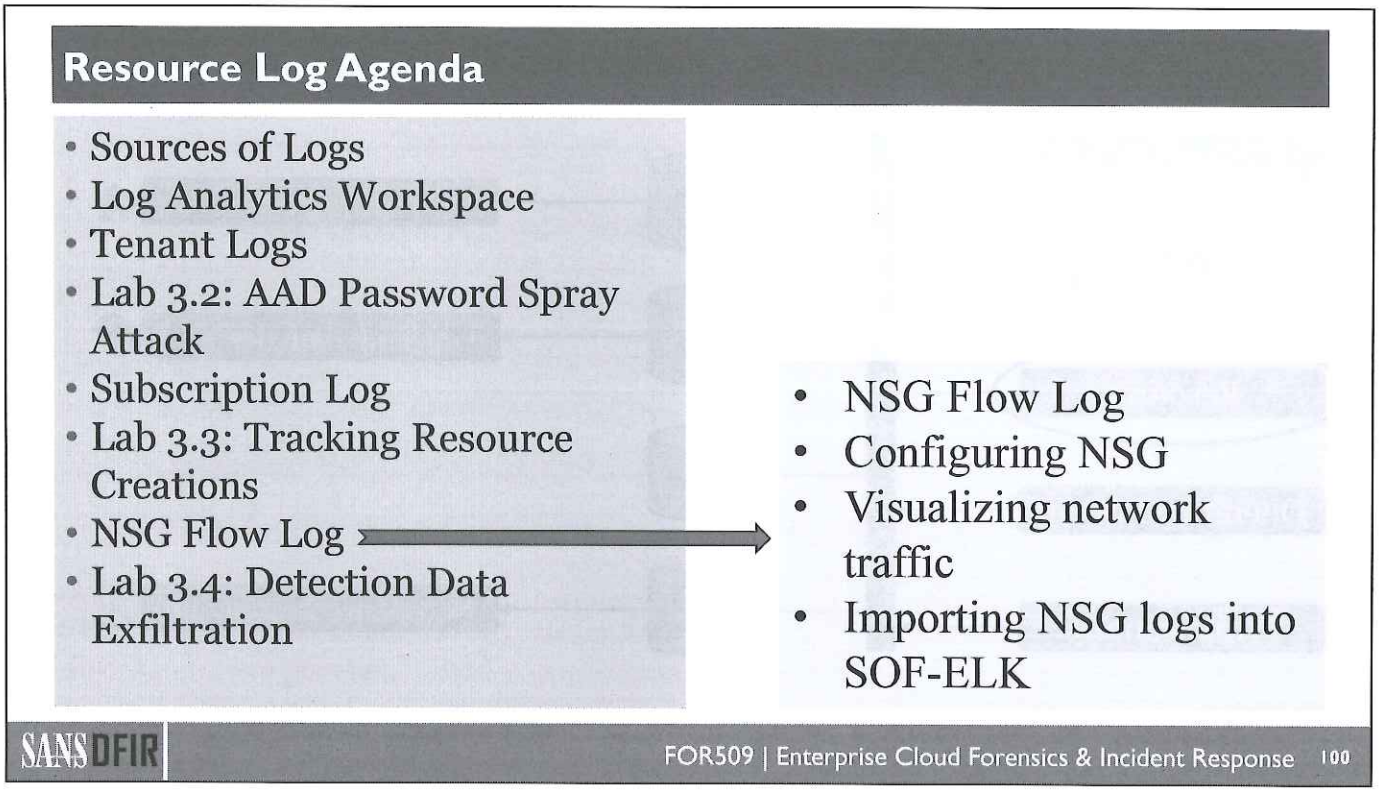
At this time, the claims ticket wasn't mapped. The Logstash parser is part of the public SOF-ELK distribution and everyone is welcome to contribute and improve the mapping.



This page intentionally left blank.



This page intentionally left blank.



Azure offers a large number of resources. Each one of these resources can generate one or more logs if configured to do so. This means that there are potentially hundreds of log categories.[1]

For the purposes of incident response and forensics, we will focus on the most important resource log: the network security group (NSG) flow log.

The portal, log analytics workspace, storage, and event hub configuration options are the same as the tenant and subscription logs but are not as useful since they only provide metadata about the creation of the flow logs.

Virtual machines are the other key Azure resource. However, we will cover them in the next section as their logs are more valuable when an agent is installed.

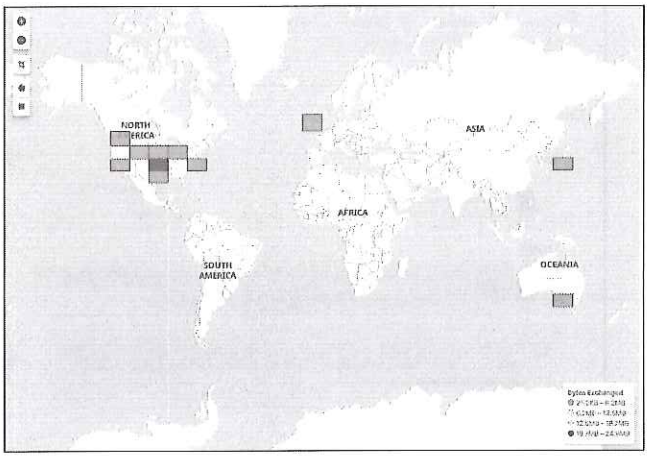
References:

[1] https://for509.com/resourcelogscategories

NSG Flow Log

- Network Security Group automatically created with every VM
- Enable log capture in Azure Network Watcher

- Import NSG logs in SIEM and create interesting visualizations



Deployment details (Download)

Resource	Type	Status
LuisVM	Microsoft.Compute/virtualMa...	OK
luisvm164	Microsoft.Network/networkIn...	Created
LuisVM-ip	Microsoft.Network/publicIpA...	OK
Maintenance-vnet	Microsoft.Network/virtualNet...	OK
LuisVM-nsg	Microsoft.Network/networkSe...	OK

As you may remember from section 2, Azure automatically creates a network security group (NSG) when you create a virtual machine: *<name of machine>-nsg*. This NSG allows you to control traffic in and out of the subnet. It's the ideal location to capture IP traffic flow log.

These logs are part of the Azure Network Watcher[1] and have the following characteristics:

- They are captured at the transport layer (layer 4)
- They are collected at one-minute intervals
- They are written in JSON format (like all other Azure logs)
- Each log record will include: the network interface, 5-tuple information, and traffic decision (Allow or Deny)
- They are retained for 1 year

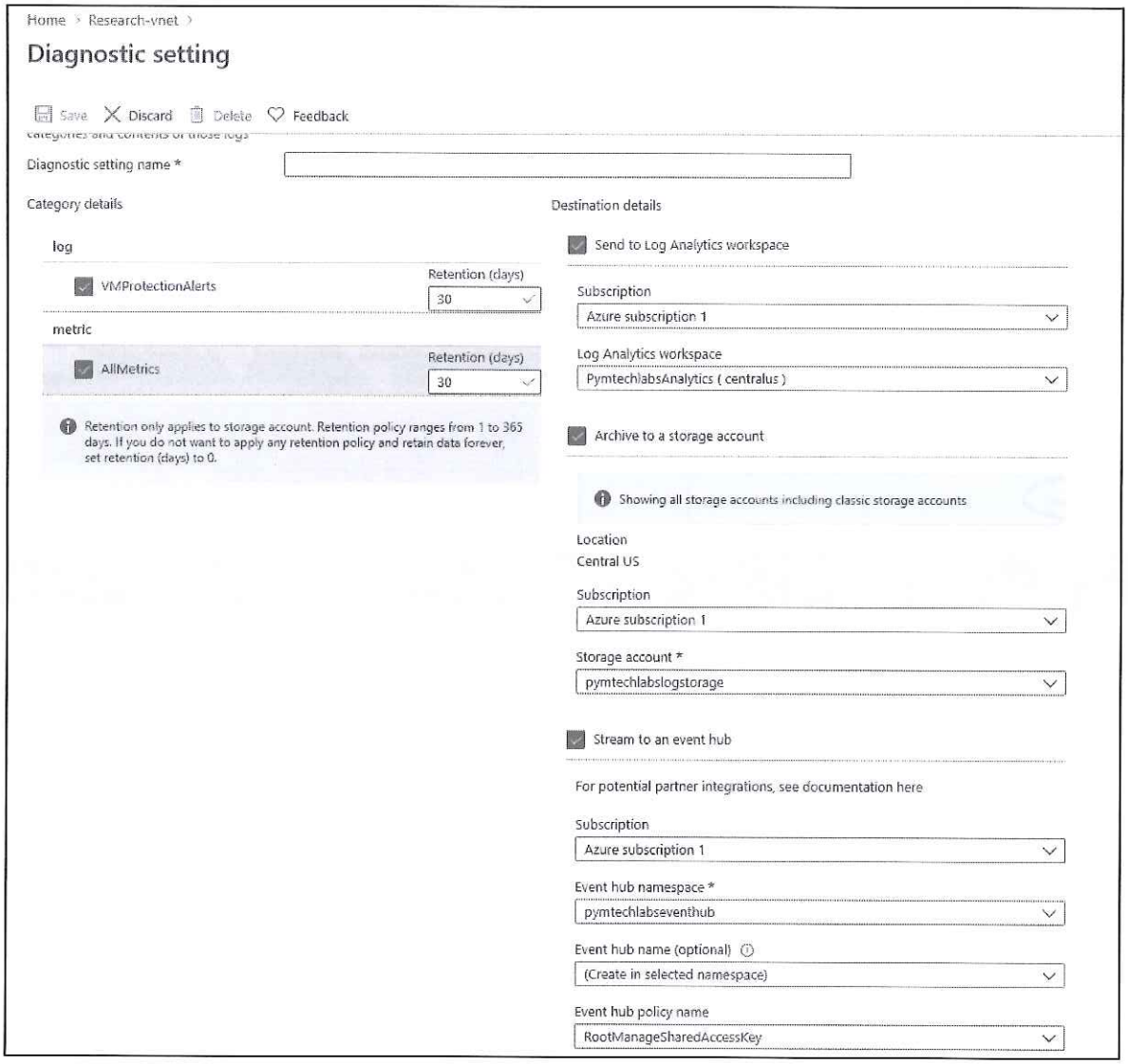
The example graph shows the traffic observed after a VM was created in Azure and left running for 1 hour doing "nothing". The port for remote desktop (RDP) was immediately scanned for potential vulnerabilities. Clearly, setting up strong rules in your NSG is a must.

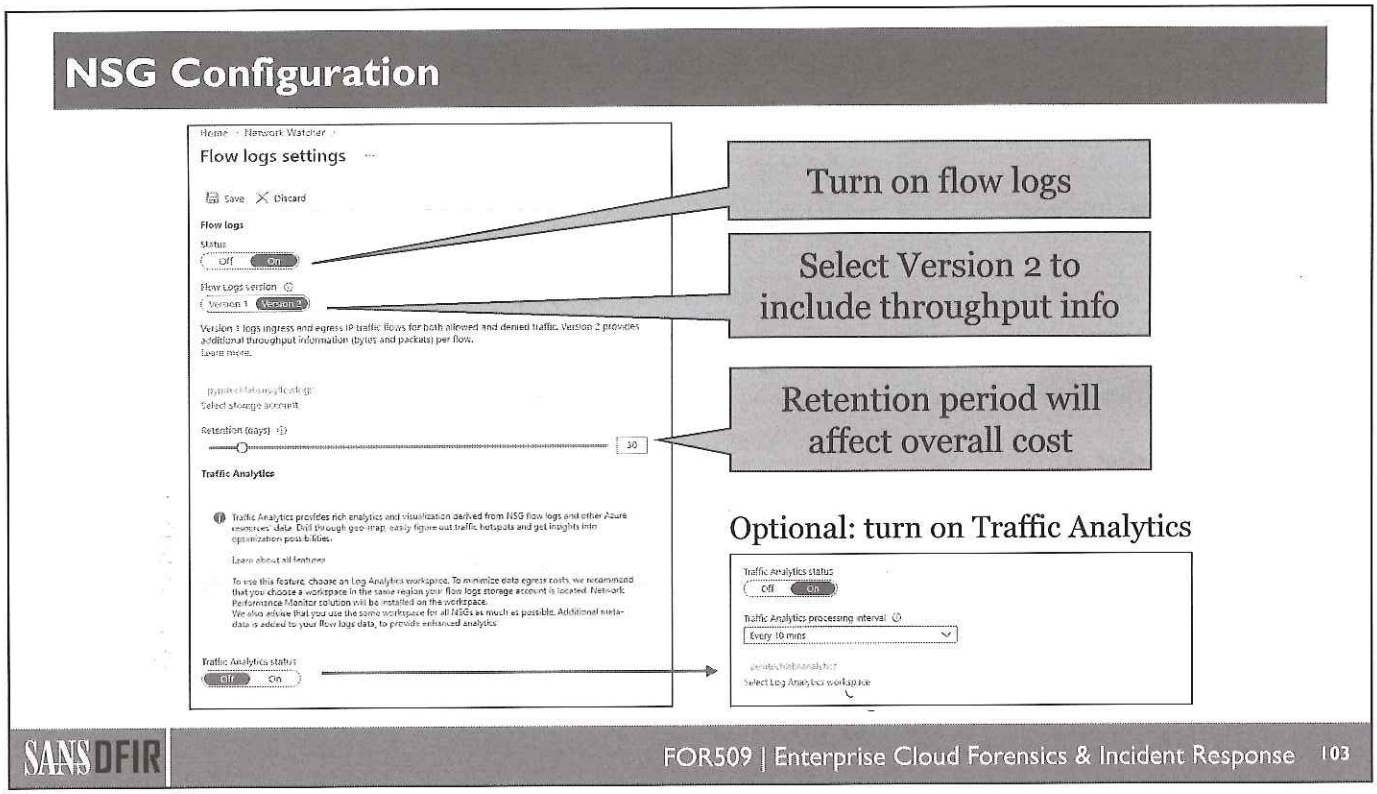
Flow logs are the source of truth for all network activity in your cloud environment and are a "must have" for any investigation.

References:

[1] https://for509.com/nsgflowlogs

Logs can also be obtained from the Virtual network (Vnet). However, they are not as valuable as the NSG flow logs. If you wish to capture these logs, you will configure them from the diagnostic settings for that specific Vnet (Research-vnet in the example below):





There are three steps to setting up an NSG Flow log:[1]

1. Enable Network Watcher. Network Watcher needs to be enabled for each region. If you have VMs in multiple regions, don't forget to enable it for every region.
2. Register Insights provider. Microsoft.Insights is the provider that enables the login and as such needs to be registered as shown in reference[1]. This needs to be done for every subscription.
3. Enable the NSG flow log. Version 2 is strongly recommended as it captures throughput information.

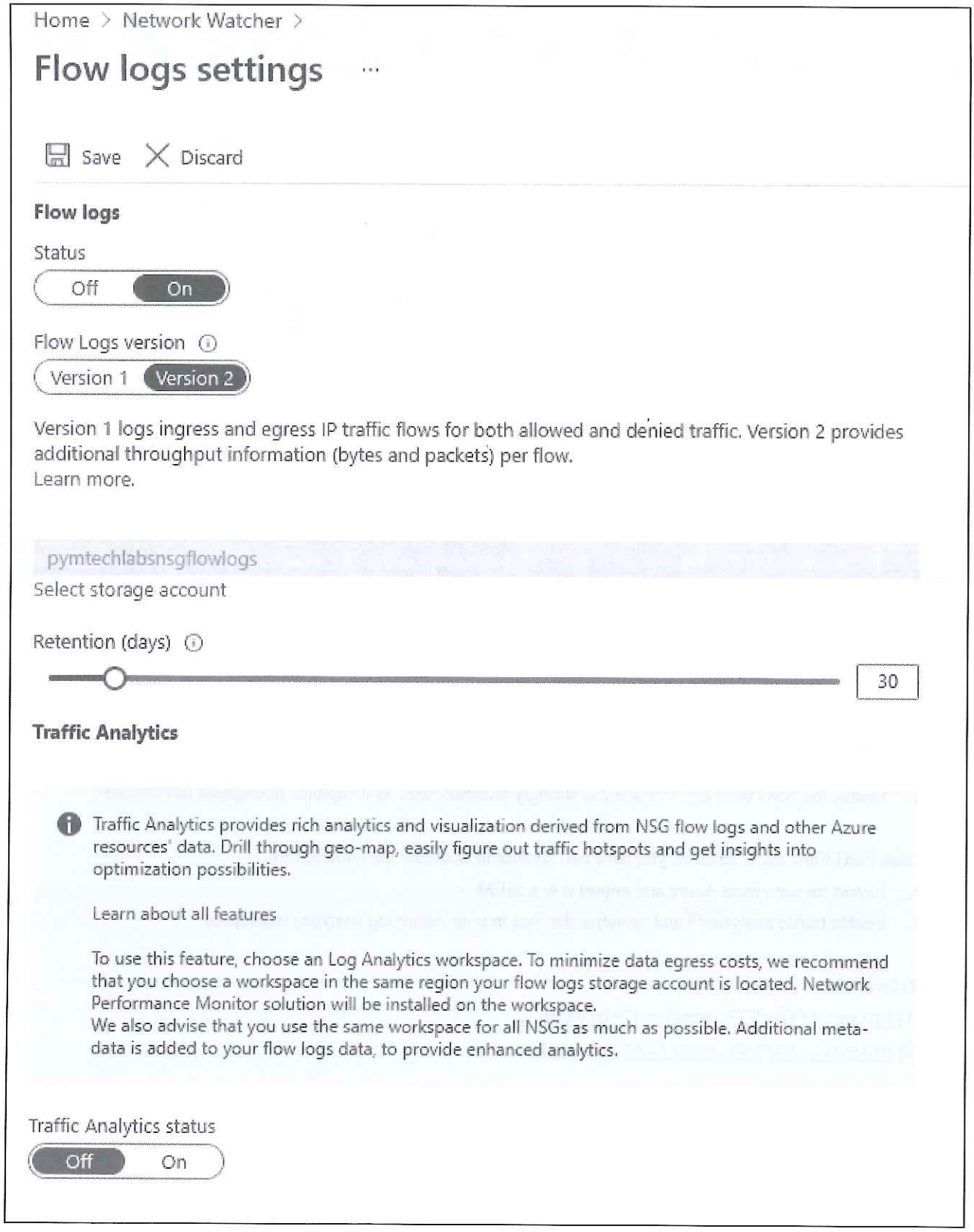
Once NSG Flow log is enabled, you have two choices to consume the information:

