**Unit 13 README: ELK Stack Project**

**Unit Description**

In the first project week, you will configure an ELK stack server in order to set up a cloud monitoring system. This project will result in tangible deliverables that demonstrate your knowledge of cloud, network security, logging and monitoring.

**Unit Objectives**

Click here to view the daily unit objectives.

**Lab Environment**

For the majority of demonstrations and activities, the class will use Microsoft Azure cloud services and the Azure cloud portal.

* You will **not** be using any of the Azure lab environments. Instead, you will be using personal Azure accounts.

**What to Be Aware Of:**

* The sample logs used in this unit are specific to the time in which they are viewed. As such, answers will vary from the answers provided in the solutions.
* The VM for the ELK server **must** have at least 4GiB of memory for the ELK container to run properly. Azure has VM options that have 3.5 GiB of memory, but **do not use them**. They will not properly run the ELK container because they do not have enough memory.
* Azure may run out of available VMs for you to create a particular region. If this happens, you will need to do one of two things:
  + Open a support ticket with Azure support using [these instructions](https://docs.microsoft.com/en-us/azure/azure-portal/supportability/how-to-create-azure-support-request). Azure support is generally very quick to resolve issues.
  + Create another vNet in another region and attempt to create the ELK sever in that region.

**Security+ Domains**

This unit covers portions of the following domains on the Security+ exam:

 Click here to view Security+ Domains that apply to this project.

For more information about these Security+ domains, refer to the following resource: [Security+ Exam Objectives](https://www.comptia.jp/pdf/Security%2B%20SY0-501%20Exam%20Objectives.pdf)

**Additional Reading and Resources**

 Click here to view additional reading materials and resources.

**Unit 13 : Homework**

This unit's homework assignment can be viewed here:

* [Unit 13 Homework File](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/2-Homework/13-Github-Fundamentals/Unsolved/README.md)

**Looking Forward**

The next unit will mark the start of the Offensive Security module. In the Web Development unit, we'll examine the infrastructure and deployment of web applications to inform how we can explore and exploit vulnerabilities within those applications.

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Day 1 Activity File: ELK Installation**

Today, you will configure an ELK server within your virtual network. Specifically, you will:

1. **Create a new vNet** in a new region, within your resource group.
2. **Create a Peer Network Connection** between your two vNets.
3. **Create a new VM.** Deploy a new VM into the new vNet with it's own Security Group. This VM will host the ELK server.
4. **Download and configure a container.** Download and configure the elk-docker container onto this new VM.
5. **Launch and expose the container.** Launch the elk-docker container to start the ELK server.
6. **Implement identity and access management.** Configure your new Security group so you can connect to ELK via HTTP, and view it through the browser.

**Resources**

* [elk-docker Container Documentation](https://elk-docker.readthedocs.io/)
* [Elastic.co: The Elastic Stack](https://www.elastic.co/elastic-stack)
* [Ansible Documentation](https://docs.ansible.com/ansible/latest/modules/modules_by_category.html)
* [elk-docker Image Documentation](https://elk-docker.readthedocs.io/#elasticsearch-logstash-kibana-elk-docker-image-documentation)
* [Virtual Memory Documentation](https://www.elastic.co/guide/en/elasticsearch/reference/5.0/vm-max-map-count.html#vm-max-map-count)
* ELK Server URL: http://[your.IP]:5601
* [Docker Commands Cheatsheet](https://phoenixnap.com/kb/list-of-docker-commands-cheat-sheet)

**Azure Documentation:**

* Azure's page on peer networks [HERE](https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview)
* Peer networking in Azure How-To: [Global vNet Peering](https://azure.microsoft.com/en-ca/blog/global-vnet-peering-now-generally-available/)
* If Microsoft Support is needed: [How to open a support ticket](https://docs.microsoft.com/en-us/azure/azure-portal/supportability/how-to-create-azure-support-request)

**Template configuration files:**

It is recommended that you use these templates, but you can build your own if you'd like an additional challenge.

* [hosts.yml](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_1/Unsolved/Resources/hosts.yml)
* [install-elk.yml](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_1/Unsolved/Resources/install-elk.yml)

To ensure that these configuration templates do not change when copying them to your Ansible Container, use curl to get them onto your container. Using standard copy and paste is known to corrupt/change characters inside the configuration file and can cause them not to work.

**Getting Started**

Before you begin, it is recommended that you create a high-level diagram of your intended new setup. You can use a tool like [**draw.io**](http://draw.io/), or a pen and paper.

* Write down each of the machine's functions (jump box, Ansible, ELK server).
* As you complete the steps below, document the IP address associated with each machine.

**Note:** You can build off of this diagram for the Day 3 network diagram activity.

We also recommend that you organize your folders in your Ansible machine as you see below:

etc # should already exist.

----> ansible # should already exist.

---------> ansible.cfg # should already exist.

---------> hosts # should already exist.

---------> roles # this is a new directory that will contain your ansible playbooks.

--------------> files # this is a new directory that will contain the configuration files we pass to you.

**Instructions**

**Part 1: Creating a New vNet**

Make sure that you are logged into your personal Azure account, where your cloud security unit VMs are located.

1. Create a new vNet located in the same resource group you have been using.
   * Make sure this vNet is located in a *new* region and not the same region as your other VM's.
   * Leave the rest of the settings at default.
     + Notice that the IP Addressing has automatically created a new network space of 10.1.0.0/16 which is exactly what you want.
2. Create a peer connection between your vNets. This will allow traffic to pass between your vNets and regions. This peer connection will make both a connection from your first vNet to your Second vNet *and* a reverse connection from your second vNet back to your first vNet. This will allow traffic to pass in both directions.
   * Navigate to 'Virtual Network' in the Azure Portal.
   * Select your new vNet to view it's details.
   * Under 'Settings' on the left side, select 'Peerings'.
   * Click the + Add button to create a new Peering.
   * Make sure your new Peering has the following settings:
     + A unique name of the connection from your new vNet to your old vNet.
       - Elk-to-Red would make sense
     + Choose your original RedTeam vNet in the dropdown labeled 'Virtual Network'. This is the network you are connecting to your new vNet and you should only have one option.
     + Name the resulting connection from your RedTeam Vnet to your Elk vNet.
       - Red-to-Elk would make sense
   * Leave all other settings at their defaults.

**Part 2: Creating a New VM**

1. Create a new Ubuntu VM in your virtual network with the following configurations:
   * **RAM**: 4 GB+
     + **Important:** The VM for the ELK server ***MUST*** have at least 4GiB of memory for the ELK container to run properly. Azure has VM options that have 3.5 GiB of memory, but *DO NOT USE THEM.* They will not properly run the ELK container because they do not have enough memory.
     + A few specific machines that will work are:
       - Standard D2s v3 (2 vcpus, 8GiB memory)
       - Standard B2s (2vcpus, 4GiB memory
     + If a VM that has 4GiB of memory is not available, the ELK VM will need to be deployed in a different region that has a VM with 4GiB available OR you will need to open a support ticket with Microsoft (they often resolve issues very fast). See instructions for opening a ticket [HERE](https://docs.microsoft.com/en-us/azure/azure-portal/supportability/how-to-create-azure-support-request)
   * **IP Address**: The VM must have a public IP address.
   * **Networking**: The VM must be added to the new region in which you created your new vNet. You want to make sure you select your new vNEt and allow a new *basic* Security Group to be created for this VM.
   * **Access**: The VM must use the same SSH keys as your WebVM's. This should be the ssh keys that were created on the *Ansible container* that's running on your jump box.
     + Open a terminal on your computer and SSH into the jump box.
     + From the jump box login shell, run the required Docker commands to start and attach to your Ansible container.
     + Use cat to retrieve your *public* ssh key (~/.ssh/id\_rsa.pub)
     + As a reminder, your SSH key should not have a password on it because the password may cause Ansible errors. You can remove a password from your SSH key with: ssh-keygen -p -f ~/.ssh/id\_rsa.
2. After creating the new VM in Azure, verify that it works as expected by connecting via SSH from the *Ansible container* on your jump box VM.
   * From the Ansible container shell, SSH into the new VM using it's internal IP.
   * Note that the new VM should be on a new subnet e.g 10.1.0.0/24 instead of 10.0.0.0/24
     + Your subnet may vary, but if you used the default network settings your networks and subnets should fall into the default pattern of 10.0.0.0, 10.1.0.0, 10.2.0.0 etc.

