



Western Washington University
Western CEDAR

WWU Honors College Senior Projects

WWU Graduate and Undergraduate Scholarship

Spring 2021

Reflections on setting up the Cyber Range Intrusion Detection System

William Pearson
Western Washington University

Follow this and additional works at: https://cedar.wwu.edu/wwu_honors



Part of the [Information Security Commons](#), and the [Mathematics Commons](#)

Recommended Citation

Pearson, William, "Reflections on setting up the Cyber Range Intrusion Detection System" (2021). *WWU Honors College Senior Projects*. 476.

https://cedar.wwu.edu/wwu_honors/476

This Project is brought to you for free and open access by the WWU Graduate and Undergraduate Scholarship at Western CEDAR. It has been accepted for inclusion in WWU Honors College Senior Projects by an authorized administrator of Western CEDAR. For more information, please contact westerncedar@wwu.edu.

William Pearson

Reflections on setting up the Cyber Range Intrusion Detection System

Over the past year, I worked in a team of four students to set up and configure an Intrusion Detection System (IDS) for Western Washington University Poulsbo's Cyber Range. This was a group project composed of two Computer Science students and two Cybersecurity students and mentored by the Cyber Range master (Vipul Kumar). The goal of the project was to set up an IDS to detect and notify on attempts at intrusion.

We set up the project using an ELK stack (Elasticsearch, Logstash, and Kibana). Suricata is used as the IDS; the Cybersecurity students were involved in configuring it (and I was not); Suricata receives data from a firewall. Data from Suricata is put into a Redis keystore, and then Logstash transfers it into an Elasticsearch cluster which acts as the database. Kibana is used to create user-/administrator-facing dashboards.

We used Microsoft Teams for meetings and coordination. We had a weekly meeting with our advisor early in the week (usually Monday or Tuesday), and then had a second weekly meeting where we worked on the project itself. During those meetings we mostly worked individually; the three other group members, for the most part, each worked on one of the three parts of the ELK stack. My role was more general; I had a basic understanding of all parts of the stack and worked to resolve blocking issues and on any other component where I would be the most useful, but did not "own" any individual component. When we were blocked, we actively communicated and troubleshooted together on Teams, but otherwise we worked individually (but with the ability to ask questions together). All work was done remotely due to the pandemic, but this setup did work similarly to everyone meeting up at the same table and working individually.

We had some issues getting everything to work. For instance, getting the various nodes in our Elasticsearch cluster to talk to each other properly proved challenging, and then getting that to work with encryption enabled also took a while. Furthermore, although our setup does support email and Microsoft Teams notifications, that functionality could not be enabled due to not having the appropriate license for Kibana; that issue should be easy to fix in the future when a license is purchased.

Overall, this project was definitely interesting. I went into it feeling like we might have had too much time – it seemed odd to me that it would take a whole year to configure existing software – but it ended up being about the right amount of time. Various technical issues, along with needing to actually learn the details of the ELK stack, resulted in it taking longer than I personally had expected (but matching the actual budgeted amount of time for a senior project). This was also my first time working directly (I would say "in person", but that does not exactly apply with the pandemic) with an organized group for a full year, which was a neat experience. I got to meet interesting people and learn neat technologies, so overall, I consider it a success.