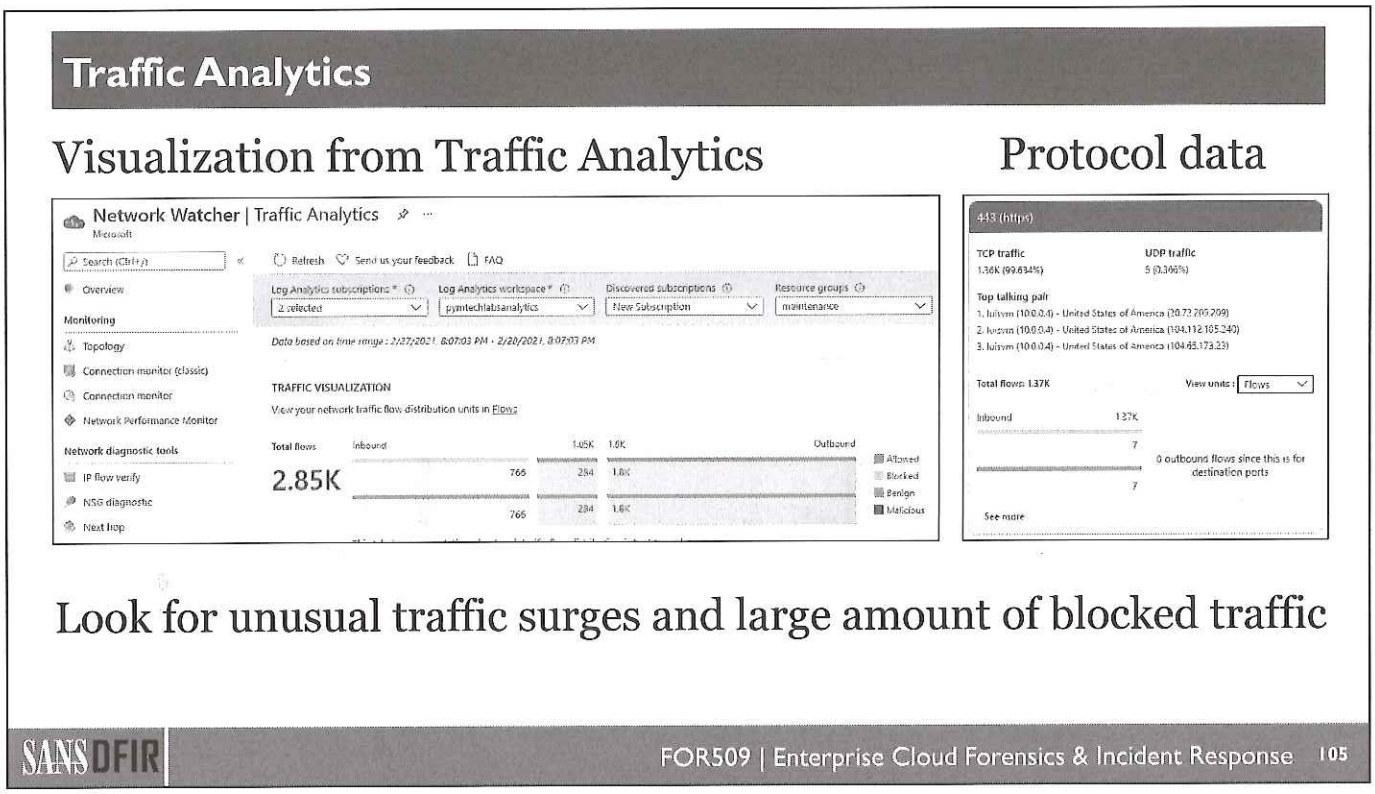
1. Export the data from Azure and import it to a SIEM.
2. Enable traffic analytics[2] and visualize the data in your Azure log analytics workspace.

References:

[1] https://for509.com/nsglogsetup

[2] https://for509.com/trafficanalytics





Traffic analytics leverages the log analytics workspace to provide insights into traffic flow. As stated by Microsoft, traffic analytics[1] enables you to:

1. Visualize network activity across your Azure subscriptions
2. Identify security threats
3. Understand traffic flow patterns and optimize your network deployment
4. Pinpoint network misconfiguration

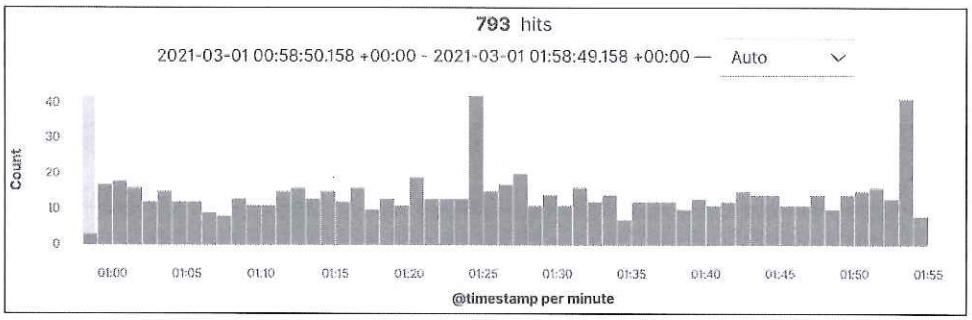
From the incident response and forensic perspective, traffic analytics is a great way to quickly identify where undesirable traffic may be coming from. While many organizations will prefer to visualize this information in their SIEM, it's good to know that Azure offers an in-cloud solution which is very simple to setup.

References:

[1] https://for509.com/trafficanalytics

NSG Flow Log – Import into SOF-ELK

- `[elk_user@sof-elk]$` **`python3 ./download_blobs_multithreaded.py`** (script provided in the notes & class github)
- `[elk_user@sof-elk]$` **`/usr/local/sof-elk/supporting-scripts/azure-vpcflow2sof-elk.py -r /directory_to_PT1H.json_files -w /logstash/nfarch/PT1H-<date>`**



NOTE

Traffic surges are easily identifiable once the NSG log is imported into SOF-ELK

Importing the NSG Flow Log to SOF-ELK is a three step process:

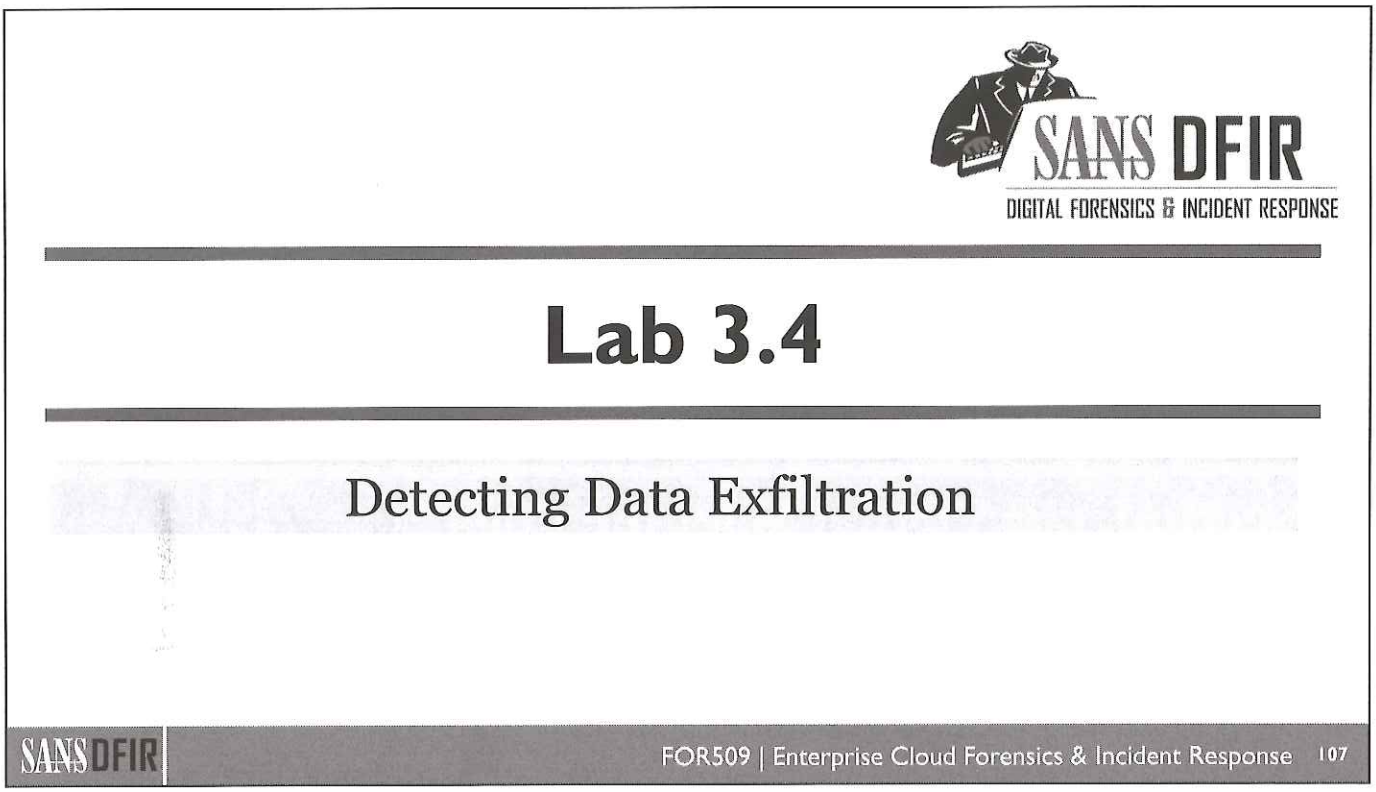
1. Transfer the NSF Flow log to your SOF-ELK VM using either Azure Storage Explorer or a python script.
2. Convert the NSF Flow log using azure-vpcflow2sof-elk.py
3. Copy the file to the `/logstash/nfarch` directory (combined with step 2 in slide example)

Once all the steps are successfully completed, the NSG flow log will be visible in SOF-ELK. In this example, we imported 793 events.

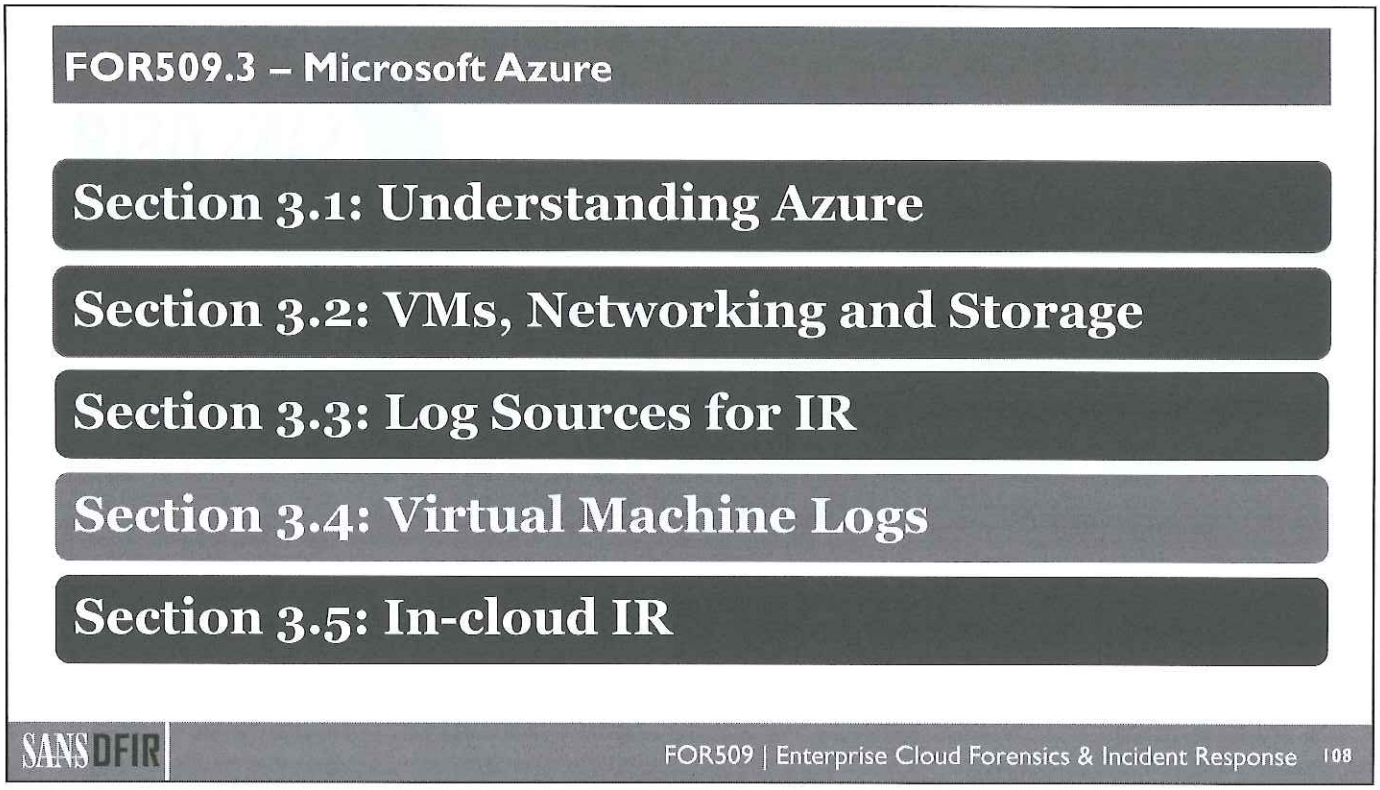
The python script to download the Azure blobs is available in reference [1]

References:

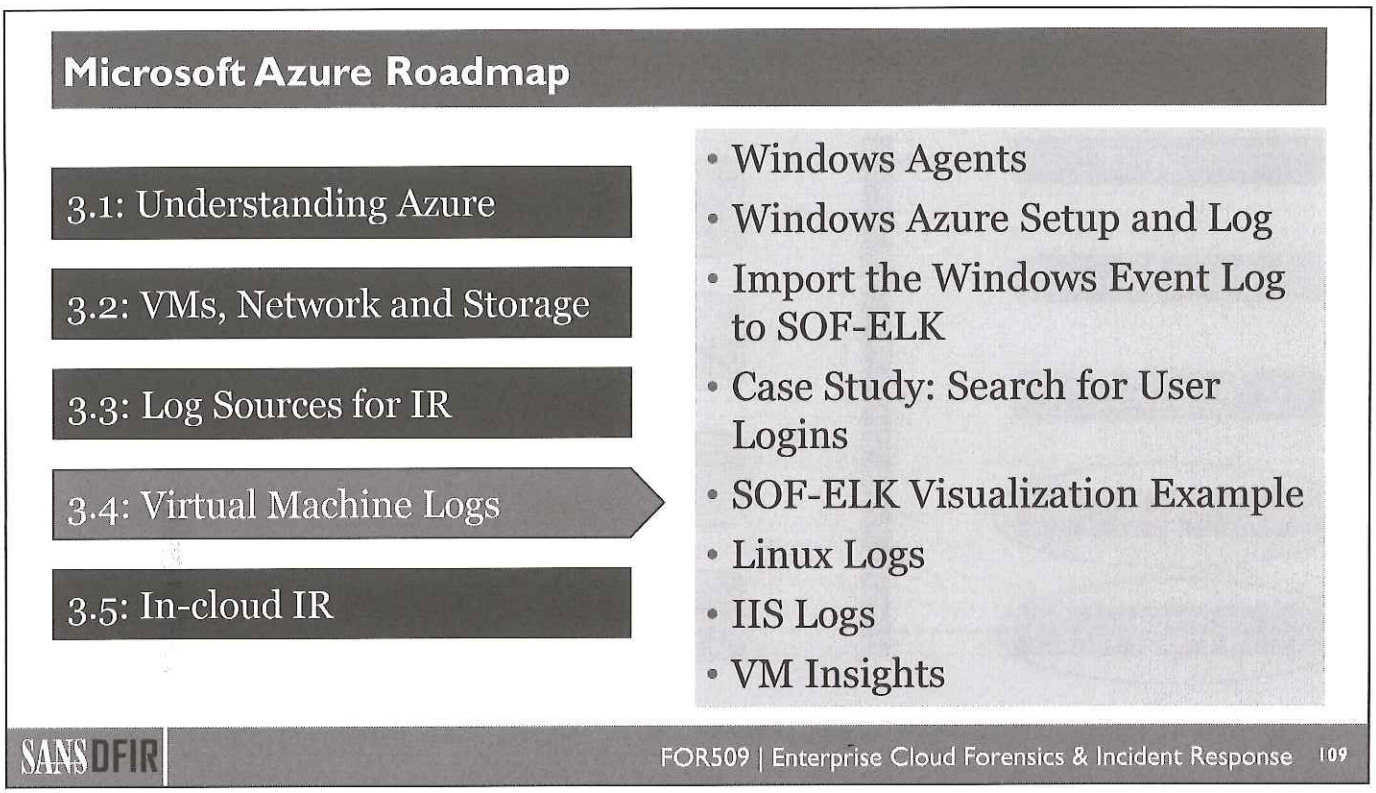
[1] https://for509.com/blobpythonscript



This page intentionally left blank.



This page intentionally left blank.