If the connection succeeds, you are ready to move on to the next step. If not, verify that you used the correct SSH key (from inside the Ansible container). If the problem persists, you will need to troubleshoot further.

**Some troubleshooting theory:**

Running into issues or errors? Try the following troubleshooting techniques:

* Change one thing and retest. If you change several things before you re-test, you will not know if any one of those things has helped the situation or made it worse.
* As per the split-half search, remove *half* of your variables that could be going wrong and re-test. If the issue is resolved, you know that your problem resides in the variables that you removed. If the problem is still present you know your problem resides in the variables that you did not remove.
  + Now take the set of variables where you know the problem resides. Remove half of them again and retest. Repeat this process until you find the problem.
* In this context, using the split-half search could mean:
  + Verifying that the VM is running in Azure. This removes your Jump Box from the equation and you can determine if their is a problem with the VM itself.
  + Use nmap or ping from your Jump-Box to the new VM to determine if you can make a connection. This removes SSH and the Ansible container from the equation. If you can't make a connection from your jump-box at all, you know the issue is *not* with SSH or the Ansible container.
  + Remove your Deny All Security Group rule to allow *all* traffic. This removes the Security Group from the equation. If you still cannot make a connection, you know the issue is *not* with the Security Group.
  + Verify that you are attached to the Ansible container by running whoami or examining the command prompt:
    - Jump-box command prompt: sysadmin@Jump-Box-Provisioner
    - Ansible container command prompt: root@6160a9be360e

Still having trouble? Ask a classmate, instructor or TA for help!

**Part 3: Downloading and Configuring the Container**

| **:warning: Checkpoint :warning:** |
| --- |
| Before continuing, make sure you have completed the following critical tasks. |
| :heavy\_check\_mark: A new vNet has been created in your resource group in a new region. |
| :heavy\_check\_mark: The new vNet is located in a region other than the region you have used for all your other resources. |
| :heavy\_check\_mark: A peer connection has been created between vNets, allowing traffic to pass between them. |
| :heavy\_check\_mark: A new VM has been created with a minimum of 4GB of memory. (8GB is preferred.) |
| :heavy\_check\_mark: The new VM is configured to use the same SSH key that your Web VMs use from the Ansible container. |
| :heavy\_check\_mark: Ansible is able to connect to the new VM. |

1. Using Ansible, configure the newly created VM.
   * From your Ansible container, add the new VM to Ansible's hosts file.
   * Create a playbook that installs Docker and configures the container.
   * Run the playbook to launch the container.
2. You just created a new VM that you will use to run your ELK stack. In order to use Ansible to configure this machine, you must add it to the list of machines Ansible can discover and connect to.
   * Don't forget to specify python3 with ansible\_python\_interpreter=/usr/bin/python3
   * This list is Ansible's **inventory** and is stored in the hosts text file:
   * # /etc/ansible/hosts
   * [webservers]
   * 10.0.0.4 ansible\_python\_interpreter=/usr/bin/python3
   * 10.0.0.5 ansible\_python\_interpreter=/usr/bin/python3
   * 10.0.0.6 ansible\_python\_interpreter=/usr/bin/python3
   * [elk]
   * 10.1.0.4 ansible\_python\_interpreter=/usr/bin/python3
   * [webservers] and [elk] are **groups**. When you run playbooks with Ansible, you specify which group to run them on. This allows you to run certain playbooks on some machines, but not on others.
3. You should already be in your Ansible VM. Add an [elk] group to your Ansible VM's hosts file by following the steps below on the command line:
   * Edit the inventory file nano /etc/ansible/hosts.
   * Add a group called [elk] and specify the IP address of the VM you just created in Azure.
   * **Note:** If you get stuck, consult the starter hosts file:
     + [Starter Host File](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_1/Unsolved/Resources/hosts.yml)
4. Once you've created the [elk] group, you'll create a playbook to configure it.
   * Today, you'll write a play to configure the ELK server. This play should only run on the VM in the [elk] group. Recall that you can use the hosts option in Ansible to specify which machines to run a set of tasks against.
   * - hosts: elk
   * - become: True
   * - tasks:
   * - name: Install Packages
   * *# Etc...*
   * Here, the hosts option specifies that these tasks should only be run on the machines in the elk group.
   * To create this playbook, continue using your terminal to complete the following steps:
     + Verify you are in your Ansible container.
     + Create a new playbook: touch /etc/ansible/install-elk.yml.
     + Ensure that the header of the playbook looks like the YAML snippet above. Specifically, it must specify elk as the target hosts.
   * Write tasks that do the following:
     + Set the vm.max\_map\_count to 262144
     + This configures the target VM (the machine being configured) to use more memory. The ELK container will not run without this setting.
       - You will want to use Ansible's sysctl module and configure it so that this setting is automatically run if your VM has been restarted.
       - The most common reason that the ELK container does not run, is caused by this setting being incorrect.
       - [Ansible sysctl](https://docs.ansible.com/ansible/latest/modules/sysctl_module.html)
     + Installs the following apt packages:
       - docker.io: The Docker engine, used for running containers.
       - python3-pip: Package used to install Python software.
     + Installs the following pip packages:
       - docker: Python client for Docker. Required by Ansbile to control the state of Docker containers.
     + Downloads the Docker container called sebp/elk:761. sebp is the organization that made the container. elk is the container and 761 is the version.
     + Configures the container to start with the following port mappings:
       - 5601:5601
       - 9200:9200
       - 5044:5044
       - **Hint:** Use the Ansible module docker-container along with published port mappings. [More info at Ansible.com](https://docs.ansible.com/ansible/latest/modules/docker_container_module.html#examples).
     + Starts the container.
     + Enables the docker service on boot, so that if you restart your ELK VM, the docker service start up automatically.
       - **Hint:** Use the Ansible module systemd to make sure the docker service is running. [More info at Ansible.com](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/systemd_module.html).
   * Refer to the documentation on Ansible's [docker\_container](https://docs.ansible.com/ansible/latest/modules/docker_container_module.html" \o "https://docs.ansible.com/ansible/latest/modules/docker_container_module.html) and [docker\_image](https://docs.ansible.com/ansible/latest/modules/docker_image_module.html" \o "https://docs.ansible.com/ansible/latest/modules/docker_image_module.html) modules for guidance.

**Part 4: Launching and Exposing the Container**

Check your playbook for typos and other errors, then run it.

1. After the playbook completes, you should still be in the Ansible container. From there, use the command line to SSH into the ELK server and ensure that the sebp/elk:761 container is running by running: docker ps.
   * You should see a single row whose second column is sebp/elk. Take a screenshot before proceeding to the next step. 761 should show under Tag
   * **Note**: If you're stuck, refer to the starter elk-playbook.yaml file provided: [Starter elk-playbook File](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_1/Unsolved/Resources/install-elk.yml).

**Troubleshooting**

Is your container failing to start? Try troubleshooting the following:

* Common issues include:
  + The ELK container quits or doesn't launch
  + The Ansible script fails

Using the split-half search to complete the following:

* In this context, removing half of your variables could mean:
  + Logging into the Elk server and implementing the steps from your Ansible script manually using the equivalent Bash commands.
    - This removes your Ansible script from the equation and you can determine if the problem is with your Ansible Script, OR the problem is on the ELK Server.
    - You can manually launch the ELK container with: sudo docker start elk OR sudo docker run -p 5601:5601 -p 9200:9200 -p 5044:5044 -it --name elk sebp/elk:761
  + Split-half could mean downloading and running a different container on the ELK server or trying different settings in your Ansible script.
    - This removes the ELK container from the equation and you can determine if the issue may be with Docker or it may be with the ELK container.
    - Docker has a simple container to verify that docker is running correctly. Just use the command: docker run hello-world
    - If you try different settings in your Ansible script, remember to change one thing at a time. If it doesn't help, you may not need it and can change it back.
  + Split-half might mean removing half of the commands of your Ansible script (or just comment them out) OR, remove the specific command that is failing.
    - This removes half of the commands you are trying to run and you can see which part of the script is failing.
    - You can also remove all the other commands that are working, and only run the command that is failing to save time.