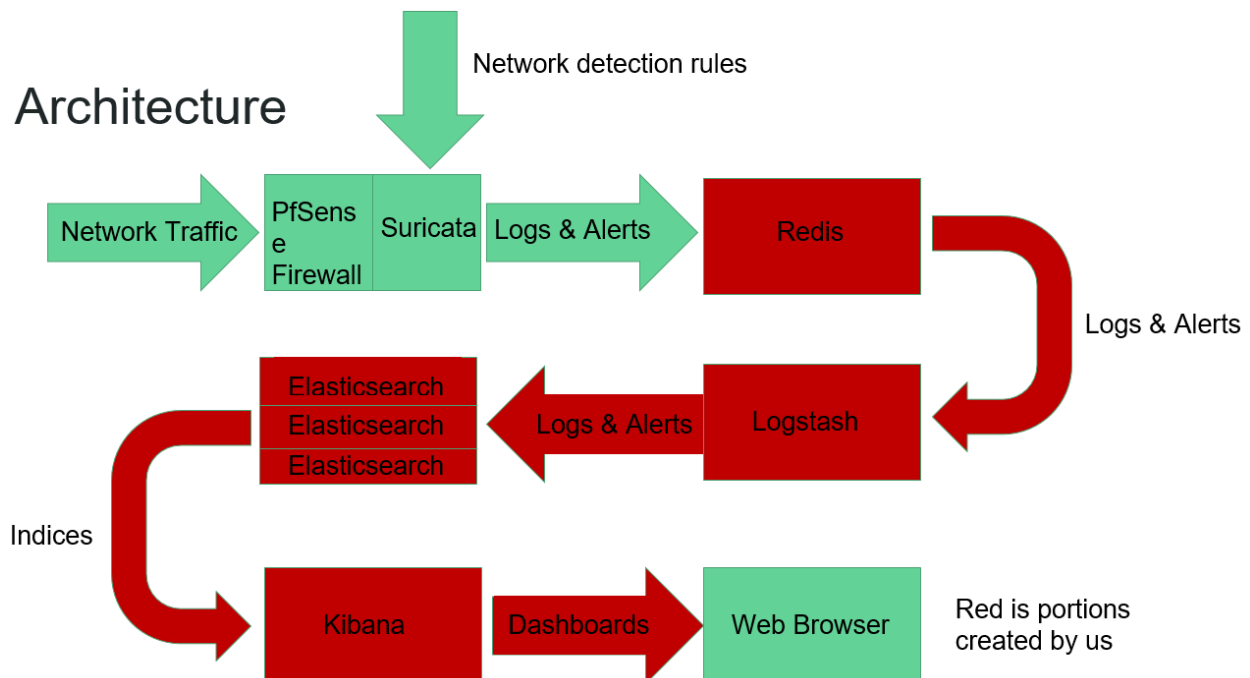
Project Deliverables/Final Report

Matthew Jackson, William Pearson, Niall O'Rourke, and Rowena Tchao

Product Summary

Western Washington University's Cyber Range required an Intrusion Detection System (IDS) to safeguard the private network from intrusion and potential abuse. Vipul Kumar, our client and advisor, implemented an IDS using Suricata on the range's firewall. Our team developed the underlying to store, index and visualize the network traffic that flows through the range. For this, we used the ELK stack (Elasticsearch, Logstash, and Kibana).

The architecture our team developed consisted of 4 components using virtual machines as our hardware: A Redis server which received the data from the IDS, Logstash that processed and indexed the data, Elasticsearch that stores the data, and Kibana that visualizes the data. Redis was the least touched component of the data pipeline, as it just receives data from Suricata and pushes it to Logstash. The rest of the architecture is all part of ELK stack and was the primary focus of our project. All 3 components have security enabled. This means that all communications are encrypted, prevents unauthorized access to the local cluster, prevents unauthorized nodes from joining the Elasticsearch cluster, and TLS communication enabled between nodes and HTTPS interface for Elasticsearch and Kibana.



Features

The ELK stack came with predefined features out of the box so that is what we used to implement the stack. Any additional features required a Gold license which we currently don't have.

Starting with the first service, Logstash, we formatted the index so that the data pattern matched the Suricata logs and the SN dashboard templates in Kibana.

For Elasticsearch, to increase reliability and availability, we created a single node Elasticsearch cluster on 3 separate virtual machines and 3 separate Docker containers.

In Kibana, we implemented dashboards that allow a user to visualize and analyze the network traffic with greater ease. An additional feature we implemented for Kibana is the ability to map the location of the traffic entering the range. With this, you can see the last location where traffic originated from. In Kibana, we used built in roles as well as a few of our own created roles to manage user privileges in Kibana.

Finally, the last feature we enabled for the entire stack, was ELK stack's x-pack security. This feature is specifically for securing the ELK stack and its HTTP traffic. This is to ensure that all communications in and out of the cluster were secure.