By using agents, it's possible to obtain operating system logs without ever logging into the VM. This is a great benefit to an investigation as it avoids trampling over potential evidence.

Many companies will have EDR (Endpoint Detection and Response) systems but for those that don't, this is a low-cost alternative. There is no charge for the agents. The cost is only in the storage used and the configuration offers the possibility to set a quota to minimize cost.

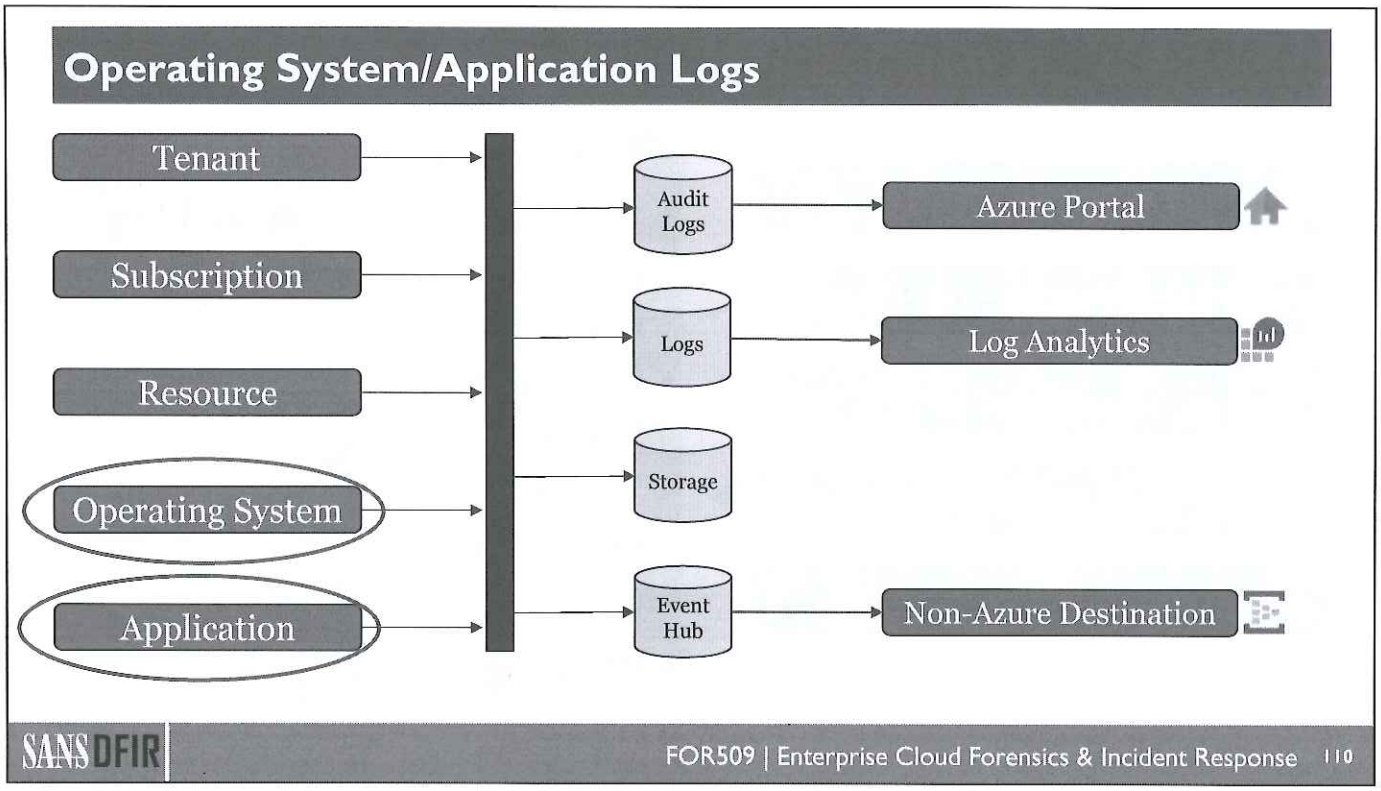
These agents have many features to monitor metrics and the health of the VMs. These aren't as interesting to us as incident responders. We will focus on the ability to obtain log information.

The Azure feature that organizes these logs and performance data is called Azure Monitor Log.[1]

We will review the Windows agent first and then the Linux one.

References:

[1] https://for509.com/azuremonitorlog



This page intentionally left blank.

Windows Agents

	Azure Monitor Agent (preview)	Diagnostics Extension (WAD)	Log Analytics Agent	Dependency Agent
Data Collected	• Event logs • Performance	• Event logs • Performance • ETW events • File based logs • IIS & .NET logs • Crash dumps	• Event logs • Performance • File based logs • IIS logs • Insights	• Process dependencies • Network connection metrics
Data Sent To	• Azure Monitor Logs • Azure Monitor Metrics	• Azure Storage • Azure Monitor Metrics • Event Hub	• Azure Monitor Logs	• Azure Monitor Logs

- Agents collect both performance metrics and logs
- 4 different agent options but most are for in-cloud consumption
- Only the Diagnostics Extension will send data to Azure Storage or an Event Hub
- Multiple agents can be installed on a VM

SANS DFIR | FOR509 | Enterprise Cloud Forensics & Incident Response 111

Azure offers 4 possible agents to collect a wide variety of metrics and logs.[1] In selecting the correct combination of agents, it's important to first consider how the data will be consumed. Only the diagnostics extension is able to write the data to a storage account or an event hub.

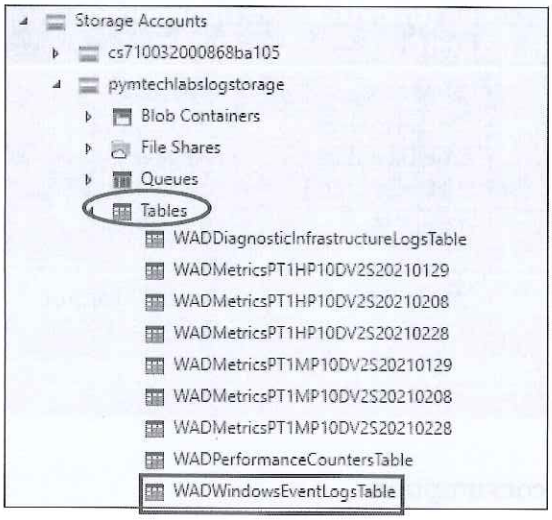
The Azure monitor agent is currently in preview and will eventually replace the log analytics agent. This is due to the recent consolidation of Azure Monitor and Log Analytics. Eventually the Azure monitor agent should also be able to write data to a storage account or to an event hub.

For the purpose of incident response and forensics, the diagnostics extension is the most interesting. Microsoft frequently refers to that feature as Windows Azure Diagnostics (WAD) and the logs start with that prefix.

References:

[1] https://for509.com/monitoragents

Windows Azure Diagnostics (WAD)



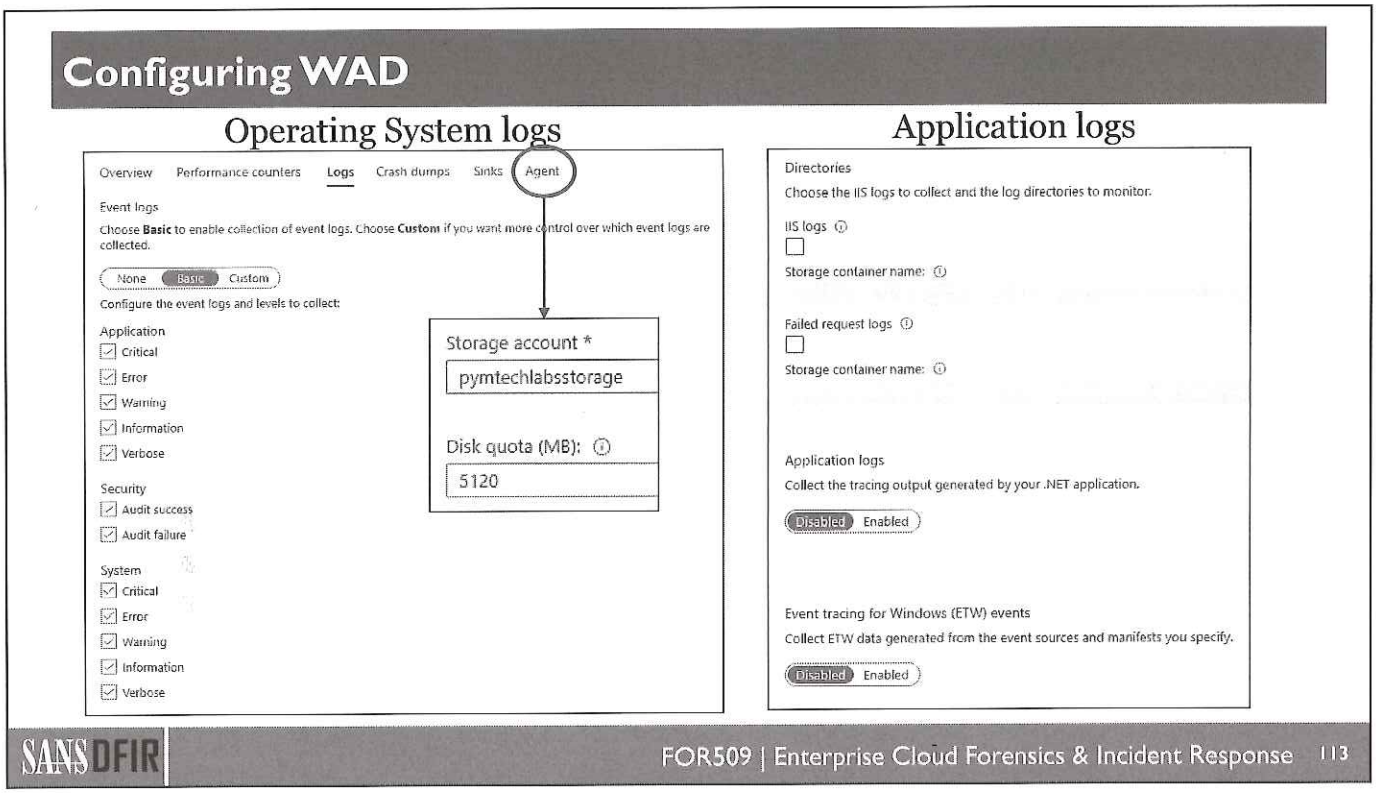
- Logs we covered earlier in the class were stored in blob containers and formatted in JSON
- WAD logs are stored in tables and exports are limited to .csv files
- SOF-ELK ingestion script is included in the latest release of SOF-ELK
- WADWindowsEventLogsTable is the most interesting table for incident response and forensics as it contains the windows event logs

The WADWindowsEventLogsTable is of particular interest because it contains Windows event logs. This is a great opportunity to obtain operating system logs without the need to login to the VM itself.

So far, all our logs have been stored in blob containers and formatted in JSON which is easy to download via the Azure Storage Explorer. As previously shown, copying these logs to the `/logstash/azure` directory of your SOF-ELK VM will make them easily accessible and searchable in Kibana.

Unfortunately, the Windows Azure diagnostics logs aren't so easily accessible. They are stored in a noSQL table in your storage account. Azure Storage Explorer can access this table and export it to a `.csv` file.

In the next slides, we will show you how to extract the information and import it to SOF-ELK.



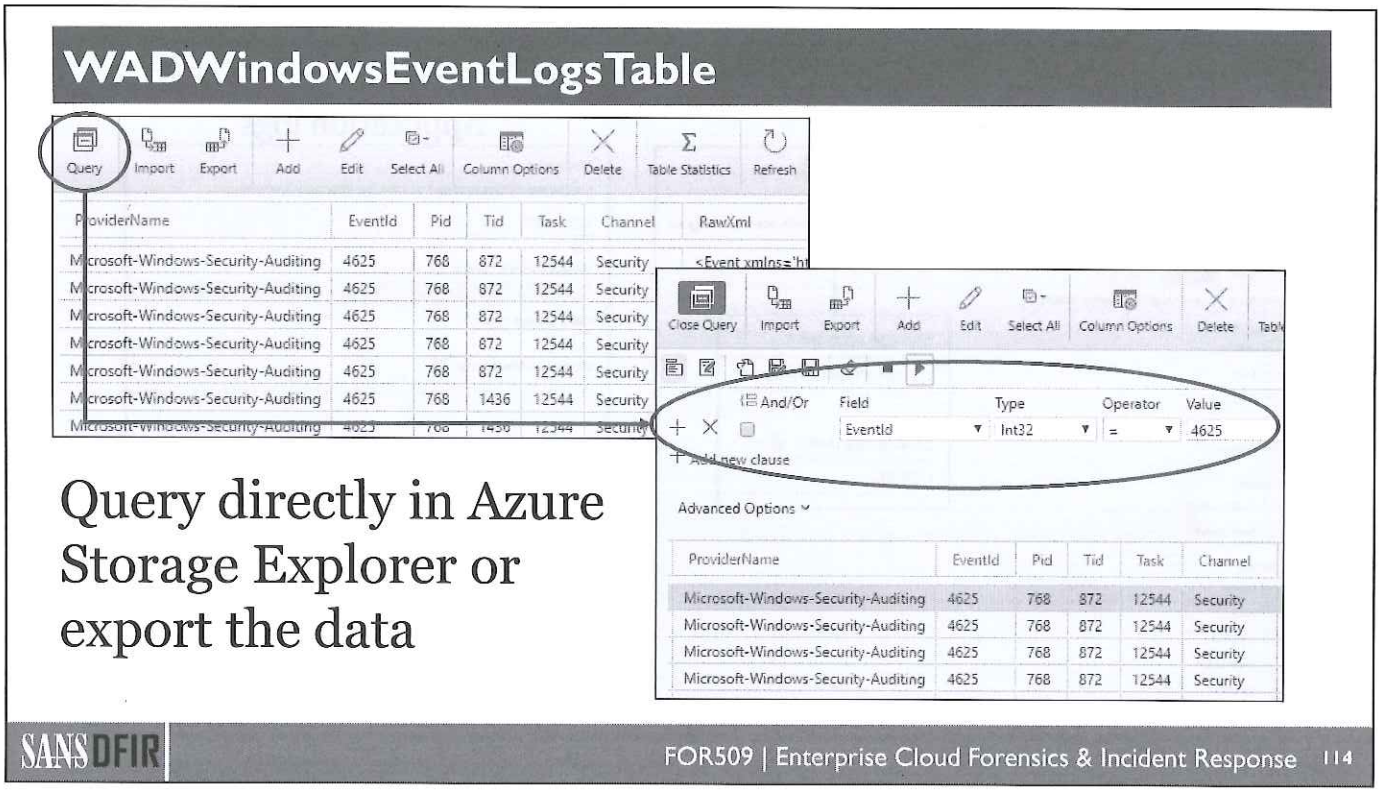
Before we can extract any logs, we need to configure WAD. First, you will select the diagnostic settings for your virtual machine. Then, under the "Logs" tab, you can select the event logs and levels that you want to collect. Finally, under "Agent" you will select the storage account to send the logs to. You can also choose a disk quota to make sure the logs don't grow indefinitely.

As with other logs, selecting everything is not recommended as you will get a lot of noise and use a lot of storage.

You may also collect logs for IIS, tracing output from .NET applications, and Event Tracing for Windows (ETW).

For the purposes of incident response, we will focus on operating system logs.

Now that these logs are collected, we need to look at the structure of WADWindowsEventLogsTable.

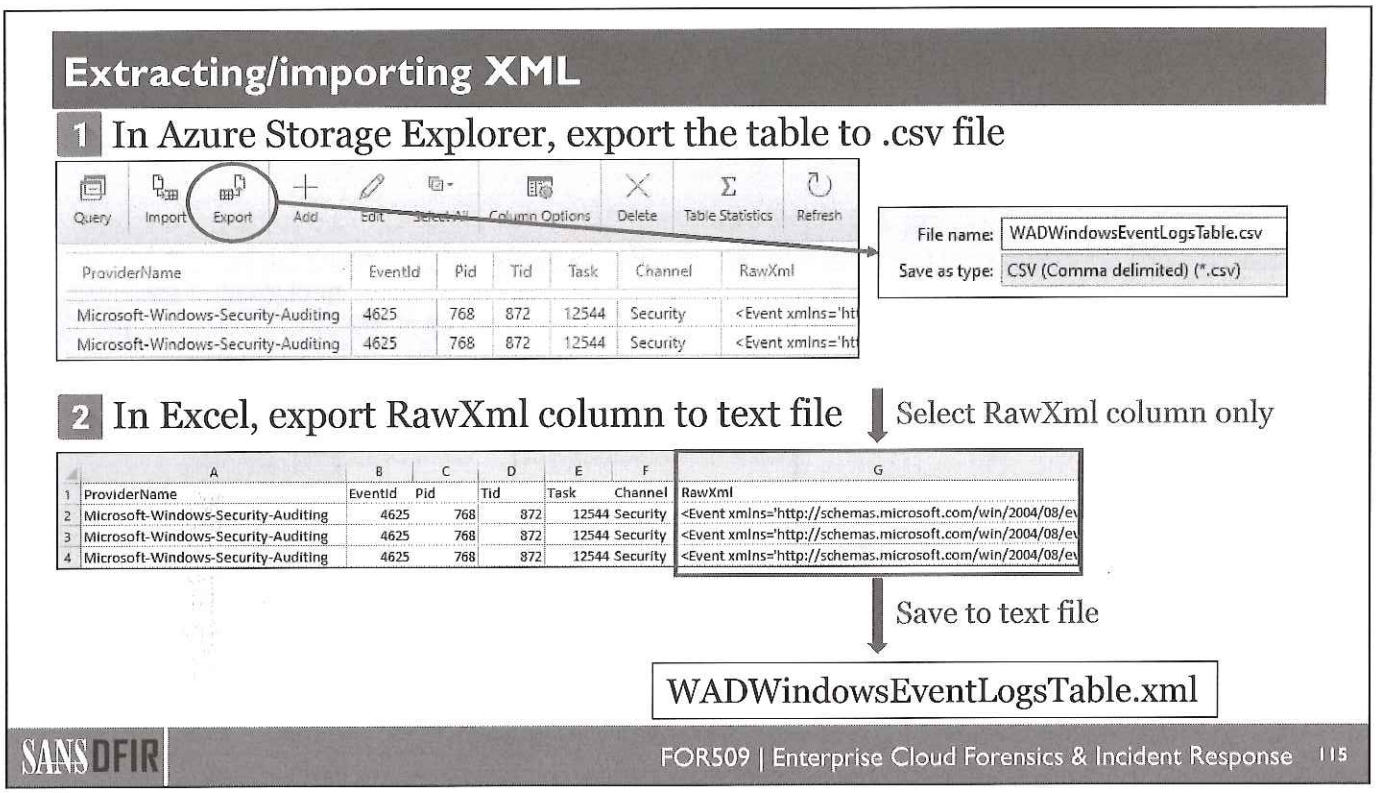


Using Azure Storage Explorer, we can examine the contents of WADWindowsEventLogsTable. You will notice a large number of columns. The column called RawXml contains all the data concerning that particular event. Every other column is derived from that column.