The most common reason for the container failing to launch is a limit on map counts. Follow [these instructions from Elastic.co](https://www.elastic.co/guide/en/elasticsearch/reference/5.0/vm-max-map-count.html#vm-max-map-count) to verify you meet the requirement.

* Review the [official ELK stack documentation](https://elk-docker.readthedocs.io/#prerequisites).
* Ask for help!

**Part 5: Identity and Access Management**

| **:warning: Checkpoint :warning:** |
| --- |
| Before continuing, make sure you have completed the following critical tasks. |
| :heavy\_check\_mark: An Ansible playbook has been created that installs and configures an ELK container. |
| :heavy\_check\_mark: The Ansible playbook can be run on the new VM. |
| :heavy\_check\_mark: The new VM is running the ELK container. |

1. The final step is to restrict access to the ELK VM using Azure's network security groups (NSGs). You need to add your public IP address to a whitelist, just as you did when clearing access to your jump box.

**Note:** Make sure you are on your Azure account for this step.

* Recall that the ELK stack's web server runs on port 5601. Open your virtual network's existing NSG and create an incoming rule for your security group that allows TCP traffic over port 5601 from your public IP address.
  + **Note:** If you finish this step in a classroom, you will need to repeat this step at home to connect to Kibana from your personal network.
* Verify that you can access your server by navigating to http://[your.ELK-VM.External.IP]:5601/app/kibana. Use the public IP address of your new VM.
* You should see this webpage:

If this is what you see, congratulations! Take a screenshot of your dashboard before proceeding.

**Troubleshooting**

Running into issues or errors? Use the split-half search!

* In this context, split-half could mean logging into the ELK VM and running curl localhost:5601/app/kibana from the command line.
  + This removes your browser and the security group from your equation and you can see if this command returns any HTML. If it does, you know that the ELK VM is working! The problem is likely with your connection through the browser. No HTML means that there is a problem with the ELK container.
* If it's not the container, think about the variables that could affect your connection to a website from your browser:
  + Your home Network Firewall
  + Your Azure Security group
  + The browser itself

Other troubleshooting issues:

* Try another browser. This removes the entire browser from the picture in case of any cache issues, plugin conflicts or other issues.
* If it's not the browser, try allowing *all* traffic in your Azure Security group. This removes the security group from the equation.
* If it's not the security group, could it be your home's firewall or your IP address has changed? You can remove this variable by switching to your phone's hotspot, or trying from a friend's network.
  + If you switch networks, remember to update your security group rules to allow traffic from your new IP address.

**Example Troubleshooting Scenario:**

Suppose the following:

* Ansbile Script has completed without errors.
* SSH from the Ansible Jump box to the ELK VM works.
* sudo docker ps shows that the ELK container is running with the correct ports.
* curl http://localhost:5601/app/kibana *does* return HTML.
* Navigating to http://<ELK.VM.External.IP>:5601/app/kibana from a browser does not load a webpage.

Using the split-haf search:

* Ansible ran just fine and the ELK container is running a webpage so these variables can be removed.
  + The variables that are left are:
    - The Local network and firewall
    - The Firefox Browser
    - The Azure Security Group
* We have 3 variables left. We can't split them in half but rather will try one at a time, starting with the Local Network.
  + Other webpages are loading without an issue, so the local network is not likely the problem.
* We have 2 variables left.
  + After trying the Google Chrome browser it also does not load the page. This suggests that the issue is *not* with the browser.
* At this point we know the issue is most likely with the Security Group.
  + Check if ping is working from the command line of the local computer to the ELK VM. This removes the browser again and removes the HTTP protocol from the equation.
  + ping is not working.
  + Use nmap -p 80 <elk.vm.external.ip> to check if port 80 is allowing traffic.
  + nmap returns that port 80 is filtered.
* Now we know that the Security Group is blocking not only ICMP but also HTTP protocols. The security group appears to be the problem.
  + Check the Security Group settings and create a new rule that allows any port and any protocol from the internet to the Virtual Network.
    - Note that this setting is *highly* insecure and should only be done for testing. This removes your public IP setting from the SG rule and checks if any computer can access the VM.
  + Check if ping is working from the command line once more.
    - ping is now working.
  + Attempt to Navigate to http://<ELK.VM.External.IP>:5601/app/kibana once more.
    - The website is now loading.
* We have identified that the problem is somewhere in the Security Group rules! How can we determine what rule is the problem? Split-Half again!
  + At this point you may know exactly what rule is the problem, or you might have a few rules that could be the issue.
  + Start with one rule at a time and modify them until you find the setting that is causing the problem.
  + You can even split-half all of the settings that make up each rule.
  + Continue with this process until you have a working setup.

If the Ansible playbook isn't running, try connecting to the ELK container manually using SSH and run the appropriate commands to expand the vm.max\_map\_count and start the docker container.

Still having trouble?

Ask for help! Work with your classmates, instructor and TA's to find a solution.

**Day 1 Milestone**

In today's class, you:

* Deployed a new VM on your virtual network.
* Created an Ansible play to install and configure an ELK instance.
* Restricted access to the new server.

Completing these steps required you to leverage your systems administration, virtualization, cloud, and automation skills. This is an impressive set of tools to have in your toolkit!

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Day 2 Activity File: Filebeat Installation**

Now that we have our ELK monitoring server up installed and configured, we're going to add another tool called **Filebeat**.

Taking raw log files and trying to make sense of all the data is often difficult and time consuming. We can use Filebeat to collect, parse, and visualize ELK logs in a single command. This will help us better track our organizational goals.

By the end of class today, you should complete the following steps:

1. **Install Filebeat on the Web VM's**. Verify your ELK server container is up and running, and install Filebeat on your WebVM's.
2. **Create the Filebeat configuration file**. Create and edit the Filebeat configuration file for your DVWA VMs.
3. **Create the Filebeat installation play**. Create another Ansible playbook that accomplishes the tasks required to install Filebeat.
4. **Verify the installation and playbook**. Confirm that your installation and playbook worked by verifying that the ELK stack is receiving logs.

**Resources**

Below are links to the Filebeat and Docker documentation. It is strongly suggested that you read through these before starting the activity:

* [Filebeat Container Documentation](https://www.elastic.co/beats/filebeat)
* [Docker Commands Cheat Sheet](https://phoenixnap.com/kb/list-of-docker-commands-cheat-sheet)

You can also use the following resources if you get stuck:

* [Docker and Ansible Cloud Week Cheat Sheet](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_2/12-Cloud-Security/CheatSheet.md)
* [Ansible Roles](https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html)

**Getting Started**

Today, you will continue building up your ELK server. Specifically, if you have completed the installation step, you will proceed to install **Filebeat**.

* Recall that Filebeat helps generate and organize log files to send to Logstash and Elasticsearch. Specifically, it logs information about the file system, including which files have changed and when.
* Filebeat is often used to collect log files from very specific files, such as logs generated by Apache, Microsoft Azure tools, the Nginx web server, or MySQL databases. Today you will be using it to monitor the Apache server and MySQL database logs generated by DVWA.
* Since Filebeat is built to collect data about specific files on remote machines, it must be installed on the VMs you want to monitor. You will install Filebeat on the DVWA container you created during the cloud security week. This will provide you with a rich source of logs after you complete your deployment.

**Instructions**

**Part 1: Installing Filebeat on the DVWA Container**

1. First, make sure that the ELK server container is up and running:
   * Navigate to http://[your.VM.IP]:5601/app/kibana. Use the public IP address of the ELK server that you created.
   * Click 'Explore on my Own'
   * If you do not see the Kibana server landing page, open a terminal on your computer and SSH into the ELK server.
     + Run docker container list -a to verify that the container is on.
     + If it isn't, run sudo docker start elk.
2. Use the ELK server's GUI to begin installing Filebeat on your DVWA VM.
   * Navigate to your ELK server's IP address:
     + Click **Add Log Data**.
     + Choose **System Logs**.
     + Click on the **DEB** tab under **Getting Started**.
   * Here you will find the most up-to-date Filebeat installation instructions for Linux.
   * Note that you do not need to do anything on this page. Since Filebeat is open source, it is updated frequently. Therefore, specific details around installation can change. This site will always have the most up-to-date instructions.