Sprints

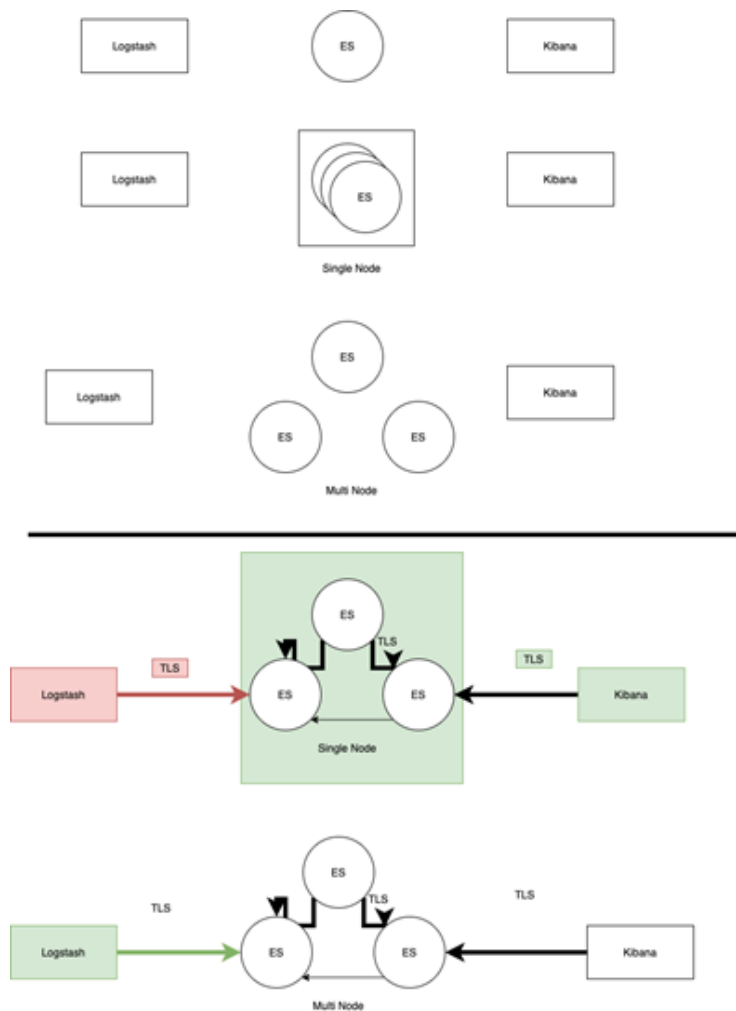
Under the advice of our mentor/client, we implemented the crawl, walk, run approach to our project. While they are similar to sprints, the spacing between these objectives were greater than agile sprints.

In our first quarter, the crawl phase, we had to learn what the ELK stack was (Elasticsearch, Logstash, Kibana) and how to use Docker. We built our own personal testing environment to practice with these services. The last part was building the stack out on our own testing environment.

In the second quarter, the walk phase, we built our first complete ELK stack. In this setup, we used a single virtual machine for Logstash, Elasticsearch and Kibana. Next, we split all 3 services onto their own virtual machine. Following this, we worked on creating additional Elasticsearch nodes and getting them to cluster, both on a single virtual as well as spread across 3 separate virtual machines. During this same period of time, we started to develop the first of our Kibana visualizations/dashboards.

In the final quarter, the run phase, we had four primary goals: build remaining visualizations, deploying pipeline on production servers, enabling security on the stack and finalizing documentation. We had two plus goals: setting up the Kibana alert system and automating deployment using Ansible. We completed all three of these tasks but failed to hit our plus goals.

The vast majority of our time was spent, enabling security. This task turned out to be incredibly challenging.



This image displays the evolution of our stack. We started with each service in a single virtual machine (not shown). After this we created it so all services were running on separate virtual machines. The squares around ES (Elasticsearch) represent the service being run on a single virtual machine. The last two images show the roadblocks we encountered during our last ELK stack test. The second to last image shows Logstash not connecting and authenticating to the ES cluster. The last image shows our final pipeline successfully implemented with TLS enabled.