The benefit of breaking down the RawXml column to individual columns is that you can perform queries directly in Azure Storage Explorer. In the above examples we are looking for EventId 4625 which is "An account failed to log on".

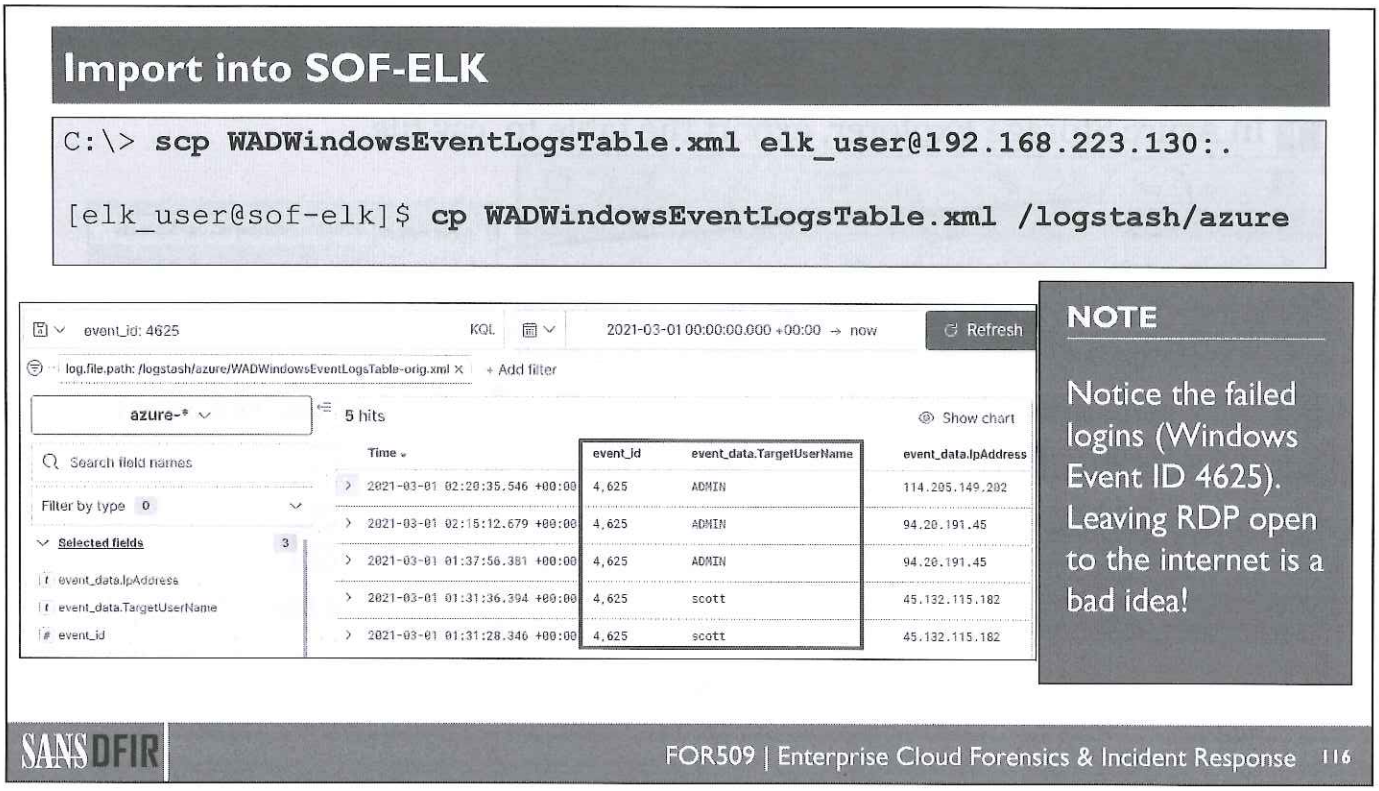
It's quite advantageous to be able to query operating system logs without needing to log into the virtual machine or even Azure. As long as Azure Storage Explorer has been configured with the storage account access key, you can query to logs directly from your computer.

While this is convenient for a quick search, for a larger investigation it's preferable to import these logs into a SIEM platform like SOF-ELK and have the ability to corelate these events with other logs.



SOF-ELK has a Logstash import script to process the RawXml column. We need to extract that column from the WADWindowsEventLogsTable. Many solutions are possible, and in this slide, we propose a simple one that consists of two steps:

1. In Azure Storage Explorer, export the table to a .csv file
2. In Excel (or any other tool of your choice), export the RawXml column to a text file.
   a) Be sure to name that file with a .xml extension as the Logstash script will expect it.
   b) Don't forget to remove the header row (1st row)



To ingest the xml file into SOF-ELK, follow these two steps:

1. Copy the xml file from your computer to the SOF-ELK VM. `scp` is the easiest way to perform this step. You will need to enter the correct IP address for your SOF-ELK VM. The password for the elk_user account is `forensics`
2. Copy the xml file to the `/logstash/azure` directory. ELK will start ingesting the data right away. Depending on the size of the file you may have to wait a few minutes before the data shows up in Kibana.

Now that the data is available to us, we can easily query for failed logons like we did in Storage Explorer (event ID 4625). You will notice 3 failed logons with a username of ADMIN and two with a username of scott. The ADMIN failed logons demonstrate the danger of leaving a VM with port 3389 open on the Internet.

Events are logged to the Windows event log by various providers. Each provider formats their log entry differently making parsing these events quite challenging. In order to speed up the data ingestion, the Logstash script is only processing events from the "Microsoft-Windows-Security-Auditing" provider.

Windows Event Field Mapping in SOF-ELK

- The event ID is shown in field: event_id
- The other windows event fields are mapped to event_data.<field name>
- Fields will vary depending on the event
- Windows VMs tend to have very noisy logs so it's best to filter by event_id first

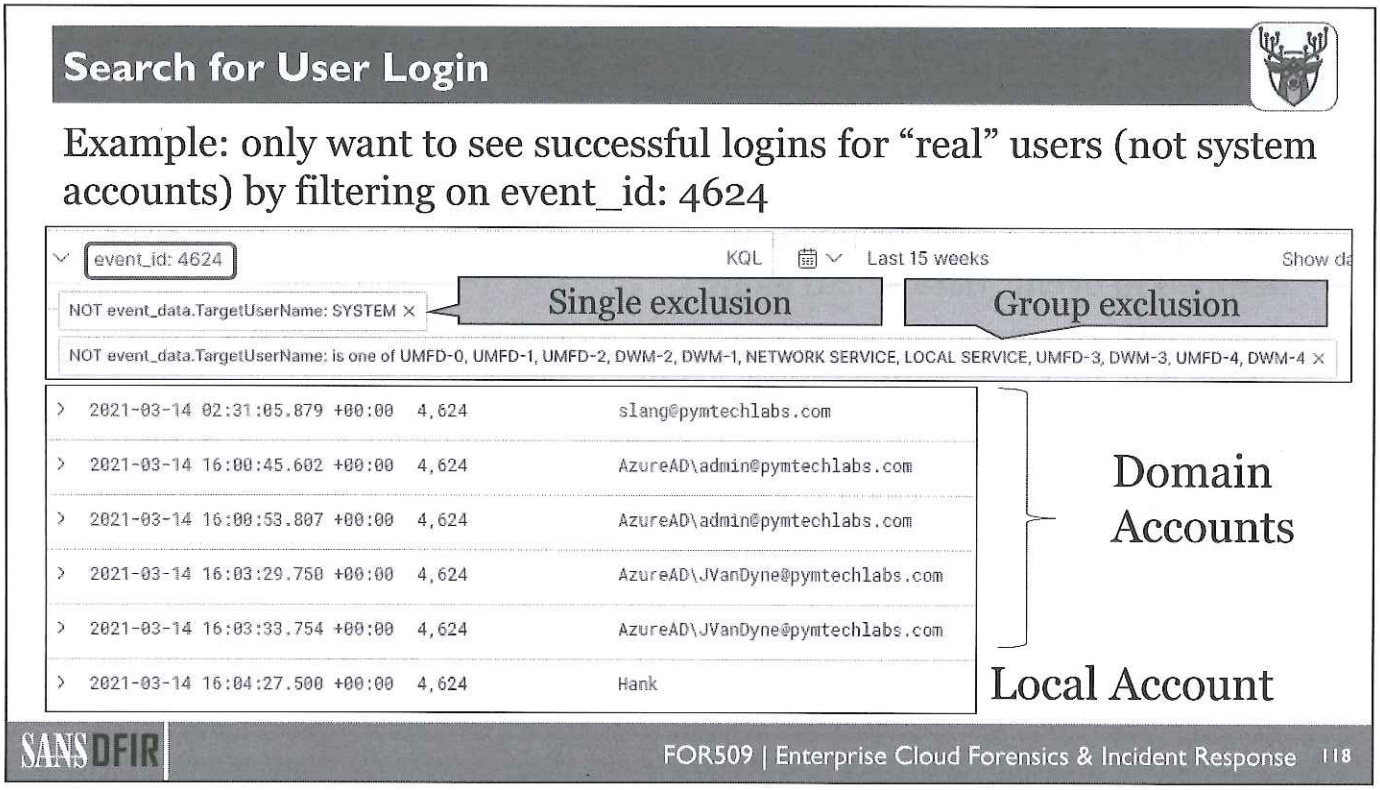
# event_id	4,624

t event_data.LogonType
t event_data.ProcessId
t event_data.ProcessName
t event_data.RestrictedAdminMode
t event_data.SubjectDomainName
t event_data.SubjectLogonId
t event_data.SubjectUserName
t event_data.SubjectUserSid
t event_data.TargetDomainName

Windows event logs are mapped in SOF-ELK under the `event_data.<field name>`.

Different fields will be available depending on the event id.

Azure Windows VMs tend to log a lot of information if the audit success and failure options are selected. It's best to first filter for a specific event id and then start removing noisy events; for example, the SYSTEM user.



If we want to see "real" user logins, it's not enough to filter for event id 4624. We also need to filter out all the system accounts. ELK supports filtering a single item at a time:

```
NOT event_data.TargetUserName: SYSTEM
```

Or you can filter a group of items with the operator "is one of":

```
NOT event_data.TargetUserName: is one of UMFD-0, UMFD-1, UMFD-2, etc.
```

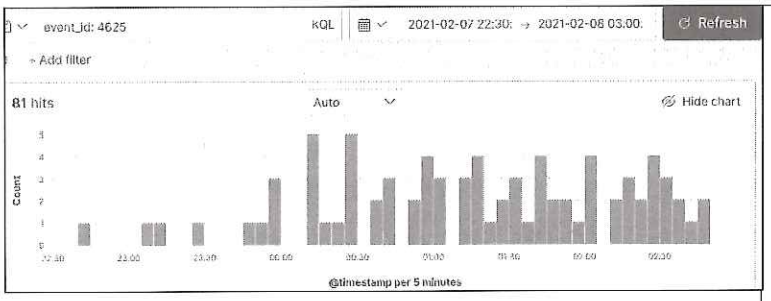
While a bit tedious, you only need to do it once as you can save your search.

For future reference:

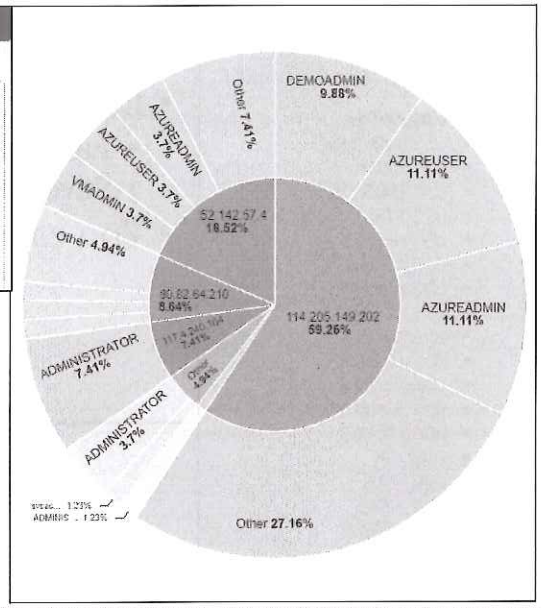
- UMFD-* are system accounts generated by the User Mode Driver Framework and is used by the Usermode Font Driver Host process (fontdrvhost.exe)

- DWM-* accounts are associated with the Desktop Window Manager process (dwm.exe)

SOF-ELK Visualization Example

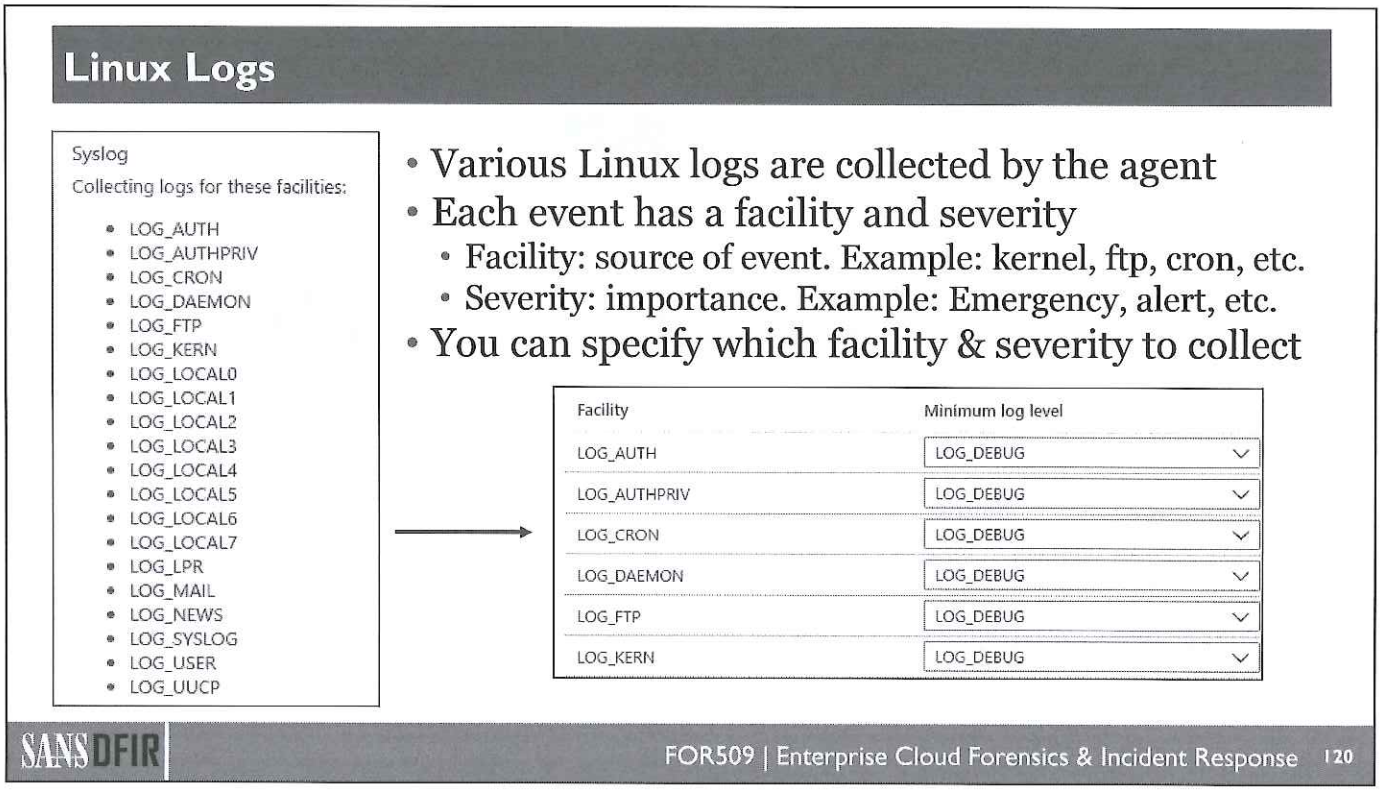


- Windows VM with RDP open to the Internet
- Four-hour time window: 81 failed login attempts
- Notice the accounts being "tested"



SANS DFIR | FOR509 | Enterprise Cloud Forensics & Incident Response 119

We have looked at numerous logs and now that they are imported in SOF-ELK we can create interesting visualizations. This example illustrates the danger of leaving port 3389 (RDP) opened to the internet. In a 4-hour window multiple unauthorized actors attempted to guess passwords to accounts that don't even exist. The choice of account names is noteworthy, if anything to avoid using these names for your real system accounts.



To setup logging for a Linux machine, you will select the diagnostic settings for your virtual machine. Similar to other diagnostic settings, you will need to select the storage account that you wish to use to store the logs.