**Part 2: Creating the Filebeat Configuration File**

1. Next, we will create a Filebeat configuration file, after which we will create the Ansible playbook file.
   * At that point, we will translate the instructions in the DEB tab into a new Ansible play, which you will use to automatically install Filebeat on your DVWA machines.
     + Translating installation instructions to reusable playbooks is a common task for modern infrastructure teams. Being able to explain the value of this task and the plays you've created will be valuable in job interviews.
     + Creating this play will allow you to easily install Filebeat on any machine you want to monitor later, whether for class, work, or a personal project.
   * Open a terminal and SSH into your jump box:
     + Start the Ansible container.
     + Use the correct Docker command to attach to your Ansible container.
   * As mentioned earlier, the Filebeat installation instructions require you to create a Filebeat configuration file.
     + You will need to edit this file so that it has the correct settings to work with your ELK server.

* You can use the provided template for the Filebeat configuration file:
  + [Filebeat Configuration File Template](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_2/Unsolved/Activities/Stu_Day_2/Unsolved/Resources/filebeat-config.yml).
    - Note that when text is copy and pasted from the web into your terminal, formatting differences are likely to occur that will corrupt this configuration file.
* Using curl is a better way to avoid errors and we have the file hosted for public download [HERE](https://gist.githubusercontent.com/slape/5cc350109583af6cbe577bbcc0710c93/raw/eca603b72586fbe148c11f9c87bf96a63cb25760/Filebeat)
  + Run: curl https://gist.githubusercontent.com/slape/5cc350109583af6cbe577bbcc0710c93/raw/eca603b72586fbe148c11f9c87bf96a63cb25760/Filebeat > /etc/ansible/filebeat-config.yml
* root@6160a9be360e:/etc/ansible*# curl https://gist.githubusercontent.com/slape/5cc350109583af6cbe577bbcc0710c93/raw/eca603b72586fbe148c11f9c87bf96a63cb25760/Filebeat > filebeat-config.yml*
* % Total % Received % Xferd Average Speed Time Time Time Current
* Dload Upload Total Spent Left Speed
* 100 73112 100 73112 0 0 964k 0 --:--:-- --:--:-- --:--:-- 964k

1. Once you have this file on your Ansible container, edit it as specified:
   * The username is elastic and the password is changeme.
   * Scroll to line #1106 and replace the IP address with the IP address of your ELK machine.
   * output.elasticsearch:
   * hosts: ["10.1.0.4:9200"]
   * username: "elastic"
   * password: "changeme"
   * Scroll to line #1806 and replace the IP address with the IP address of your ELK machine.
   * setup.kibana:
   * host: "10.1.0.4:5601"
   * Note that the default credentials are elastic:changeme and should not be changed at this step.
   * Save this file in /etc/ansible/files/filebeat-config.yml.

**Part 3: Creating the Filebeat Installation Play**

1. Next, create a new playbook that installs Filebeat and then copies the Filebeat configuration file you just made to the correct location.
   * On the Ansible VM, create a playbook file, filebeat-playbook.yml.
     + Locate this file in your /etc/ansible/roles/ directory.
   * Open your playbook and implement the following tasks:
     + Download the .deb file from [artifacts.elastic.co](https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-amd64.deb).
     + Install the .deb file using the dpkg command shown below:
       - dpkg -i filebeat-7.4.0-amd64.deb
     + Copy the Filebeat configuration file from your Ansible container to your WebVM's where you just installed Filebeat. Make sure it is copied to: /etc/filebeat/filebeat.yml
       - Use Ansible's copy module to copy the entire configuration file to the correct place.
     + Run the following commands:
       - filebeat modules enable system
       - filebeat setup
       - service filebeat start
     + Enable the filebeat service on boot.
       - **Hint:** Use the Ansible module systemd to make sure the filebeat service is running. [More info at Ansible.com](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/systemd_module.html).
   * You may find the following hints and links helpful:
     + This play should only run on the web machines that are running the DVWA containers.
     + Refer to the [Ansible playbook documentation](https://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html#playbooks-intro) if needed.
     + Use the Ansible copy module to move filebeat-config.yml onto the Web VMs.
     + You can use the command module to run curl, dpkg, and Filebeat commands.
     + Use curl -O or curl -o to download the dpkg file.

**Note:** You can use the following template for configuring the Filebeat playbook: [Filebeat Playbook Template](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Activities/Stu_Day_2/Unsolved/Resources/filebeat-playbook.yml" \o "Resources/filebeat-playbook.yml). You can also build your own if you'd like an additional challenge.

After you create and save this file, run it to install Filebeat on the DVWA machines.

**Part 4: Verifying Installation and Playbook**

1. After the playbook completes, follow the steps below to confirm that the ELK stack is receiving logs from your DVWA machines:
   * Navigate back to the Filebeat installation page on the ELK server GUI.
   * On the same page, scroll to **Step 5: Module Status** and click **Check Data**.
   * Scroll to the bottom of the page and click **Verify Incoming Data**.

If your installation was successful, take a screenshot of what you see before proceeding.

**Day 2 Milestone**

If your ELK server is receiving logs, congratulations! You've successfully deployed a live, functional ELK stack and now have plays that can:

* Install and launch Docker containers on a host machine.
* Configure and deploy an ELK server.
* Install Filebeat on any Debian-flavored Linux server.

| **:warning: Checkpoint :warning:** |
| --- |
| :heavy\_check\_mark: Please use the checklist to verify that following critical tasks were completed. |
| :heavy\_check\_mark: Created an Ansible Playbook that installs and configures Filebeat. |
| :heavy\_check\_mark: The Ansible playbook is able to be run on any of your Web VMs. |
| :heavy\_check\_mark: Filebeat is installed and running on each of your Web VMs. |
| :heavy\_check\_mark: The ELK server is receiving logs from each of your Web VMs. |

Even more significant is that you've done all of this through automation with Ansible. Now you can recreate exactly the same setup in minutes.

If you have time, create a play to install Metricbeat. After this, you'll have programmed plays to automatically install 25% of the most common Beats.

**Bonus: Creating a Play to Install Metricbeat**

Note that there are fewer instructions and setup files provided here. However, the process is similar to the one used for the Filebeat installation.

* Navigate to your ELK server's IP.
  + Click **Add Metric Data**.
  + Click **Docker Metrics**.
  + Click the **DEB** tab under **Getting Started** for the correct Linux instructions.
* Return to your Ansible VM. Update your playbook with tasks that perform the following:
  + Download the [Metricbeat .deb file](https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.4.0-amd64.deb" \o "https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.4.0-amd64.deb).
  + Use dpkg to install the .deb file.
  + Update and copy the provided [Metricbeat config file](https://gist.githubusercontent.com/slape/58541585cc1886d2e26cd8be557ce04c/raw/0ce2c7e744c54513616966affb5e9d96f5e12f73/metricbeat" \o "https://gist.githubusercontent.com/slape/58541585cc1886d2e26cd8be557ce04c/raw/0ce2c7e744c54513616966affb5e9d96f5e12f73/metricbeat).
  + Run the metricbeat modules enable docker command.
  + Run the metricbeat setup command.
  + Run the metricbeat -e command.
  + Enable the Metricbeat service on boot.
* Verify that your play works as expected:
  + On the Metricbeat Installation Page in the ELK server GUI, scroll to **Step 5: Module Status** and click **Check Data**.

If your installation was successful, take a screenshot of what you see before proceeding.

**Troubleshooting and Common ELK Container Issues:**

**Common Issue:** When the ELK VM is restarted, the container doesn't restart automatically, ELK doesn't run and logs are not transferred.

Check the following:

* Make sure you have the 'restart\_policy' set correctly in the Ansible playbook:

restart\_policy: always

* Make sure you have the vm\_map\_max setting in the Ansible playbook:

*# Use command module*

- name: Increase virtual memory

command: sysctl -w vm.max\_map\_count=262144

*# Use shell module*

- name: Increase virtual memory on restart

shell: echo "vm.max\_map\_count=262144" >> /etc/sysctl.conf

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Activity File: Completing the README**

As you know, all technical projects come with some sort of documentation, which serves as a technical brief. One of the most common forms of documentation is the README file: a simple markdown file containing information about a project.