Testing

We have created a script called `test_online.py` where it checks if Elasticsearch and Kibana servers are online based on a list of IP addresses stored in the script and a username and password to authenticate with. Apart from that, there is little testable code that was written for this project. However, we still tested for ELK stack functionality throughout each development phase.

For overall testing, each ELK stack piece was tested based on its effectiveness in the stack. Since our goal is to implement a secure ELK stack structure with single node multicluster Elasticsearch, we had to test the ELK stack separately from the security piece to ensure the stack was in working order before laying security on top of it. By any chance that Elasticsearch

was not clustering or Kibana and Logstash were not authenticating to the Elasticsearch cluster properly, workflow halted to dissect and troubleshoot the problem.

Bugs/Errors

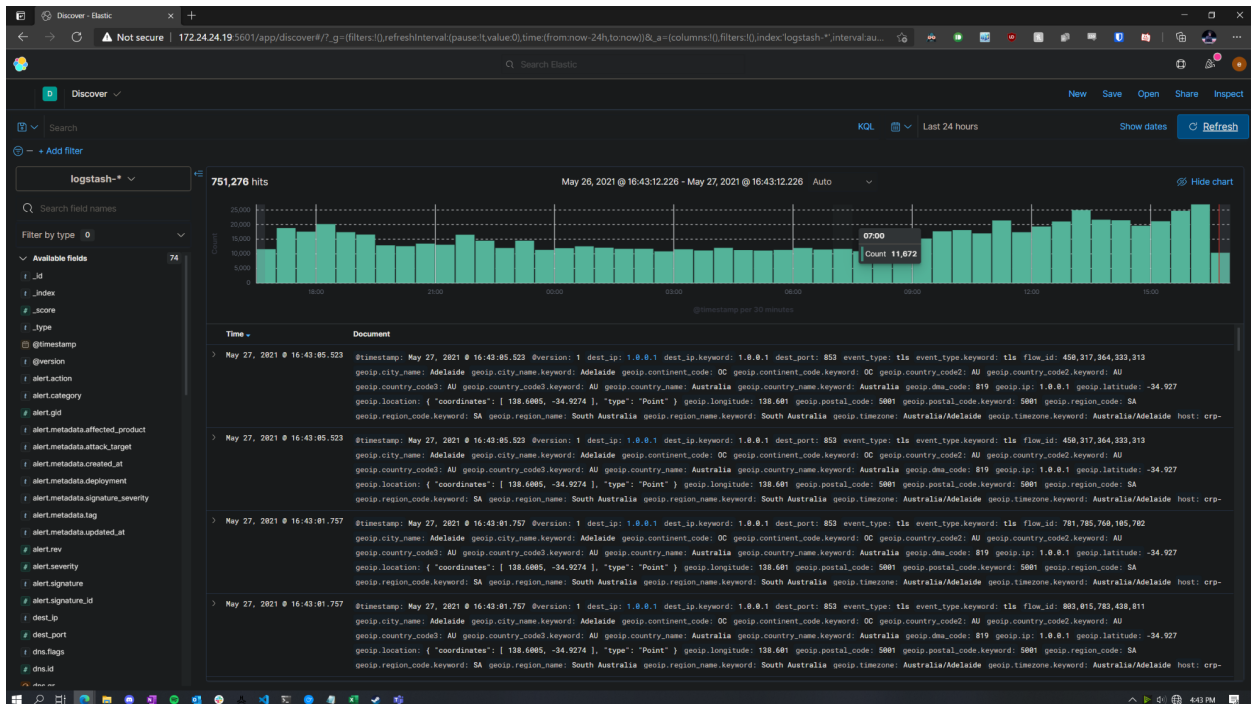
In the ELK stack itself, there were little bugs to be found. Currently, Logstash is slightly out of date causing indices to be labeled incorrectly. Outside the ELK stack, the Redis database used to store network traffic is notorious for having memory problems. Once memory runs out, Redis becomes unstable and the machine crashes.