For Linux machines, you have two tabs: metrics and syslog. The metrics tab will allow you to select the sample rate for Processor, Memory, Network, File System, and Disk. Metrics can sometimes be useful in incident response. For example, an investigation of crypto mining may show a high-processor utilization, or an investigation of ransomware may show high disk and file system utilization.

However, we are mostly interested in the syslog configuration. On that tab, you will see a number of "Facility" options. A facility represents the machine process that created the syslog event. For example, that could be the kernel, ssh daemon, mail system, etc. On a Linux machine these logs may be stored in separate files: auth.log, kern.log, syslog, etc. Azure combines everything in a single table. If you want to learn more about the syslog protocol, please see RFC 5424.[1]

The last selection you need to make is the log level. There are 7 log levels:

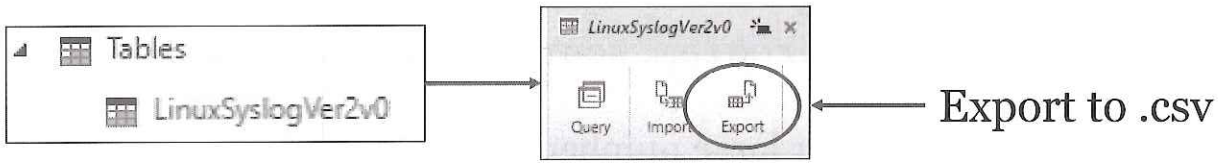
1. **Debug**: very verbose logs, mostly used to debug problems.
2. **Info**: informational messages – no action required.
3. **Notice**: normal but significant condition.
4. **Err**: error condition, but non-urgent failure.
5. **Crit**: critical condition, should be corrected immediately as there is a failure.
6. **Alert**: action must be taken immediately.
7. **Emerg**: emergency, the system in unusable.

References:

[1] https://for509.com/rfc5424

LinuxSyslogVer2v0 Table

Linux logs can be accessed in Azure Storage Explorer under Tables -> LinuxSyslogVer2v0



Example of user scott login to host UbuntuMachine using ssh

EventTime	Facility	Host	Msg	Severity	ident	pid
2021-03-16T01:21:45+0000	auth	UbuntuMachine	Accepted password for scott from 45.56.183.51 port 53501 ssh2	info	sshd	7085
2021-03-16T01:21:45+0000	authpriv	UbuntuMachine	pam_unix(sshd:session): session opened for user scott by (uid=0)	info	sshd	7085
2021-03-16T01:21:45+0000	auth	UbuntuMachine	New session 27 of user scott.	info	systemd-logind	1083
2021-03-16T01:21:45+0000	daemon	UbuntuMachine	Started Session 27 of user scott.	info	systemd	1

The logs are found in a table called LinuxSyslogVer2v0. While the logs can be reviewed and searched inside Azure Storage Explorer, exporting them to a .csv file provides more options.

The .csv file will need a bit of cleaning up and you should focus on the following fields:

- **EventTime**: the time at which the event occurred. The log contains numerous timestamps, but this is the most important one.
- **Facility**: as described in the previous slide.
- **Host**: the name of the machine (there is a redundant field called hostname).
- **Msg**: the most important field that contains the actual event.
- **Severity**: as described in the previous slide.
- **Ident**: the process that generated the event.
- **Pid**: the process id.

The example above shows user scott logged into a machine called UbuntuMachine using ssh with password authentication (a bad practice, should be using a certificate!).

Excel is a great tool for analyzing a small amount of data. However, importing this log in SOF-ELK is preferable when dealing with multiple hosts and to correlate the activity with other logs.

LinuxSyslogVer2v0 Table to SOF-ELK

To search syslog in SOF-ELK:

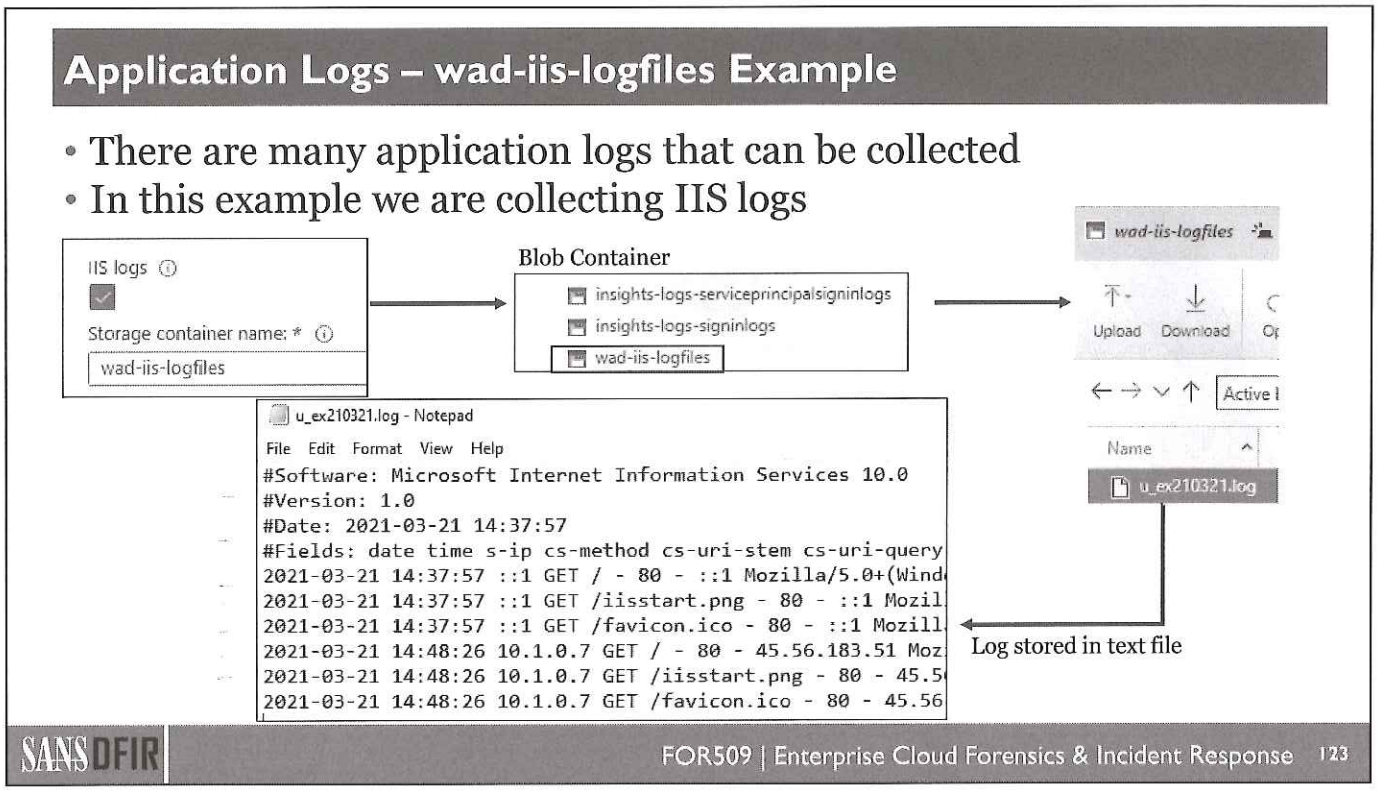
- Syslog events are in the `logstash-*` index
- Filter out CRON events
- Select fields `syslog_program`, `source_ip`, `message`
- Better way to search for large number of events than Azure Storage Explorer

Time	syslog_program	source_ip	message
2021-03-16 01:21:45.000Z	systemd	-	Started Session 27 of user scott.
2021-03-16 01:21:45.000Z	systemd-logind	-	New session 27 of user scott.
2021-03-16 01:21:45.000Z	sshd	45.56.183.51	Accepted password for scott from 45.56.183.51 port 53501 ssh2
2021-03-16 01:21:45.000Z	sshd	-	pam_unix(sshd:session): session opened for user scott by (uid=0)

SOF-ELK will easily give us the same results as long as we select the correct index. Syslog events are standardized across most unix platforms so SOF-ELK will store these events in the `logstash-*` index (not `azure-*` as most of our other logs).

`cron` (unix equivalent of the task scheduler) is very noisy, so we need to eliminate these events with the filter `NOT syslog_program: CRON`. You can add additional filters if you have other noisy processes.

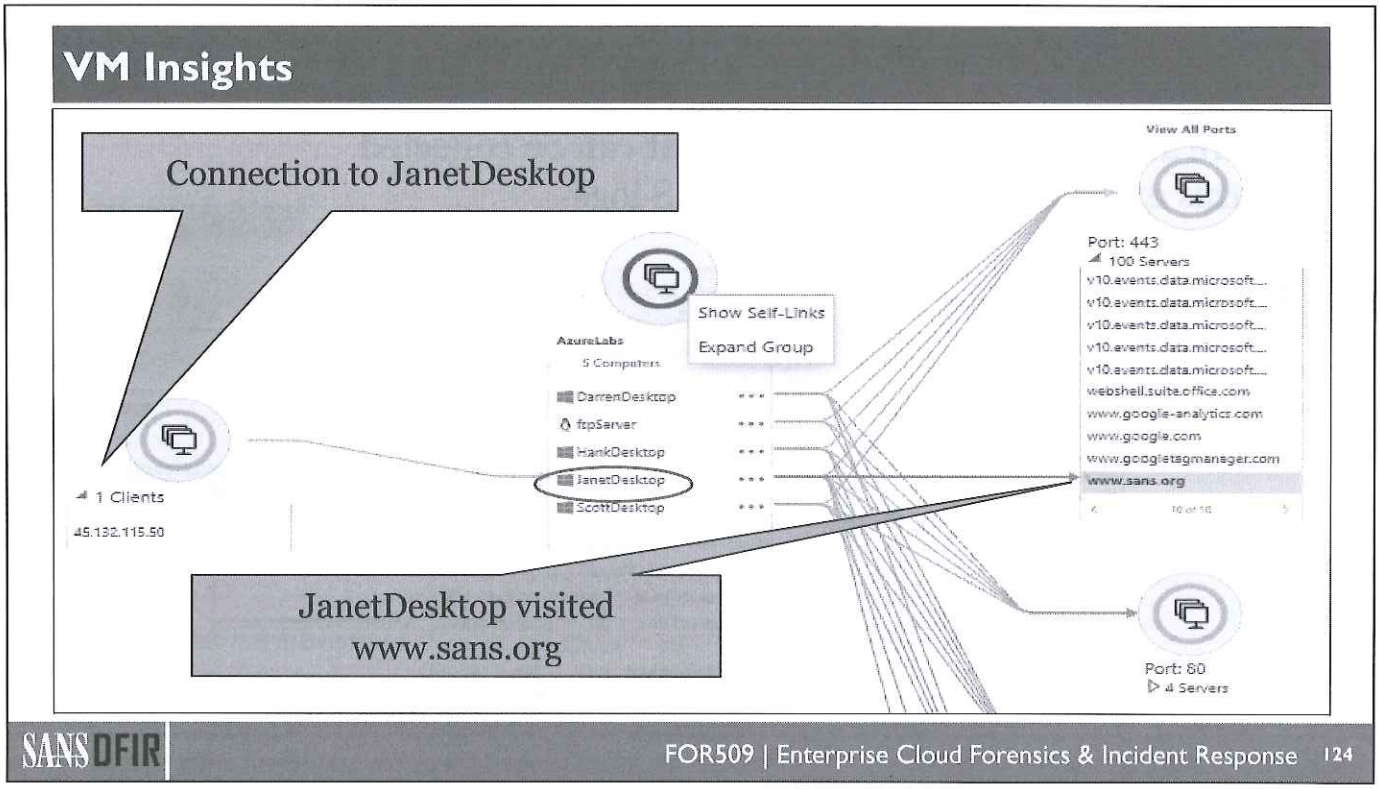
By selected the fields `syslog_program`, `source_ip` and `message`, you will get a nice table that will help you easily all logins to the machine.



There are many application logs that can be collected by the diagnostic agent. This includes tracing output generated by your .NET application and Event Tracing for Windows (ETW) events.

In case you are not familiar with ETW, it's the ability to capture kernel and application events in order to diagnose system and application performance issues.

In this example we are collecting IIS logs which are stored in a blob called `wad-iis-logfiles`. The log is stored in a plain text file which is different from the other logs we have seen earlier in this class which were all in JSON format.



Azure has an interesting feature called VM Insights.[1] When enabled, this feature allows you to visualize various components for Windows and Linux virtual machines. The primary purpose of VM Insights is to monitor the performance and health of virtual machines.

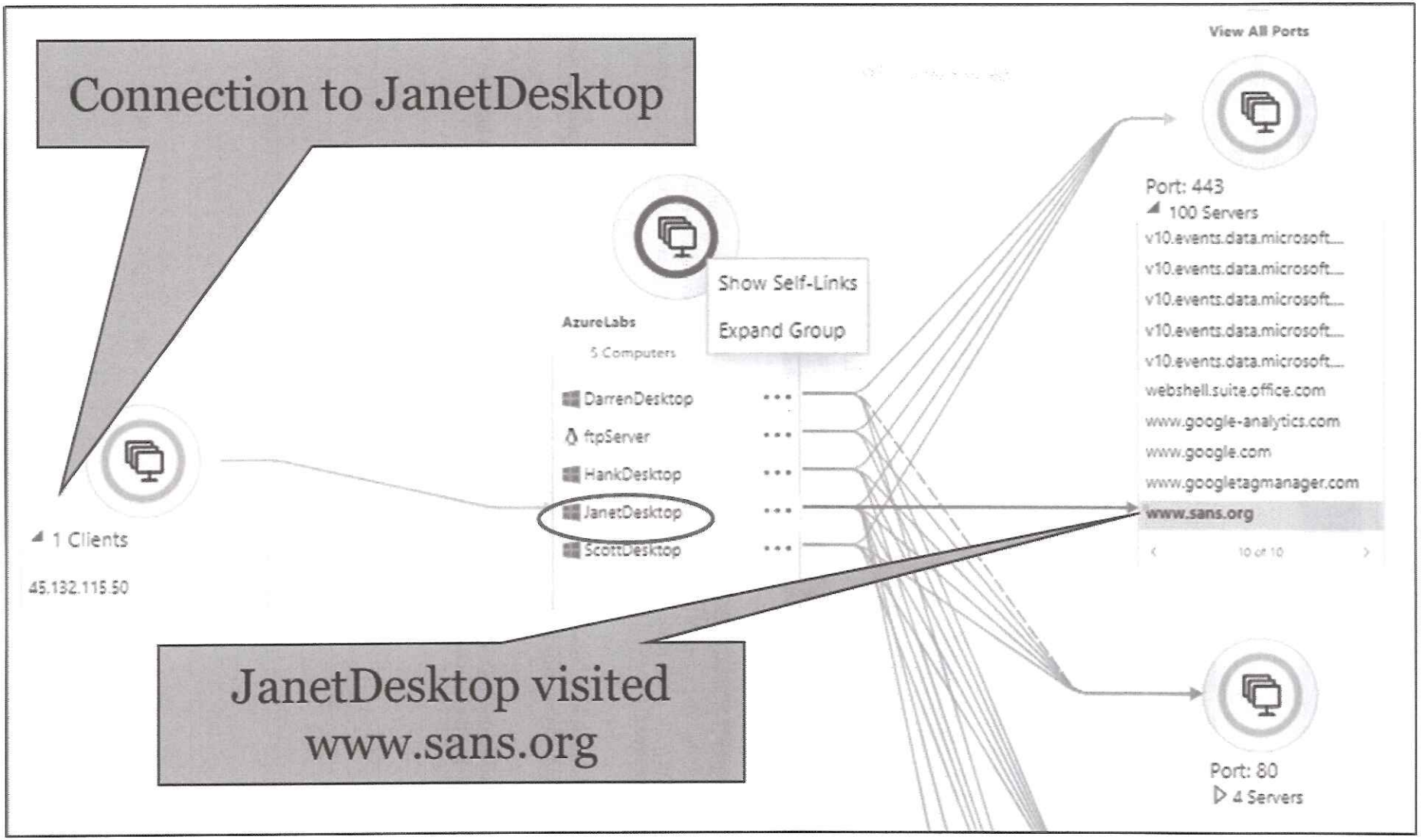
However, one of the features of VM Insights is to display a map of the environment that shows the connections to and from each virtual machine. This can be very valuable if you need to investigate an incident in an environment that's new to you.