**Instructions**

README formats vary across projects, but you can use this template to get started: [README.zip](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/1-Lesson-Plans/13-Elk-Stack-Project/Resources/README.zip).

* Download and unzip the template. Inside, you'll find a file called README.md, which contains the template. Much of the contents are provided, but you must fill in the TODO fields.
* You will also notice an Images folder. A few TODO items require you to place screenshots in your README. Place your screenshots in the Images folder, and update the README template with the appropriate file name.

For homework, you will create a GitHub repository where you will save your project files and this README. Your repository will include:

* Your network diagram.
* A description of the deployment.
* Tables specifying access policies and network addresses.
* A description of the investigation you completed using Kibana.
* Usage instructions.

This professional-level repository will prove you have the knowledge and communication skills that hiring managers are looking for.

While it may feel less substantial than the project itself, one of the most important skills a cybersecurity professional can have is the ability to articulate what they know. The README is an important capstone to the project and will serve as a compelling portfolio item for prospective employers.

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

# Automated ELK Stack Deployment

This document contains the following details:

* Description of the Topology
* ELK Configuration
  + Beats in Use
  + Machines Being Monitored
* How to Use the Ansible Build
* Access Policies

### Description of the Topology

This repository includes code defining the infrastructure below.

The main purpose of this network is to expose a load-balanced and monitored instance of DVWA, the "D\*mn Vulnerable Web Application"

Load balancing ensures that the application will be highly **available**, in addition to restricting **inbound access** to the network. The load balancer ensures that work to process incoming traffic will be shared by both vulnerable web servers. Access controls will ensure that only authorized users — namely, ourselves — will be able to connect in the first place.

Integrating an ELK server allows users to easily monitor the vulnerable VMs for changes to the **file systems of the VMs on the network**, as well as watch **system metrics**, such as CPU usage; attempted SSH logins; sudo escalation failures; etc.

The configuration details of each machine may be found below.

| **Name** | **Function** | **IP Address** | **Operating System** |
| --- | --- | --- | --- |
| Jump Box | Gateway | 10.0.0.4 | Linux |
| DVWA 1 | Web Server | 10.0.0.5 | Linux |
| DVWA 2 | Web Server | 10.0.0.6 | Linux |
| ELK | Monitoring | 10.0.0.8 | Linux |

In addition to the above, Azure has provisioned a **load balancer** in front of all machines except for the jump box. The load balancer's targets are organized into the following availability zones:

* **Availability Zone 1**: DVWA 1 + DVWA 2
* **Availability Zone 2**: ELK

## ELK Server Configuration

The ELK VM exposes an Elastic Stack instance. **Docker** is used to download and manage an ELK container.

Rather than configure ELK manually, we opted to develop a reusable Ansible Playbook to accomplish the task. This playbook is duplicated below.

To use this playbook, one must log into the Jump Box, then issue: ansible-playbook install\_elk.yml elk. This runs the install\_elk.yml playbook on the elk host.

### Access Policies

The machines on the internal network are not exposed to the public Internet.

Only the **jump box** machine can accept connections from the Internet. Access to this machine is only allowed from the IP address 64.72.118.76

* **Note**: Your answer will be different!

Machines within the network can only be accessed by **each other**. The DVWA 1 and DVWA 2 VMs send traffic to the ELK server.

A summary of the access policies in place can be found in the table below.

| **Name** | **Publicly Accessible** | **Allowed IP Addresses** |
| --- | --- | --- |
| Jump Box | Yes | 64.72.118.76 |
| ELK | No | 10.0.0.1-254 |
| DVWA 1 | No | 10.0.0.1-254 |
| DVWA 2 | No | 10.0.0.1-254 |

### Elk Configuration

Ansible was used to automate configuration of the ELK machine. No configuration was performed manually, which is advantageous because...

* TODO: What is the main advantage of automating configuration with Ansible?

The playbook implements the following tasks:

* TODO: In 3-5 bullets, explain the steps of the ELK installation play. E.g., install Docker; download image; etc.
* ...
* ...

The following screenshot displays the result of running docker ps after successfully configuring the ELK instance.

* TODO: Update the image file path with the name of your screenshot of docker ps output:

The playbook is duplicated below.

**---**

*# install\_elk.yml*

- name: Configure Elk VM with Docker

hosts: elkservers

remote\_user: elk

become: true

tasks:

*# Use apt module*

- name: Install docker.io

apt:

update\_cache: yes

name: docker.io

state: present

*# Use apt module*

- name: Install pip3

apt:

force\_apt\_get: yes

name: python3-pip

state: present

*# Use pip module*

- name: Install Docker python module

pip:

name: docker

state: present

*# Use command module*

- name: Increase virtual memory

command: sysctl -w vm.max\_map\_count=262144

*# Use sysctl module*

- name: Use more memory

sysctl:

name: vm.max\_map\_count

value: "262144"

state: present

reload: yes

*# Use docker\_container module*

- name: download and launch a docker elk container

docker\_container:

name: elk

image: sebp/elk:761

state: started

restart\_policy: always

published\_ports:

- 5601:5601

- 9200:9200

- 5044:5044

### Target Machines & Beats

This ELK server is configured to monitor the DVWA 1 and DVWA 2 VMs, at 10.0.0.5 and 10.0.0.6, respectively.

We have installed the following Beats on these machines:

* Filebeat
* Metricbeat
* Packetbeat

These Beats allow us to collect the following information from each machine:

* **Filebeat**: Filebeat detects changes to the filesystem. Specifically, we use it to collect Apache logs.
* **Metricbeat**: Metricbeat detects changes in system metrics, such as CPU usage. We use it to detect SSH login attempts, failed sudo escalations, and CPU/RAM statistics.
* **Packetbeat**: Packetbeat collects packets that pass through the NIC, similar to Wireshark. We use it to generate a trace of all activity that takes place on the network, in case later forensic analysis should be warranted.

The playbook below installs Metricbeat on the target hosts. The playbook for installing Filebeat is not included, but looks essentially identical — simply replace metricbeat with filebeat, and it will work as expected.

**---**

- name: Install metric beat

hosts: webservers

become: true

tasks:

*# Use command module*

- name: Download metricbeat

command: curl -L -O https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.4.0-amd64.deb

*# Use command module*

- name: install metricbeat

command: dpkg -i metricbeat-7.4.0-amd64.deb

*# Use copy module*

- name: drop in metricbeat config

copy:

src: /etc/ansible/files/metricbeat-config.yml

dest: /etc/metricbeat/metricbeat.yml

*# Use command module*

- name: enable and configure docker module for metric beat

command: metricbeat modules enable docker

*# Use command module*

- name: setup metric beat

command: metricbeat setup

*# Use command module*

- name: start metric beat

command: service metricbeat start

### Using the Playbooks

In order to use the playbooks, you will need to have an Ansible control node already configured. We use the **jump box** for this purpose.

To use the playbooks, we must perform the following steps:

* Copy the playbooks to the Ansible Control Node
* Run each playbook on the appropriate targets

The easiest way to copy the playbooks is to use Git:

$ cd /etc/ansible

$ mkdir files

*# Clone Repository + IaC Files*

$ git clone https://github.com/yourusername/project-1.git

*# Move Playbooks and hosts file Into `/etc/ansible`*

$ cp project-1/playbooks/\* .

$ cp project-1/files/\* ./files

This copies the playbook files to the correct place.

Next, you must create a hosts file to specify which VMs to run each playbook on. Run the commands below:

$ cd /etc/ansible

$ cat > hosts <<EOF

[webservers]

10.0.0.5

10.0.0.6

[elk]

10.0.0.8

EOF

After this, the commands below run the playbook:

$ cd /etc/ansible

$ ansible-playbook install\_elk.yml elk

$ ansible-playbook install\_filebeat.yml webservers

$ ansible-playbook install\_metricbeat.yml webservers

To verify success, wait five minutes to give ELK time to start up.