Product Maintenance

The ELK stack itself is very low maintenance. Functionality works great overall and there are very few bugs to work through. However, the ELK stack is configured to only work with Cyber Range Poulso IP addresses. Any IP address changes will break the ELK stack and it will no longer be in working order. Furthermore, security certificates should be changed from self-signed to using a service such as Let's Encrypt so it will be more secure. In addition, the current ELK stack is using a trial license. It is set to expire on May 31, 2021. The license should be upgraded to Gold.

Regarding users and passwords, a separate file containing all the created users and their passwords has been created and handed over to our client.

Results



Discover page of Kibana, this an overview of all the data coming into Kibana

Users

Create user

Search... Show reserved users

<input type="checkbox"/> User Name ↑	Full Name	Email Address	Roles	Status
<input type="checkbox"/> Erik.Fretheim	Erik Fretheim	Erik.Fretheim@wwu.edu	faculty	
<input type="checkbox"/> apm_system			apm_system	Reserved
<input type="checkbox"/> beats_system			beats_system	Reserved
<input type="checkbox"/> elastic			superuser	Reserved
<input type="checkbox"/> kibana			kibana_system	Reserved Deprecated
<input type="checkbox"/> kibana_system			kibana_system	Reserved
<input type="checkbox"/> logstash_internal	Internal Logstash User		logstash_writer	
<input type="checkbox"/> logstash_system			logstash_system	Reserved
<input type="checkbox"/> logstash_user	Kibana User for Logstash		logstash_reader logstash_admin	
<input type="checkbox"/> remote_monitoring_user			remote_monitoring_collector remote_monitoring_agent	Reserved
<input type="checkbox"/> test.faculty	Test Faculty	testFaculty@email.com	faculty	
<input type="checkbox"/> test.readOnly	Test	ReadOnly	kibana_read_only	
<input type="checkbox"/> vipul	Vipul Kumar	vipul.kumar@wwu.edu	superuser	