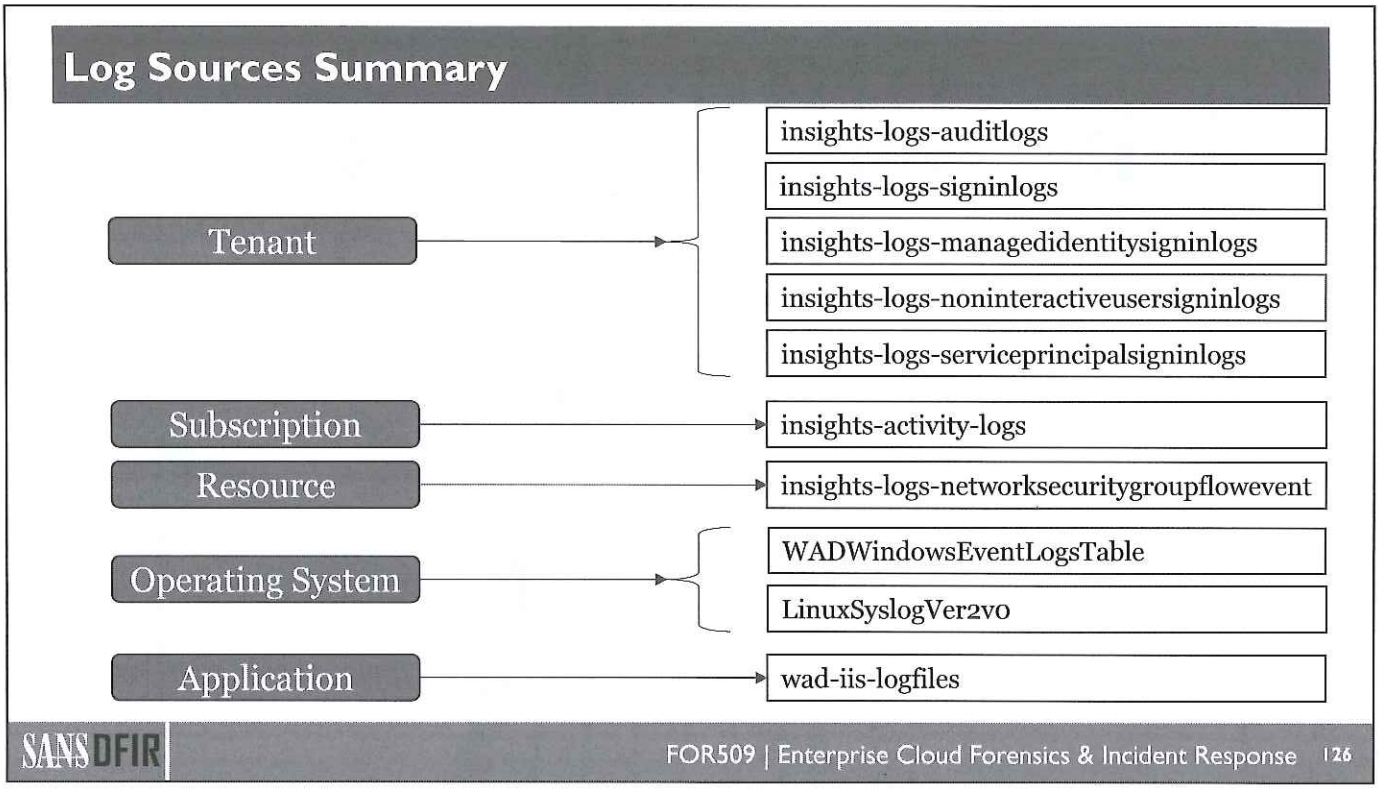
In this example, you can see on the left side that a machine located at IP 45.132.115.50 connected to JanetDesktop. In the middle you see every VM that's been configured with VM Insights. On the right side, you see every outbound connection. By selecting a specific outbound connection, it will show you which VM initiated that connection.

To enable VM Insights, you will need a log analytics workspace and to configure each virtual machine to send data to that workspace.

References:

[1] https://for509.com/vminsightsmap

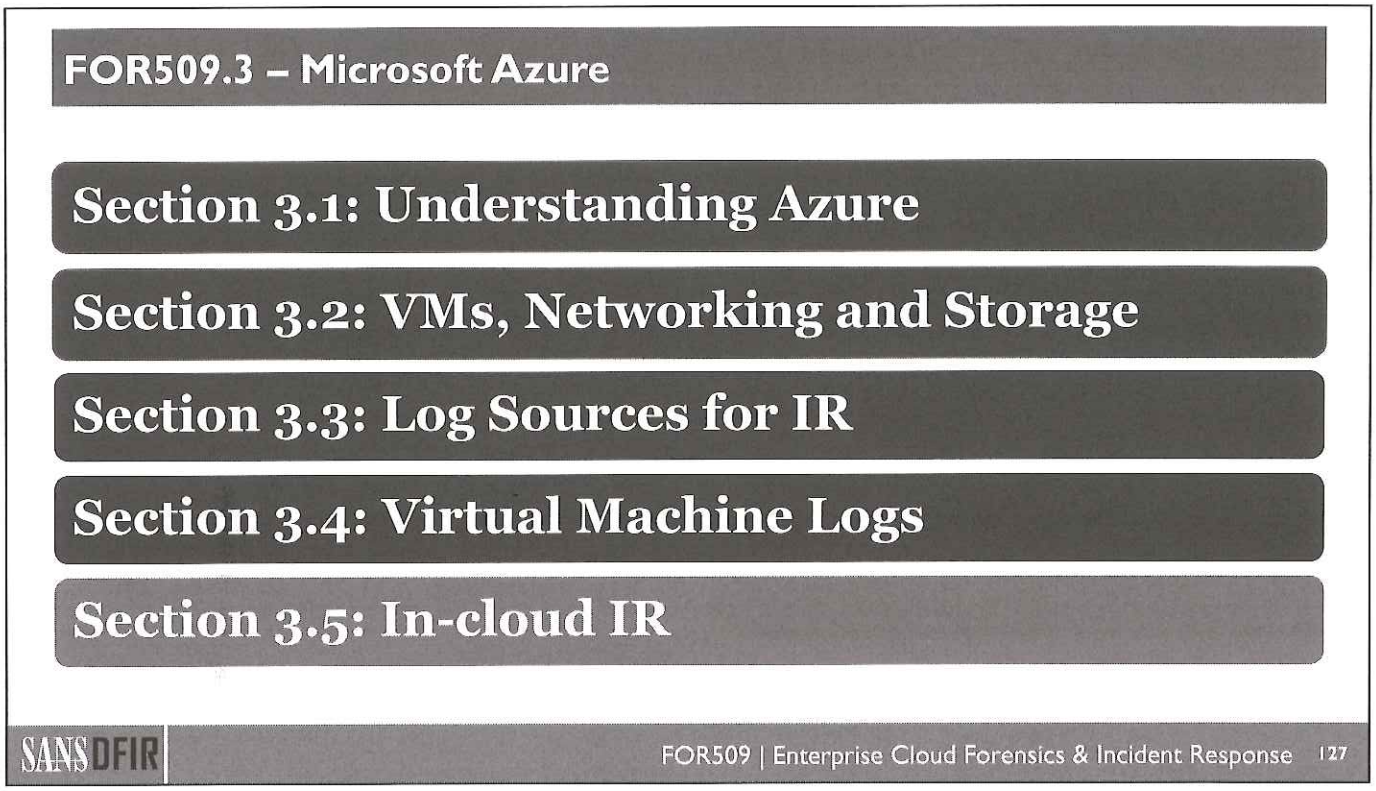




We have looked at numerous log sources. In this slide, we summarize these log sources with their names. These are just the ones we have discussed in this class and are most relevant for incident response and forensics. There are many more log sources available in Azure, but they usually focus on the performance and health of resources.

The tenant, subscription, resource, and application logs are found in container blobs. The operating system logs are found in tables which adds some complexity when trying to export them.

We have created Logstash ingestion scripts for all these log sources (except IIS application log since it's a plain text file) so that you may import and analyze them in SOF-ELK.



This page intentionally left blank.

Microsoft Azure Roadmap

- 3.1: Understanding Azure
- 3.2: VMs, Network and Storage
- 3.3: Log Sources for IR
- 3.4: Virtual Machine Logs
- 3.5: In-cloud IR

- Imaging a Drive in the Cloud
- In-Cloud Investigations
  - Snapshots
  - Create a Forensic VM
  - Run Forensic Tools
- Forensic VM Portability
- Azure Defender
- Microsoft Defender for Identity
- Azure Sentinel

This page intentionally left blank.

Imaging a Drive in the Cloud

Once we have reviewed all the logs and still need more data, how do we image a drive in the cloud?

The "old" method with a disk duplicator is clearly not possible

For large investigations, it's possible to use Azure's Import/Export service

Otherwise, an in-cloud investigation is the most efficient option

We have spent a lot of time reviewing all the log sources available to us for our investigations. There comes a point where we need the actual data from the machine. This section will take you through the steps necessary to acquire an image of a VM.

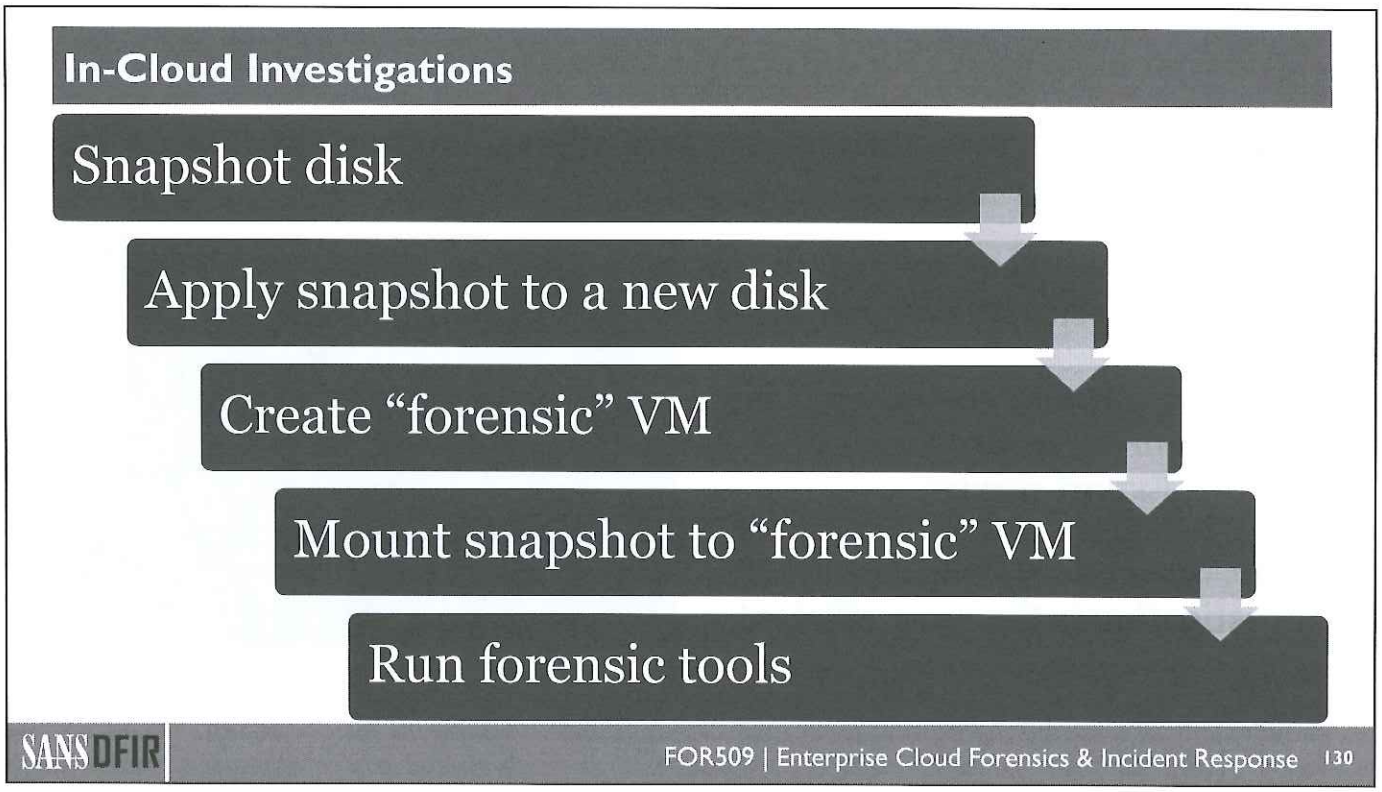
Once an image has been acquired, what do we do with it? While downloading it to our traditional forensic workstation may sounds like the easiest option, there is a cost for data egress. Given disk sizes in the 100s of gigabytes, that cost can be significant. Further, downloading a large amount of data may take a lot of time which would hinder our investigation.

The solution is to perform the forensic analysis in-cloud. For that purpose, we will create a new VM, called "forensic VM", to access the imaged disk, therefore maintaining the integrity of the original VM which we will designate as "victim VM".

If you are facing an investigation requiring a large amount of data from Azure, you can use the Azure Import/Export service to request that data.[1]

References:

[1] https://for509.com/exportservice



Our in-cloud investigation will follow these 5 steps:

1. Snapshot the OS disk from the "victim VM"
2. Create a new disk based on the contents of the snapshot
3. Create a new VM with our forensic tools: "forensic VM"
4. Mount the disk from step 2 to "forensic VM"
5. Run your favorite forensic tool

Step 1a: Snapshot VM's Disk

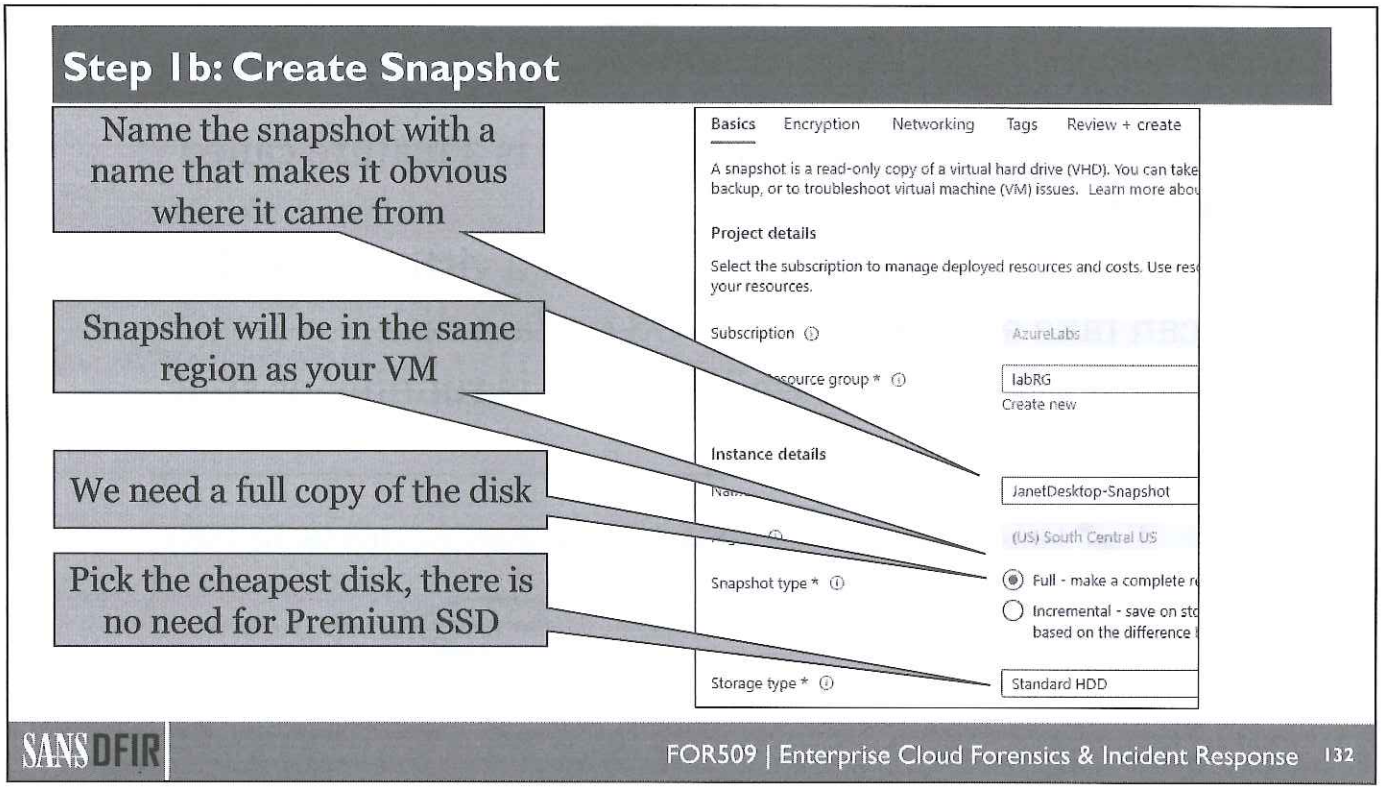
The key to imaging a drive in the cloud is a feature called **snapshot**.

- A snapshot is a full, read-only copy of a virtual hard drive
- You can take a snapshot of an OS or data disk
- Snapshots can be taken while the VM is running or shut down

A snapshot is a full, read-only copy of a virtual disk. It's an amazing technology that allows you to make a copy of a disk in just seconds. You can snapshot a disk even when the VM is running.

To create a snapshot, you will select the VM and then the disk. From there, you will have an option to create a snapshot.

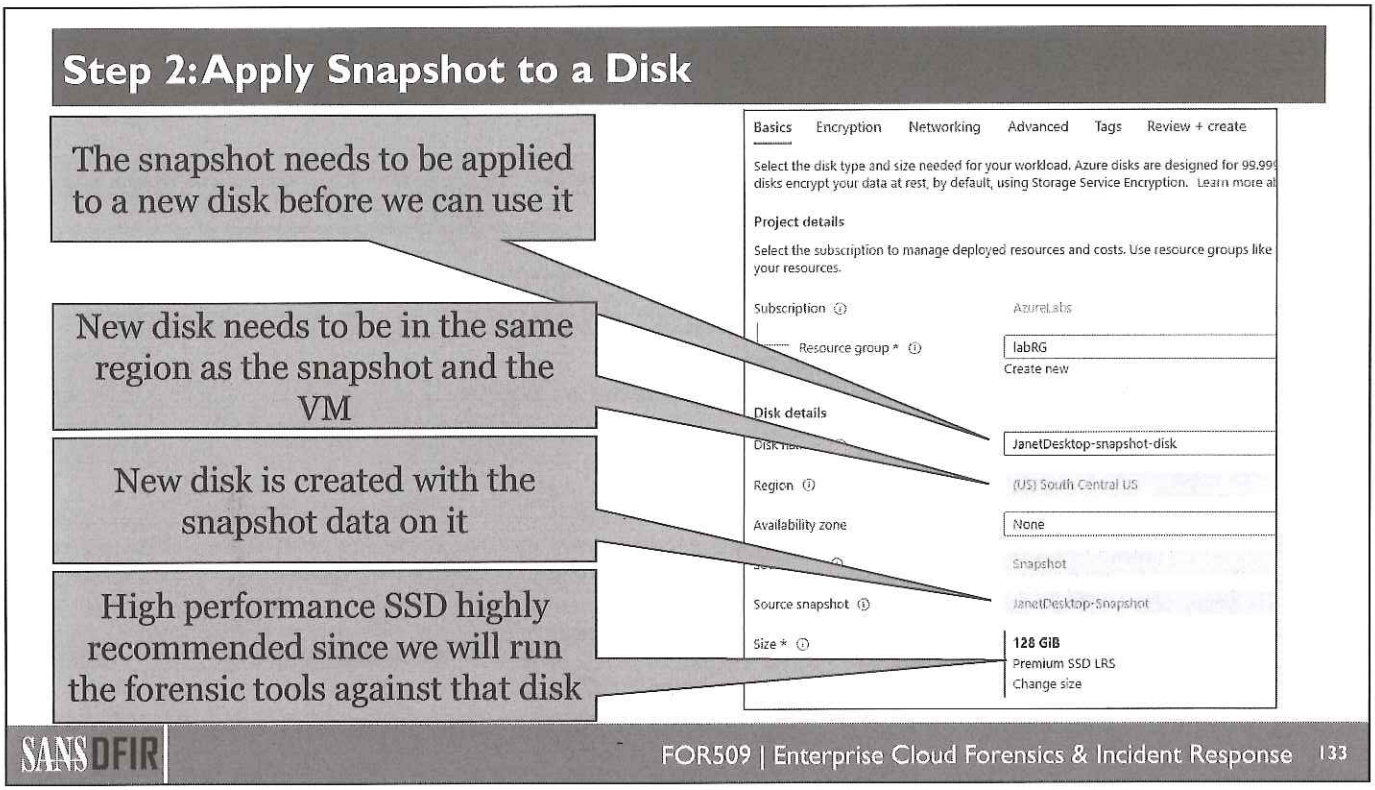
For most investigations, it should be sufficient to snapshot the operating system (OS) disk. However, if required you may also snapshot any data disk associated with that VM. Be aware of the on-going costs associated with snapshots ($0.05/GB/month for standard storage and $0.132/GB/month for premium storage).