Then, run: curl http://10.0.0.8:5601. This is the address of Kibana. If the installation succeeded, this command should print HTML to the console.

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

## Day 3 Solution Guide: Diagramming the Network

Your network diagram should resemble the following:

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Activity File: Diagramming the Network**

Now that you've deployed your ELK instance, your virtual network is stable. You won't be adding anything to it for awhile.

Since the network is essentially complete, it's time to document what you've built. This is an important last step for any deployment.

**Instructions**

Use [Gliffy](https://www.gliffy.com/" \o "https://www.gliffy.com) or [Draw.io](https://draw.io/) to diagram your network. Make sure your diagram includes:

* **VNet**: Create a box that contains the machines on your virtual network. Within your VNet, diagram the following:
  + Jump box and other VMs.
  + Ansible control node.
  + Specify which VM hosts the DVWA containers.
  + Specify which VM hosts ELK stack containers.
* **Security group**: Create a box around your VNet to indicate the security group, and use a text field to specify the rules you have in place.
* **Access from the internet**: Add an icon representing the public internet and indicate how it connects to VMs in your VNet.

Use a text field to label each VM with the following information:

* Network (IP) address
* Operating system and version
* Installed containers
* Exposed ports
* Allowed IP addresses

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Activity File: Exploring Kibana**

* You are a DevOps professional and have set up monitoring for one of your web servers. You are collecting all sorts of web log data and it is your job to review the data regularly to make sure everything is running smoothly.
* Today, you notice something strange in the logs and you want to take a closer look.
* Your task: Explore the web server logs to see if there's anything unusual. Specifically, you will:

:warning: **Heads Up**: These sample logs are specific to the time you view them. As such, your answers will be different from the answers provided in the solution file.

**Instructions**

1. Add the sample web log data to Kibana.
2. Answer the following questions:
   * In the last 7 days, how many unique visitors were located in India?
   * In the last 24 hours, of the visitors from China, how many were using Mac OSX?
   * In the last 2 days, what percentage of visitors received 404 errors? How about 503 errors?
   * In the last 7 days, what country produced the majority of the traffic on the website?
   * Of the traffic that's coming from that country, what time of day had the highest amount of activity?
   * List all the types of downloaded files that have been identified for the last 7 days, along with a short description of each file type (use Google if you aren't sure about a particular file type).
3. Now that you have a feel for the data, Let's dive a bit deeper. Look at the chart that shows Unique Visitors Vs. Average Bytes.
   * Locate the time frame in the last 7 days with the most amount of bytes (activity).
   * In your own words, is there anything that seems potentially strange about this activity?
4. Filter the data by this event.
   * What is the timestamp for this event?
   * What kind of file was downloaded?
   * From what country did this activity originate?
   * What HTTP response codes were encountered by this visitor?
5. Switch to the Kibana Discover page to see more details about this activity.
   * What is the source IP address of this activity?
   * What are the geo coordinates of this activity?
   * What OS was the source machine running?
   * What is the full URL that was accessed?
   * From what website did the visitor's traffic originate?
6. Finish your investigation with a short overview of your insights.
   * What do you think the user was doing?
   * Was the file they downloaded malicious? If not, what is the file used for?
   * Is there anything that seems suspicious about this activity?
   * Is any of the traffic you inspected potentially outside of compliance guidlines?

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Solution Guide: Exploring Kibana**

1. Start by adding the sample web log data to Kibana.
   * You can import it by clicking **Try our sample data**.
   * Or you can import it from the homepage by clicking on **Load a data set and a Kibana dashboard** under **Add sample data**.
   * Click **Add Data** under the **Sample Web Logs** data pane.
   * Click **View Data** to pull up the dashboard.
2. Answer the following questions:
   * In the last 7 days, how many unique visitors were located in India?
     + **Example Answer:** 253
   * In the last 24 hours, of the visitors from China, how many were using Mac OSX?
     + **Example Answer:** 7
   * In the last 2 days, what percentage of visitors received 404 errors? How about 503 errors?
     + **Example Answer:** 404: 6.667% and 503: 13.333%
   * In the last 7 days, what country produced the majority of the traffic on the website?
     + **Example Answer:** China
   * Of the traffic that's coming from that country, what time of day had the highest amount of activity?
     + **Example Answer:** 12 p.m. and 1 p.m. (hours 12 and 13)
   * List all the types of downloaded files that have been identified for the last 7 days, along with a short description of each file type (use Google if you aren't sure about a particular file type).
     + **Example Answer:**
       - **gz:** .gz files are compressed files created using the gzip compression utility.
       - **css:** .css files can help define font, size, color, spacing, border and location of HTML information on a webpage. They are downloaded with their .html counterparts and rendered by the browser.
       - **zip:** A lossless compression format. A .zip file may contain one or more files or directories that have been compressed.
       - **deb:** A file with the .deb file extension is a Debian (Linux) Software Package file. These files are installed when using the apt package manager.
       - **rpm:** .rpm file formats are a Red Hat Software Package file. RPM stands for Red Hat Package Manager.
3. Look at the chart that shows Unique Visitors Vs. Average Bytes.
   * Locate the time frame in the last 7 days with the most amount of bytes (activity).
   * In your own words, is there anything that seems potentially strange about this activity?

**Example Answer:** (Your results may be different.) In our example, it seems strange that *one* visitor is using a number of bytes that is considerably higher than all other usage.

1. Filter the data by this event.
   * What is the timestamp for this event?
     + **Example Answer:** The time filter shows Sep 13, 2020 @ 21:00 -> Sep 14, 2020 @ 00:00. The time stamp is 22:55.
   * What kind of file was downloaded?
     + **Example Answer:** An RPM file
   * From what country did this activity originate?
     + **Example Answer:** India
   * What HTTP response codes were encountered by this visitor?
     + **Example Answer:** 200 OK
2. Switch over to the Kibana Discover page to see more details about this activity.
   * What is the source IP address of this activity?
     + **Example Answer:** 35.143.166.159
   * What are the geo coordinates of this activity?
     + **Example Answer:** { "lat": 43.34121, "lon": -73.6103075 }
   * What OS was the source machine running?
     + **Example Answer:** Windows 8
   * What is the full URL that was accessed?
     + **Example Answer:** <https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-6.3.2-i686.rpm>
   * From what website did the visitor's traffic originate?
     + **Example Answer:** Facebook
3. Finish your investigation with a short overview of your insights.
   * What do you think the user was doing?
     + **Example Answer:** This event appears to be a user downloading a Linux package from the website being monitored.
   * Was the file they downloaded malicious? If not, what is the file used for?
     + Linux packages aren't typically malicious, but they could be. Depending on the website, this could be harmless traffic from a sysadmin performing an update.
   * Was there anything that seems suspicious about this activity?
   * Is any of the traffic you inspected potentially outside of compliance guidelines?
     + **Example Answer:** The main concern is the referral link from Facebook, as it's probably not within compliance to post package update links on Facebook.
     + This user could be further investigated and monitored for suspicious activity.

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Activity File: Interview Questions**

* This first project covers a wide range of topics including cloud, network security, and logging and monitoring.
* When networking and talking to potential employers, you should be able to reference the work done on this project to answer specific interview questions or demonstrate your skills within a specific domain.
* You will choose a domain that you're interested in pursuing as a career and answer mock questions based on the suggested response format.​

**Instructions**

1. Choose one of the following domains:
   * Network security
   * Cloud security
   * Logging and monitoring

If you are unsure of which domain you want to focus on, that's okay. You can either choose the one you're most comfortable discussing, or complete the tasks in two or all three domains.

1. Select one domain and one question.
   * Questions are provided for each domain. Choose one to answer from your chosen domain.​
2. Write a one-page response that answers the question using specific examples from your work on Project 1. Your response should flow and read like a presentation while keeping the general structure of the technical question response guidelines.

You will submit this one-page response.

**Reminder: Response Guidelines**

As a reminder, good responses do the following.​

1. Restate the problem.
2. Provide a concrete example scenario.
3. Explain the solution requirements.
4. Explain the solution details.
5. Identify advantages and disadvantages of the solution​.​Including each of these components will ensure you prove your competency of subject matter and critical thinking.​

**Interview Questions by Domain**

Below you will find a list of questions, grouped by specific domains. Select one question to answer.​

For each question, where appropriate, we have provided you with specific prompts to consider as you structure each section of your response. Feel free to use these prompts or your own examples.