Rows per page: 20 < 1 >

Roles

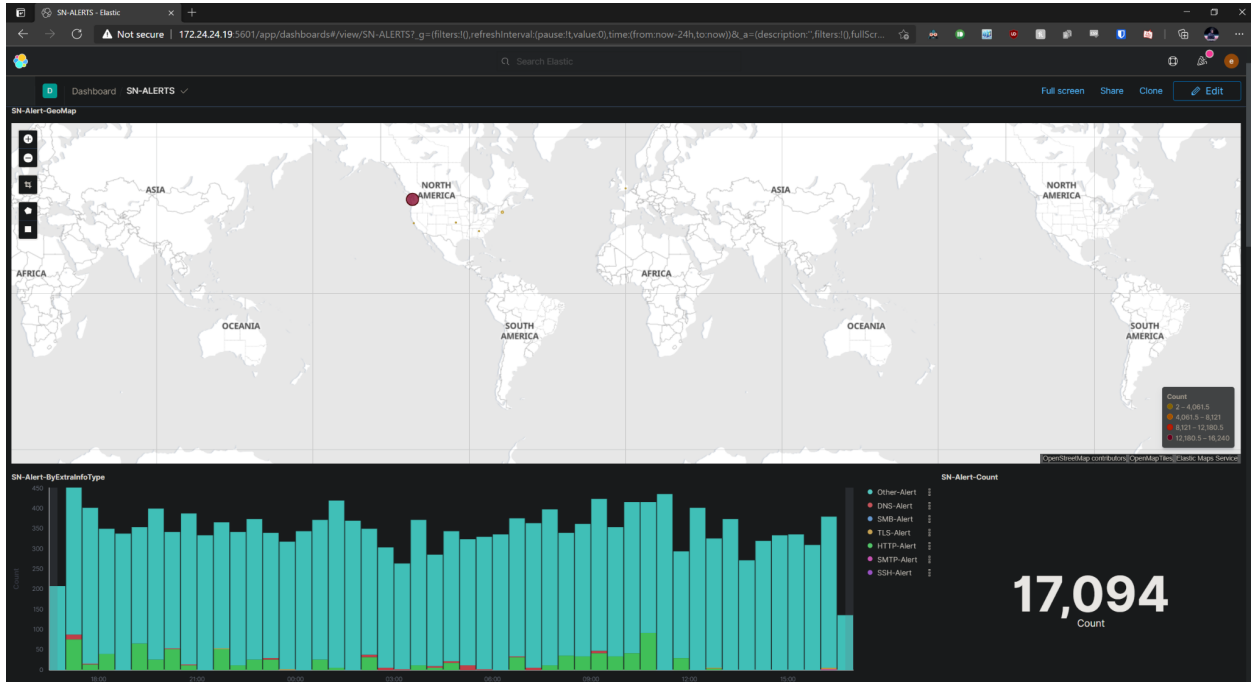
Create role

Apply roles to groups of users and manage permissions across the stack.

Search... Show reserved roles

<input type="checkbox"/> Role ↑	Status	Actions
<input type="checkbox"/> apm_system	Reserved	
<input type="checkbox"/> apm_user	Reserved	
<input type="checkbox"/> beats_admin	Reserved	
<input type="checkbox"/> beats_system	Reserved	
<input type="checkbox"/> data_frame_transforms_admin	Reserved	Deprecated
<input type="checkbox"/> data_frame_transforms_user	Reserved	Deprecated
<input type="checkbox"/> enrich_user	Reserved	
<input type="checkbox"/> faculty		 

These two images cover all the users and some of the roles created for Kibana.



This is one of the dashboards that was created. It shows the incoming data based on their geoiip location.

```

root@debian:/home/student/cyberange-intrusion-detection# python3 test_online.py elastic [redacted]
Checking if es01 is online
es01 is ok
Checking if es02 is online
es02 is ok
Checking if es03 is online
es03 is ok
Checking if kibana is online
kibana is ok

```

This is the output of our script that tests if Elasticsearch and Kibana services are running correctly.

```

root@debian:/home/student/cyberange-intrusion-detection# curl 172.24.24.6:9200/_cat/health
curl: (52) Empty reply from server
root@debian:/home/student/cyberange-intrusion-detection# curl -k -u elastic:[redacted] -XGET 'https://172.24.24.6:9200/_cat/health'
1622159577 23:52:57 crpElasticCluster green 3 3 78 39 0 0 0 - 100.0%

```

This shows the difference between curling to the Elasticsearch cluster with and without security features. Without providing the username and password, the user cannot access the Elasticsearch API.

Documentation

<https://gitlab.cyberangepoulsbo.live/william.pearson/cyberange-intrusion-detection>

Gitlab Cyber Range Poulsbo was used to document the IDS project. It contains a number of resources, including configuration files and pertinent instructions on how to build the ELK stack. Additionally, Elasticsearch and Kibana security certificates for Western Washington University's production environment are also bundled into the repo.

A supplemental guide is added to the repo to further explain the exact steps that we took to build the ELK stack referencing specific instructions from the Elastic guide. A command list was also included to help reference key steps in building a secure ELK stack. These guides will assist future developers in creating new opportunities to extend the project and enhance it.

Future Extensions

Two key extensions to elevate this project are Microsoft Teams alerting and Ansible orchestration.

Currently, the ELK stack is using the trial license where it restricts certain features of the product. In order to enable Microsoft Teams and email alerting, the Elasticsearch license needs to be upgraded to Gold. Please note that Microsoft Teams exhibited peculiar behavior where it will not allow administrators to create incoming webhooks on one team. A separate team needs to be created for the new incoming webhook to work. This bug is specific to Microsoft Teams and is not a problem with Kibana itself. As far as we are aware, creating alerts is a fairly easy process that can be done through the Kibana dashboards.

Ansible is an automation engine used for configuration management, application deployment, and service orchestration. This is the perfect open source tool to orchestrate the IDS and ELK stack in the Cyber Range environment without additional overhead costs and operating expenses. If the project were to ever expand with additional clusters in the ELK stack, it will help with a seamless and efficient deployment. Another note is that we are unfamiliar with this product, so we don't know the complexity of adding it onto our secure ELK stack.