When creating a snapshot, it's important to give it a name that quickly identifies it as a snapshot. The reason is that snapshots can't be used as-is. They need to be applied to a new disk (which we will see in the next step). When performing that step, you want to be sure to select the correct snapshot, hence the importance of a descriptive name.

For the purposes of our investigation, we want a point-in-time copy of the entire disk. Hence, we will select "Full" in the snapshot type. The "Incremental" choice is used when using snapshots for on-going backups.

Since the snapshot will be applied to a new disk, there is no need to spend money on Premium storage at this point. We recommend using "Standard HDD".

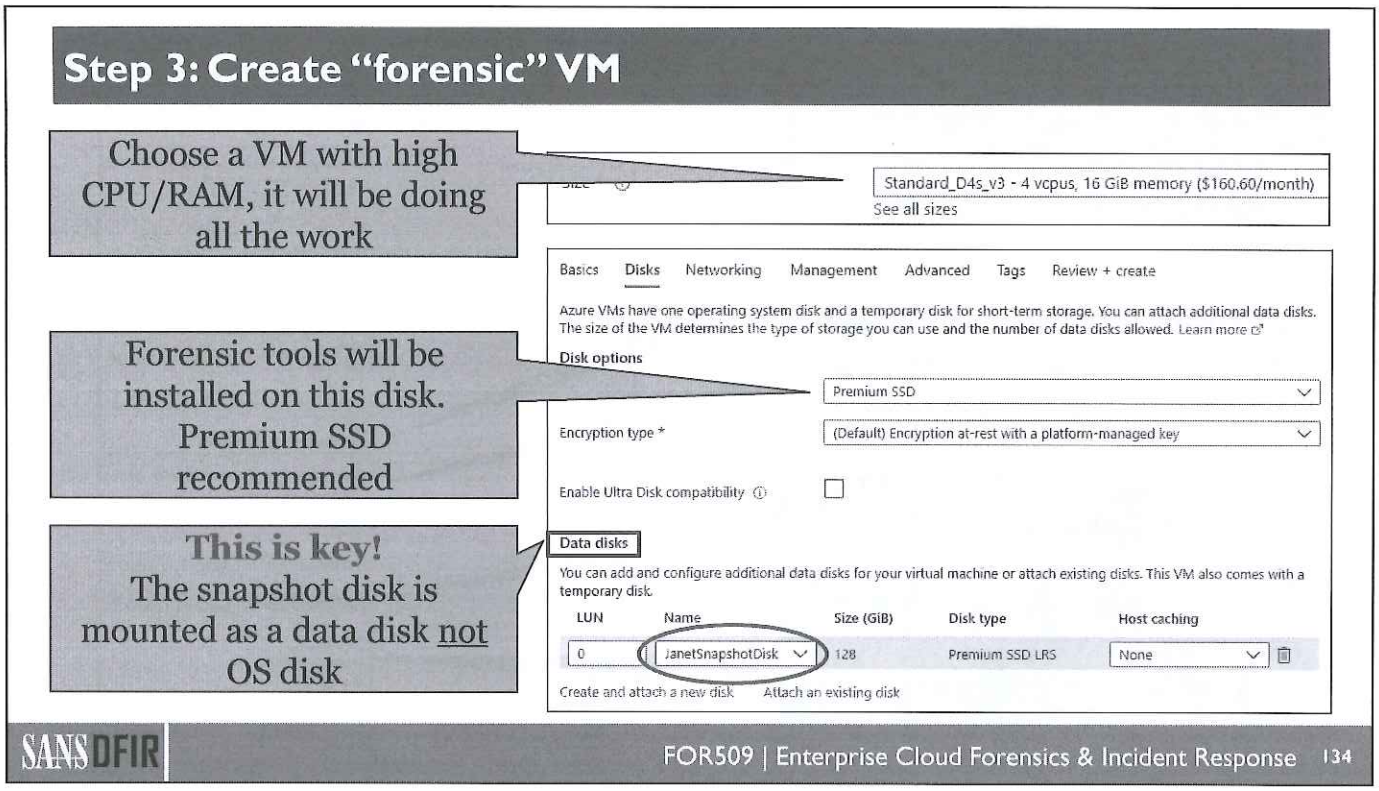


We now need to create a disk that contains the snapshot data. A good practice would be to name this disk the same as the snapshot and add "-disk" at the end. This will prevent getting confused with all the other disks you may have in your subscription.

Instead of creating a blank disk, we are specifying a source type of "Snapshot" as well as the name of the snapshot. This way, the disk will be created with all the snapshot data on it.

For this disk, it's worth getting the Premium SSD as we will be running our forensic tools against it. A fast disk will help you process the data faster.

If you need to save on cost, at this point you could delete the snapshot. We strongly recommend against it, as you may need to repeat this procedure if the data gets corrupted on the disk. Remember that while the snapshot is read-only, this disk isn't.

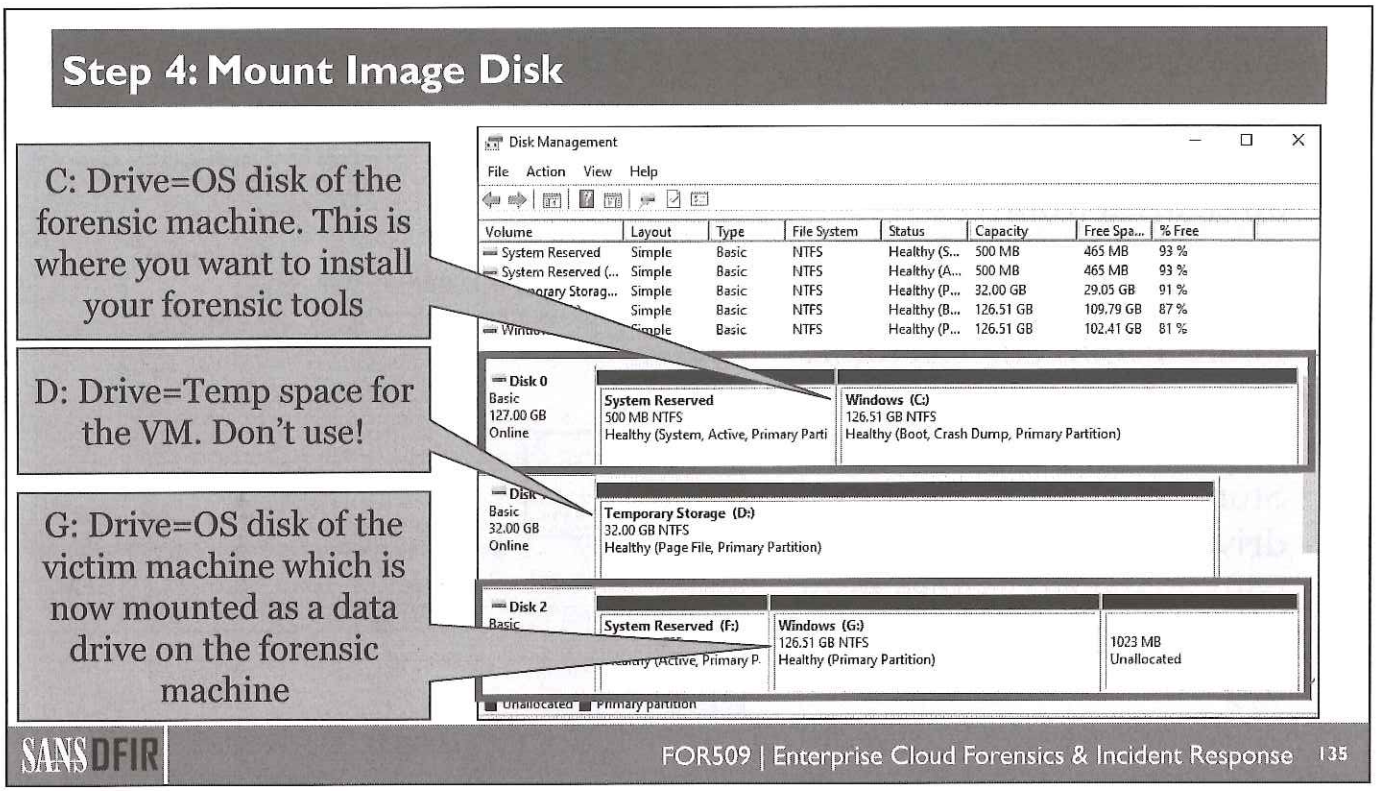


The "forensic VM" is the VM that will do all the work, so it should be created with robust CPU and memory. The exact specifications will depend on the forensic software you choose, but in general 4 vcpus and 16GB of memory should provide plenty of horsepower. Be sure to create this VM in the same region as the snapshot disk from the previous step.

This VM will be created with its own OS disk. This is the disk where you will install your forensic software and store the results. Premium SSD is recommended to optimize the performance of the VM. Azure will provision a 128GB OS disk with over 100GB of free space which should be plenty for your needs.

Here is the critical part: under data disk you will select "attach an existing disk" in order to mount the disk that we created in the previous step. If you forget to specify the data disk during the VM creation, you can add it afterwards. We strongly recommend that you shutdown the VM before attaching a data disk. VMs have been known to get corrupted if you add and remove a data disk while the VM was running.

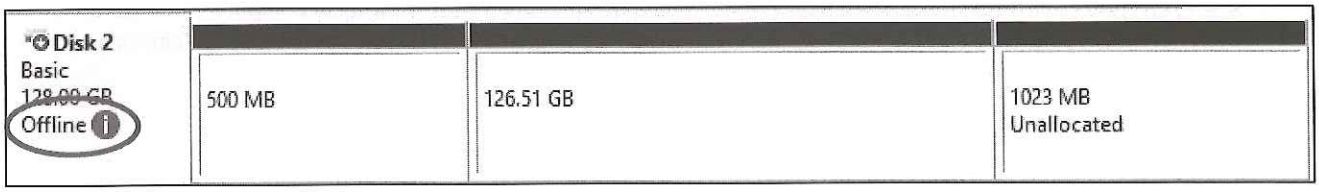
During the disk creation step, you will have an option to create the VM. Don't do it! If you do, the VM will be created with the snapshot disk as its OS disk. In effect you will just be cloning the "victim VM".



Now that the VM is running, you will see 3 disks under Disk Management:

1. Disk 0: the VM's OS disk (C: drive) plus the typical "system reserved" partition for Windows
2. Disk 1: temporary storage for the VM (D: drive). You may use it but it's not a persistent disk so don't put anything important on there
3. Disk 2: a perfect copy of the "victim VM" OS drive. As you would expect it contains the "system reserved" partition plus the OS partition. We are interested in the OS partition which is mounted as the G: drive in this example

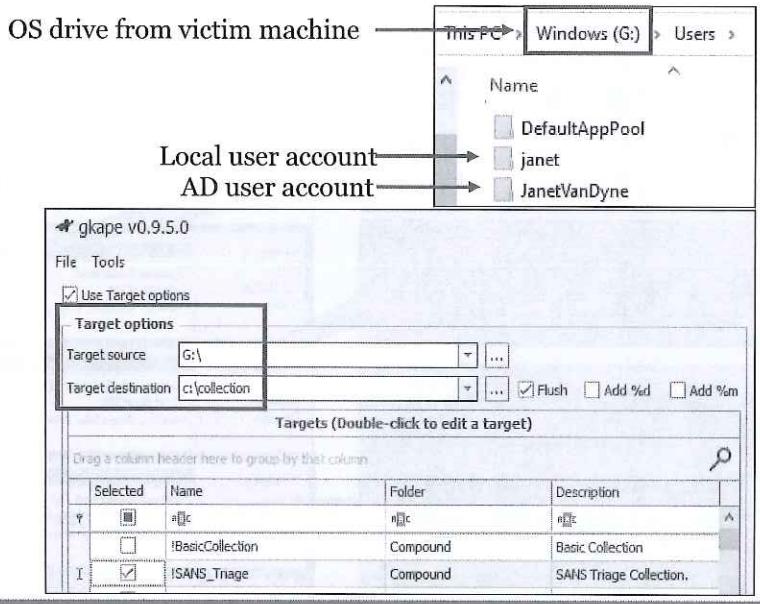
When you first start Disk Management, you will see that Disk 2 is "Offline". Right click on it and select "Online". Windows will then assign a letter to each partition.



Remember that all drives are writable. If you corrupt the G: drive by mistake, you can repeat the process from step 2 since the original snapshot is forensically sound.

Step 5: Run Forensic Tools

- G: drive contains the user directories from JanetDesktop as expected
- Install and run your favorite forensic tools such as Kape.
- Point your forensic tool to the G: drive to collect data
- Store your results on the C: drive so you don't overwrite your evidence (restore from snapshot if you accidently do)



You are now ready to run your favorite forensic tool as taught in FOR500 and FOR508.

From a process standpoint, we would recommend running KAPE[1] with the SANS Triage Collection option to extract key files from the G: drive. You can store these files in the C: drive or the D: drive and process them with Eric Zimmerman's excellent suite of forensic tools.[2]

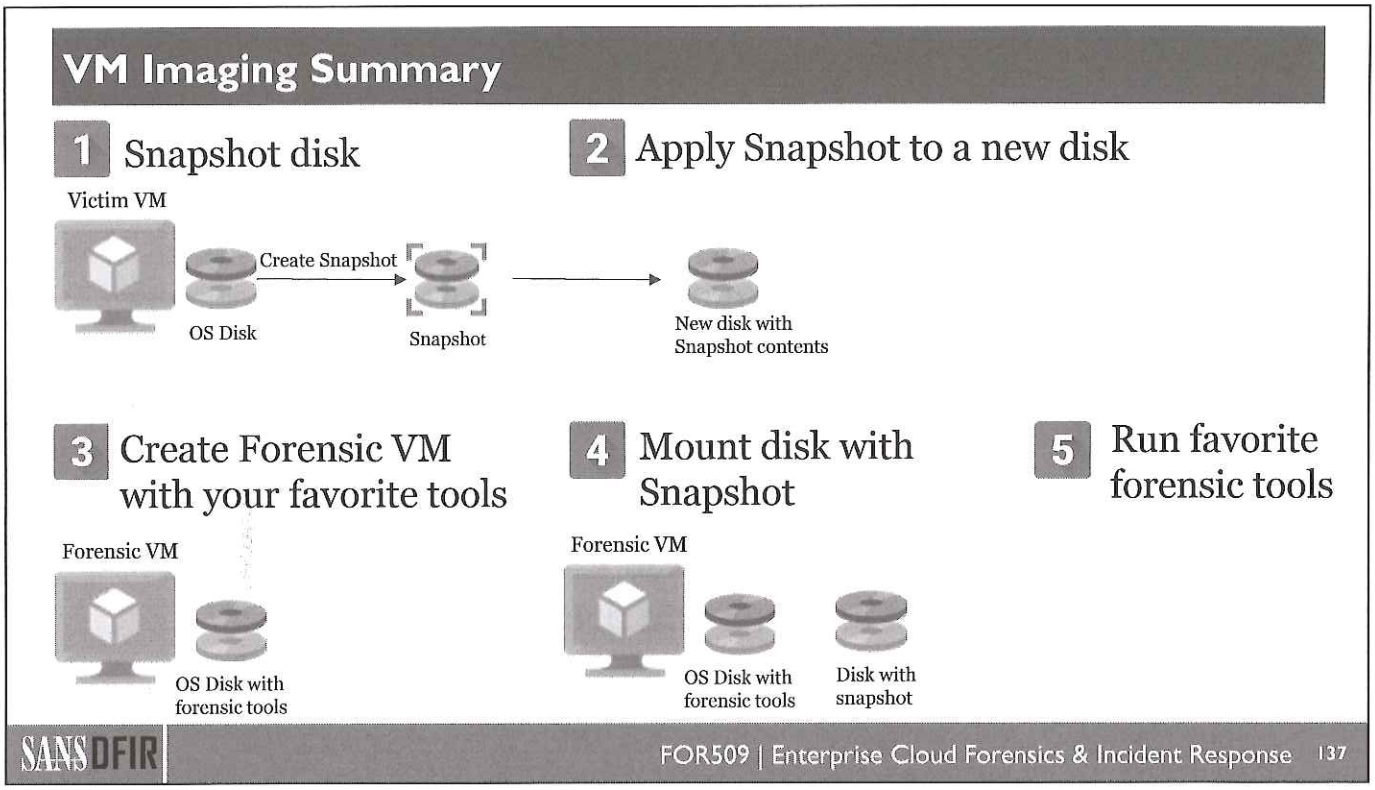
In case you haven't heard of KAPE, it stands for Kroll Artifact Parser and Extractor. KAPE is a triage program that collects the most forensically relevant artifacts from a target. Optionally, it can also parse this data and run analysis programs against it.