**Domain: Network Security**

**Question 1: Faulty Firewall**

Suppose you have a firewall that's supposed to block SSH connections, but instead lets them through. How would you debug it?

Make sure each section of your response answers the questions laid out below.​

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, did you allow SSH traffic to all of the VMs on your network?
   * Which VMs did accept SSH connections?
   * What happens if you try to connect to a VM that does not accept SSH connections? Why?
3. Explain the Solution Requirements
   * If one of your Project 1 VMs accepted SSH connections, what would you assume the source of the error is?
   * Which general configurations would you double-check?
   * What actions would you take to test that your new configurations are effective?
4. Explain the Solution Details
   * Which specific panes in the Azure UI would you look at to investigate the problem?
   * Which specific configurations and controls would you check?
   * What would you look for, specifically?
   * How would you attempt to connect to your VMs to test that your fix is effective?
5. Identify Advantages/Disadvantages of the Solution
   * Does your solution guarantee that the Project 1 network is now "immune" to all unauthorized access?
   * What monitoring controls might you add to ensure that you identify any suspicious authentication attempts?​

**Question 2: Unsecured Web Server**

Suppose you find a server running HTTP on port 80, despite compliance guidelines requiring encryption in motion. What do you do?​​

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, did you have servers running HTTP on port 80? If so, why was it permissible to do so?
   * In a real deployment, which specific machine would you configure differently? How, and why?
3. Explain the Solution Requirements
   * Why is running HTTP on port 80 a potential problem?
   * How would you reconfigure a server to serve HTTP traffic safely?
   * How does this solution fix the problem?
4. Explain the Solution Details
   * Which tools and technologies would you use to implement this solution in Project 1?
   * How, specifically, would you use these tools to harden your deployment?
5. Identify Advantages and Disadvantages of the Solution
   * Will your solution break clients that used to communicate with the server over port 80?
   * Do you have to do any work to keep this solution running longterm? Or can you simply "set it and forget it?”

**Domain: Cloud Security**

**Question 1: Cloud Access Control**

How would you control access to a cloud network?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, did you deploy an on-premises or cloud network?
   * Did you have to configure access controls to this network?
   * What kinds of access controls did you configure, and why were they necessary?
   * How do these details relate to the interview question?
3. Explain the Solution Requirements
   * In Project 1, what kinds of access controls did you have to implement? Consider:
     + NSGs around the VNet? Around the VMs?
     + Local firewalls (ufw, etc.) on each VM?
     + Protocol allow/deny lists?
   * What did each access control achieve, and why was this restriction necessary for the project?
4. Explain the Solution Details
   * Which rules do you set for each NSG in the network?
   * How does access to the jump box work?
   * How does access from the jump box to the web servers work?
5. Identify Advantages/Disadvantages of the Solution
   * Does your solution scale?
   * Is there a better solution than a jump box?
   * What are the disadvantages of implementing a VPN that kept you from doing it this time?
   * What are the advantages of a VPN?
   * When is it appropriate to use a VPN?

**Question 2: Corporate VPN**

What are the advantages and disadvantages of using a corporate VPN, and under what circumstances is using one appropriate?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, which VMs did you have on the network?
   * Which tools did you use to control access to and from the network?
   * If you didn't use a VPN, what did you use?
   * What disadvantage(s) did your non-VPN solution have?
   * What advantage(s) did your non-VPN solution have?
3. Explain the Solution Requirements
   * Would a VPN meet the access control requirements you had for Project 1?
   * How would a VPN protect the network just as well, or better, than your current solution?
4. Explain the Solution Details
   * Which Azure tools would you use to implement a VPN to your Project 1 network?
   * How would you onboard users to the new VPN system?
5. Identify Advantages and Disadvantages of the Solution
   * In Project 1, would a VPN have been an appropriate access control solution?
   * Under what circumstances is a VPN a good solution?
   * When, if ever, is a VPN "overkill"?

**Question 3: Containers**

When is it appropriate to use containers in cloud deployments, and what are the security benefits of doing so?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, when did you use containers?
   * What did you use containers for?
3. Explain the Solution Requirements
   * Why was this an appropriate use for containers?
   * What security benefits did you expect from using containers?
4. Explain the Solution Details
   * In Project 1, how did you configure VMs to be able to run containers?
   * How did you select and install the correct container?
   * How did you verify that it was running correctly?
5. Identify Advantages/Disadvantages of the Solution
   * How would you have achieved the same thing without containers?
   * What are the advantages to doing it without containers?
   * What are the disadvantages?

**Question 4: Cloud Infrastructure as Code**

What are the security benefits of defining cloud infrastructure as code?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, when did you use infrastructure as code (IaC)?
   * What tool did you use?
   * What did you use it to do?
3. Explain the Solution Requirements
   * Were there any alternatives to IaC?
   * What benefits does IaC have over alternative approaches?
4. Explain the Solution Details
   * In Project 1, which specific configurations did your IaC set up?
   * How did you run and test these configurations?
5. Identify Advantages/Disadvantages of the Solution
   * Are there any disadvantages to using IaC over the "traditional" approach?

**Domain: Logging and Monitoring**

**Question 1: Setting Alerts in a New Monitoring System**

How do you determine which alerts to set in a new monitoring system?

Note: In Project 1, you did not set up any alerts. However, you still have enough experience to answer this question.

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * Describe the network you built for Project 1. Identify the VMs on the network and what they do.
   * Which VMs should be publicly accessible?
   * Which VMs should not be publicly accessible?
3. Explain the Solution Requirements
   * Consider the VMs that should not be publicly accessible from the internet. Which alert(s) should these VMs fire and when?
   * Why should these VMs be associated with these alerts?
4. Explain the Solution Details
   * Which tool in Project 1 would you use to set such an alert?
   * What would the alert rule be? For example, would the alert fire upon a failed SSH attempt or a ping request?
5. Identify Advantages and Disadvantages
   * Are there any malicious circumstances that the alert(s) discussed above do not address?

**Question 2: Challenges of Collecting Large Amounts of Log Data**

What are the challenges of collecting huge amounts of log data? How do security analysts deal with them?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * In Project 1, when did you deal with log data?
   * What kind(s) of data did you investigate?
   * How much data were you dealing with?
   * What were you looking for?
3. Explain the Solution Requirements
   * What information did you need to find what you were looking for?
   * What does an analyst need to analyze large amounts of log data to find this information?
   * In Project 1, what tools did you use to analyze log data?
4. Explain the Solution Details
   * How did you use these tools to find the log data? E.g., which charts, graphs, etc. were useful for parsing the logs?
5. Identify Advantages and Disadvantages of the Solution
   * What kinds of data did you not inspect during Project 1?
   * Would having access to this additional data have changed your process or conclusions? If so, how?

**Question 3: Escalating Security Events**

How do you determine if a security event or alert is important enough for escalation?

1. Restate the Problem
2. Provide a Concrete Example Scenario
   * What kinds of events and alerts did you encounter in Project 1?
   * Which of these events was most interesting or suspicious?
   * Why was the event suspicious? What led you to investigate it?
3. Explain the Solution Requirements
   * What do you need to figure out in order to determine if this event is worth escalating?
4. Explain the Solution Details
   * How did you use Kibana to find this information?
5. Identify Advantages and Disadvantages of the Solution
   * How confident are you in your conclusion?
   * What additional data would be useful to determine if your conclusions are correct?

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Activity File: Kibana Continued**

* This week, you created the infrastructure behind a security information and event management system such as Kibana. Once that set up is complete, you will have finished the project.
* This optional activity tasks you with exploring more Kibana capabilities, some of which you will use in future projects.
* **Note**: In order to complete these activities, you will need to complete the optional Metricbeat configuration.

**Scenario**

In this activity, you will suppose the role of a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team.

Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

You will have three tasks:

1. Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.
2. Generate a high amount of CPU usage on the pen-testing machines and verify that Kibana picks up this data.
3. Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

These activities will guide you though generating some data to visualize in Kibana. Each of these activity will require the following high level steps:

1. Use your jump-box to attack your web machines in various ways.
2. Use a Linux utility to stress the system of a webVM directly.
3. Subsequently generate traffic and logs that Kibana will collect.
4. View that traffic in various ways inside Kibanna.