Using this process avoids the egress charges of downloading an entire VM to your local computer. You have the option of simply downloading the output for your forensic tools or the data collected by KAPE. This is a much smaller and therefore less costly amount of data to transfer compared to an entire VM.

References:

[1] https://for509.com/kape

[2] https://for509.com/ztools

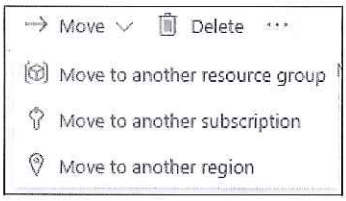


To summarize the process, you will:

1. Create a snapshot of the victim's OS disk. Some investigations may also require you to snapshot the data disk(s).
2. Create a new disk based on the snapshot so that all the information is written to that disk.
3. Create a new VM which we call the Forensic VM with its own OS disk. Be careful not to create this VM based on the snapshot you just created. If you do that, you will simply create a clone of the victim VM. During this step, you should also install your favorite forensic tools on the VM.
4. Mount the disk that contains the snapshot as a data disk on the forensic VM. This step may also be performed during the Forensic VM creation.
5. Run your favorite forensic tools as you have learned in FOR500 and FOR508.

Forensic VM Portability

- You don't need to create a new Forensic VM for each investigation
- Azure has the option to move VMs to another resource group, subscription, or region
- The option is found in the Resource Group that contains the VM you wish to move
- Great idea to create the Forensic VM ahead of time so it's ready as soon as you are tasked with an investigation



You may be wondering if there is a shortcut to creating a forensic VM for every investigation. The good news is YES!

Azure has an option to move a VM to a different resource group, another subscription, or another region. Therefore, once you create your forensic VM and install your favorite tools, you can simply move that VM around as needed. Two recommendations:

1. Make a snapshot of your forensic VM in case it ever gets corrupted
2. Shutdown your VM before attaching and detaching data drives to minimize the risk of corruption

There is another option you may want to consider. It's more complicated and you will need to experiment with it. You could create a blob and copy all your forensic tools to that blob. You would then create a shared access signature which is an external access key for the blob. Using azcopy[1] or Azure Storage Explorer on your forensic VM, you can now access your blob from the forensic VM.

References:

[1] https://for509.com/azcopy

Other Azure Services

Azure has other services that could be helpful in your investigation:

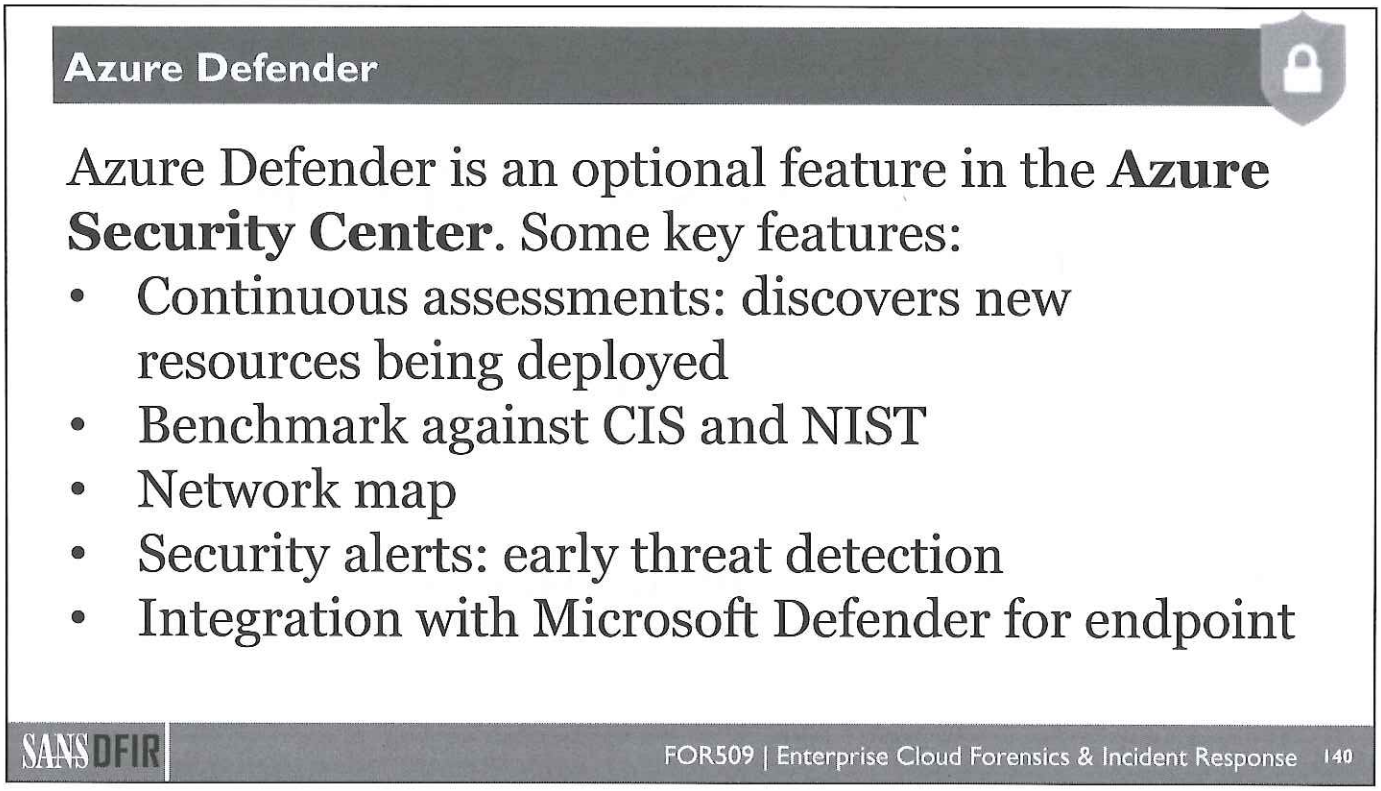
- Azure Defender
- Microsoft Defender for Identity
- Azure Sentinel

These services have additional costs.

Other services offered by Azure might be helpful to you if they have been purchased and implemented by your organization.

This class is about the in-cloud resources you can leverage to perform incident response and forensics. We focused on the features that are included with every tenant. The next few slides will describe optional features so that you are aware of them as well.

Further information can be obtained directly from Microsoft.



Azure Defender is part of the Azure Security Center.[1] By default, Azure includes the Azure Security Center free tier. The free tier provides limited information and benefits for our purposes. On the other hand, Azure Defender[2] includes features which could yield interesting clues during our investigation.

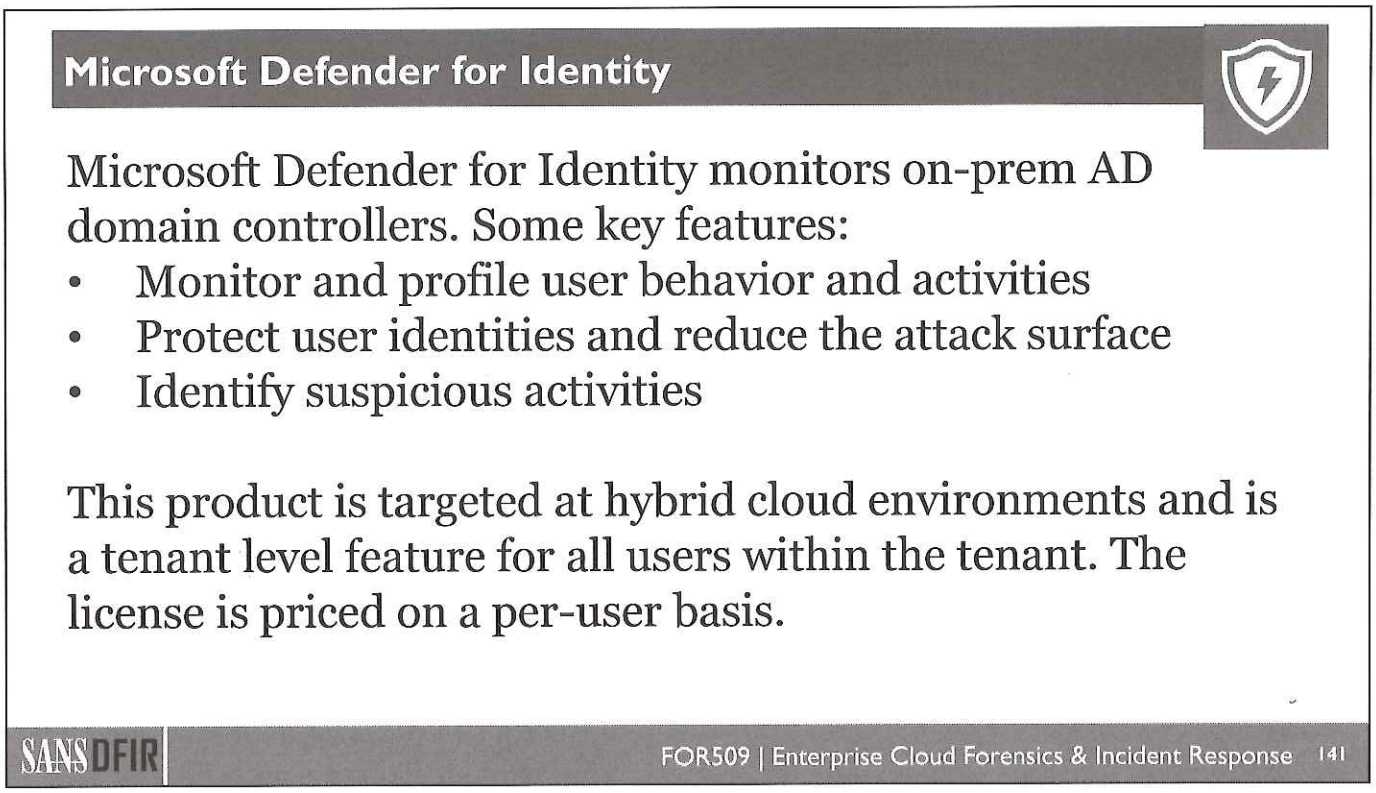
Some of the most interesting features are:

- Continuous assessment that discovers new resources being deployed
- Network map which helps to understand the environment visually
- Security alerts which may provide an early warning to threats in your environment
- Integration with Microsoft Defender for endpoint

References:

[1] https://for509.com/securitycenter

[2] https://for509.com/azuredefender



You may have heard of Azure Advanced Threat Protection (Azure ATP). It's been rebranded as Microsoft Defender for Identity. The goal of this product is identify, detect, and investigate advanced threats, compromised identities, and malicious insider actions.[1] The keys features are:

- Monitor and profile user behavior and activities
- Protect user identities and reduce the attack surface
- Identify suspicious activities

The goal of this product is to monitor your on-prem AD domain controllers and leverage the Azure AD cloud infrastructure to detect potential threats. This product is targeted at hybrid cloud environments.

Microsoft Defender for Identity has its own portal located at `https://portal.atp.azure.com/`

References:

[1] https://for509.com/defenderidentity

Azure Sentinel – SIEM/SOAR

Cloud-native SIEM/SOAR

- Can ingest data from multiple clouds & on-prem infrastructure
- Detects and correlates threats using artificial intelligence
- Built-in orchestration to automate tasks

Sentinel is priced per GB of data ingested, making it a significant investment.

Microsoft Azure Sentinel is a scalable, cloud-native, security information event management (SIEM) and security orchestration automated response (SOAR) solution.[1]

The key features are:

- Cloud-scale SIEM/SOAR. Sentinel is not limited to Azure. It can ingest data from multiple clouds as well as on-prem infrastructure
- Sentinel leverages Microsoft threat intelligence to detect threats
- Sentinel uses artificial intelligence to analyze and correlate threats
- Sentinel includes built-in orchestration to automate common tasks
- Sentinel leverages the MITRE framework to enable you to proactively hunt for threats

Sentinel uses a log analytics workspace and implements numerous queries to hunt for potential threats. Here are some examples of pre-built hunting queries:

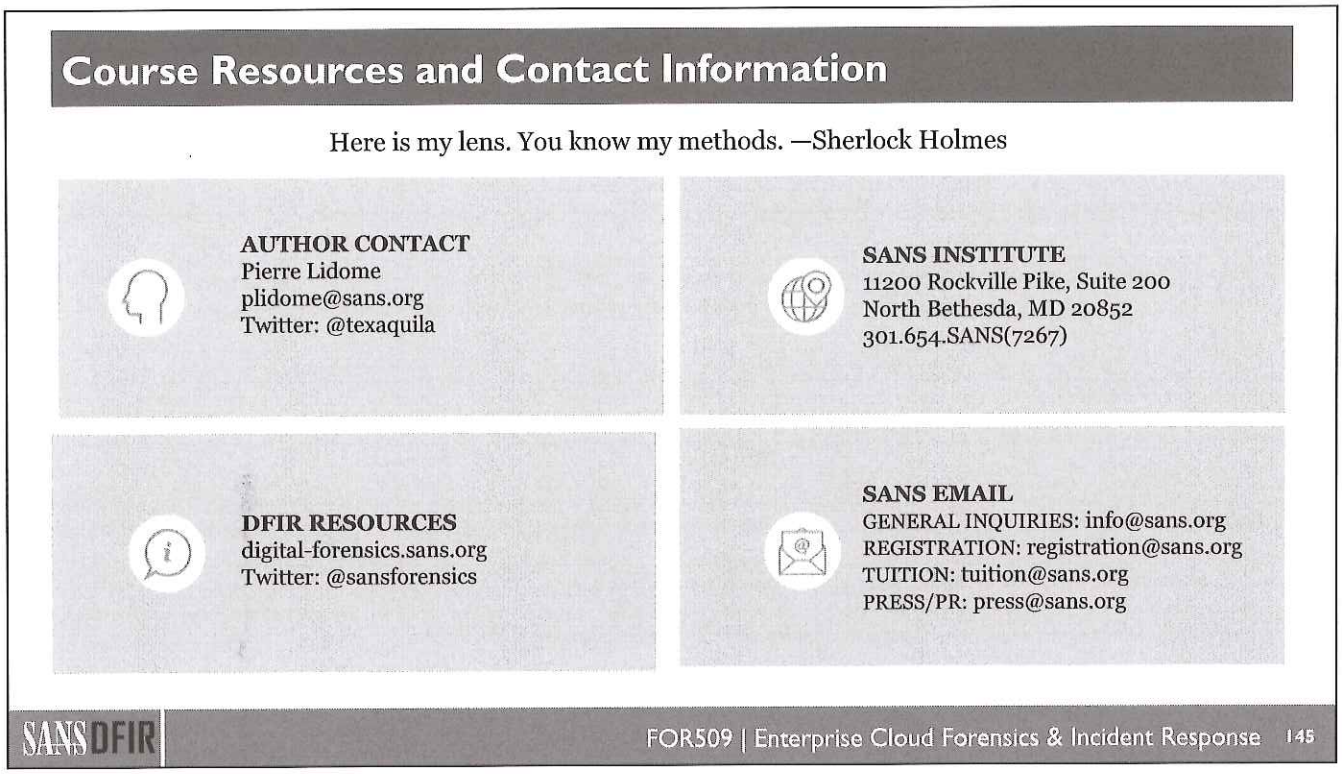
★	Abnormally long DNS URI queries	Microsoft	DnsEvents
★	DNS Domains linked to WannaCry ransomware campai...	Microsoft	DnsEvents
★	Cobalt Strike DNS Beaconing	Microsoft	DnsEvents +1 ⓘ
★	Failed service logon attempt by user account with avail...	Microsoft	AuditLogs +1 ⓘ
★	Failed Login Attempt by Expired account	Microsoft	SecurityEvent +1 ⓘ
★	Multiple Password Reset by user	Microsoft	AuditLogs +4 ⓘ

Sentinel is priced per GB of data ingested. The pricing depends on the region. A large infrastructure will likely generate TB of data per day, making a tool like Sentinel a significant investment.

References:

[1] https://for509.com/sentinel

This page intentionally left blank.



Author: Pierre Lidome

Email: plidome@sans.org

Twitter: @texaquila

"As usual, SANS courses pay for themselves by Day 2. By Day 3, you are itching to get back to the office to use what you've learned."
Ken Evans, Hewlett Packard Enterprise - Digital Investigation Services

SANS Programs
sans.org/programs

GIAC Certifications
Graduate Degree Programs
NetWars & CyberCity Ranges
Cyber Guardian
Security Awareness Training
CyberTalent Management
Group/Enterprise Purchase Arrangements
DoDD 8140
Community of Interest for NetSec
Cybersecurity Innovation Awards

Search SANSInstitute

SANS Free Resources
sans.org/security-resources

- E-Newsletters
  NewsBites: Bi-weekly digest of top news
  OUCH!: Monthly security awareness newsletter
  @RISK: Weekly summary of threats & mitigations
- Internet Storm Center
- CIS Critical Security Controls
- Blogs
- Security Posters
- Webcasts
- InfoSec Reading Room
- Top 25 Software Errors
- Security Policies
- Intrusion Detection FAQ
- Tip of the Day
- 20 Coolest Careers
- Security Glossary

SANS Institute
8120 Woodmont Avenue | Suite 310
Bethesda, MD 20814
301.654.SANS(7267)
info@sans.org