It's also worth noting that these activities comprise different job roles:

* Getting the infrastructure setup and maintaining it is the role of a security engineer or cloud architect.
* Using that infrastructure by creating dashboards and alerts fall under the security analyst role. It would be rare to have a position where you would be required to do both.

That said, now that we have Kibana setup and gathering data from three web servers, its worth learning how to visualize data in Kibana.

Before getting started, we'll have to complete some metrics and logs set up.

**Setup: Kibana Metrics and Logs Orientation**

Before we begin generating traffic, locate the two screens inside Kibana that you will use to visualize this traffic:

* Logs
* Metrics

These pages will show you the changes in data that we will create.

**Logs**

* Click **Logs** to see some general system logs coming from the web machines.
* Notice that you can stream logs live from the machines.

**Metrics**

* Next, click **Metrics** on the left side.
  + Here we can see each of our VMs that are sending metrics.
* Click on one of the large squares that represent one of your VMs.
* Choose **View metrics** from the dropdown that appears.
* Notice that you can see CPU and memory usage here.

Now that we know where to look for this data, let's generate some unusual network traffic.

**Activity Tasks**

Expand the provided activity files to complete each task. These tasks can be completed in any order.

**SSH Barrage**

Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.

 Activity File: SSH Barrage

**Linux Stress**

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

 Activity File: Linux Stress

**wget-DoS**

Task: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

 Activity File: wget-DoS

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Solution Guide: Kibana Continued**

**Scenario**

In this activity, you played the role of a cloud architect and were tasked with setting up an ELK server to gather logs for the Incident Response team.

Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

You had three tasks:

1. Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.
2. Generate a high amount of CPU usage on the pen-testing machines and verify that Kibana picks up this data.
3. Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

**SSH Barrage**

Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.

 Solution Guide: SSH Barrage

**Linux Stress**

Task: Generate a high amount of CPU usage on the pen-testing machines and verify that Kibana picks up this data.

 Solution Guide: Linux Stress

**wget-DoS**

Task: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

 Solution Guide: wget-DoS

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**Unit 13 README: ELK Stack Project**

**Unit Description**

In the first project week, you will configure an ELK stack server in order to set up a cloud monitoring system. This project will result in tangible deliverables that demonstrate your knowledge of cloud, network security, logging and monitoring.

**Unit Objectives**

Click here to view the daily unit objectives.

**Lab Environment**

For the majority of demonstrations and activities, the class will use Microsoft Azure cloud services and the Azure cloud portal.

* You will **not** be using any of the Azure lab environments. Instead, you will be using personal Azure accounts.

**What to Be Aware Of:**

* The sample logs used in this unit are specific to the time in which they are viewed. As such, answers will vary from the answers provided in the solutions.
* The VM for the ELK server **must** have at least 4GiB of memory for the ELK container to run properly. Azure has VM options that have 3.5 GiB of memory, but **do not use them**. They will not properly run the ELK container because they do not have enough memory.
* Azure may run out of available VMs for you to create a particular region. If this happens, you will need to do one of two things:
  + Open a support ticket with Azure support using [these instructions](https://docs.microsoft.com/en-us/azure/azure-portal/supportability/how-to-create-azure-support-request). Azure support is generally very quick to resolve issues.
  + Create another vNet in another region and attempt to create the ELK sever in that region.

**Security+ Domains**

This unit covers portions of the following domains on the Security+ exam:

 Click here to view Security+ Domains that apply to this project.

For more information about these Security+ domains, refer to the following resource: [Security+ Exam Objectives](https://www.comptia.jp/pdf/Security%2B%20SY0-501%20Exam%20Objectives.pdf)

**Additional Reading and Resources**

 Click here to view additional reading materials and resources.

**Unit 13 : Homework**

This unit's homework assignment can be viewed here:

* [Unit 13 Homework File](vscode-webview-resource://3dc8bd42-e305-4b76-b6c6-667ee1c655a6/file/c%3A/class/ku-ove-cyber-pt-12-2020-u-c/CourseMaterial/2-Homework/13-Github-Fundamentals/Unsolved/README.md)

**Looking Forward**

The next unit will mark the start of the Offensive Security module. In the Web Development unit, we'll examine the infrastructure and deployment of web applications to inform how we can explore and exploit vulnerabilities within those applications.

© 2020 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.

**13.1-13.3 Student Guide: ELK Stack Project Week**

**Overview**

This week, you will set up a cloud monitoring system by configuring an ELK stack server.

**Week Objectives**

You will use the following skills and knowledge to complete the following project steps:

* Deploying containers using Ansible and Docker.
* Deploying Filebeat using Ansible.
* Deploying the ELK stack on a server.
* Diagramming networks and creating a README.

**Note:** While you must complete your projects individually, you can work through problems together, and should ask instructional staff for help if you get stuck.

**Important:** Due to Azure Free account limitations, you can only utilize 4vCPUs per region in Azure. Because of this, we will need to create a *new* vNet in another region for our ELK server.

**Lab Environment**

You will continue using your personal Azure account and build upon your existing Azure VMs. You will **not** be using your cyberxsecurity accounts.

**Additional Resources**

* [Ansible Documentation](https://docs.ansible.com/ansible/latest/modules/modules_by_category.html)
* [elk-docker Documentation](https://elk-docker.readthedocs.io/#Elasticsearch-logstash-kibana-elk-docker-image-documentation)
* [Virtual Memory Documentation](https://www.elastic.co/guide/en/elasticsearch/reference/current/vm-max-map-count.html)
* ELK Server URL: [http://your-IP:5601/app/kibana#/home?\_g=()](http://your-ip:5601/app/kibana#/home?_g=())
* [Docker Commands Cheatsheet](https://phoenixnap.com/kb/list-of-docker-commands-cheat-sheet)

**Slideshow**

The slideshow to this week is located on Google Drive here: [ELK Stack Project Week (13) Slides](https://docs.google.com/presentation/d/1b0jbp5L_ws2iCFuOSnU7BfoXb6oSiWccqmwXKk8yJ0w/edit#slide=id.g4789b2c72f_0_6)

**Day 1: Configuring an ELK Server**

Lectures cover the following:

* Give an overview of the ELK stack and how it performs network security monitoring. This overview will also give you valuable context for why you're configuring and deploying these tools during the week.
* Provide the project overview as well as suggested milestones for each day.
* Due to Azure Free account limitations, you can only utilize 4vCPUs per region in Azure. Therefore, we will need to create a new vNet in another region for our ELK server.
* By the end of the project, we will have an ELK server deployed and receiving logs from all three web VMs created in the previous cloud weeks.

Activities involve the following:

* Create a new vNet in Azure in a different region, within the same resource group.
* Create a peer-to-peer network connection between your vNets.
* Create a new VM in the new vNet that has 2vCPUs and a minimum of 4GiB of memory.
* Add the new VM to Ansible’s hosts file in your provisioner VM.
* Create an Ansible playbook that installs Docker and configures an ELK container.
* Run the playbook to launch the container.
* Restrict access to the ELK VM.

**Click here to view the 13.1 Student Guide.**

**Day 2: Filebeat**

Lectures cover the following

* Provide a brief overview of Filebeat.

Activities involve the following:

* Navigate to the ELK server’s GUI to view Filebeat installation instructions.
* Create a Filebeat configuration file.
* Create an Ansible playbook that copies this configuration file to the DVWA VMs and then installs Filebeat.
* Run the playbook to install Filebeat.
* Confirm that the ELK Stack is receiving logs.
* Install Metricbeat as a bonus activity.

**Click here to view the 13.2 Student Guide.**

**Day 3: Exploration, Diagramming and Documentation**

Lectures will cover:

* Kibana's features and demonstration of how to navigate datasets.
* Supplemental assignment in which you answer interview-style questions about your project.

Activities involve the following:

* Finalize the network diagram you began during the cloud security week.
* Draft a README explaining what you've built.
* Craft interview response questions.

**Click here to view the 13.3 Student Guide.**