Global Information Assurance Certification Paper

Container-Based Networks: Lowering the TCO of the Modern Cyber Range

GIAC (GCIH) Gold Certification and ISE 5501

Author: Bryan Scarbrough, bryan.scarbrough@gmail.com

Advisor: *David Hoelzer*

Accepted: 7/30/2019

Abstract

The rapid pace and ever-changing environment of cybersecurity make it difficult for companies to find qualified individuals, and for those same individuals to receive the training and experience they need to succeed. Some are fortunate enough to use cyber ranges for training and proficiency testing, but access is often limited to company employees. Limited access to cyber ranges precludes outsiders or newcomers from learning the skills necessary to meet the ever-growing demand for cybersecurity professionals. There have been several open-sourced initiatives such as Japan's Cybersecurity Training and Operation Network Environment (CyTrONE), and the University of Rhode Island's Open Cyber Challenge Platform (OCCP), but they require significant hardware to support. The average security professional needs a cyber range environment that replicates real-world Internet topologies, networks, and services, but operates on affordable equipment.

1. Introduction

Cybersecurity issues are becoming a day-to-day concern for many organizations. Companies are spending more on cybersecurity than ever before with 2019 projections reaching more than $124 billion (Security Team, 2019), and yet data breaches still occur. Amidst this outcry of necessity, companies are still struggling to find and keep trained and qualified cyber-professionals, and are turning to security tools to meet their growing requirements. Tools alone, however, will not fix human behavior. Often improperly installed, configured, and tuned, these tools can provide a false sense of security instead of reliable safeguards. Thus, when the inevitable data breach occurs, companies resort to outsourcing cybersecurity needs to someone else, hoping for practical solutions.

The inability of many companies to entice and retain cyber-professionals (Van Camp, 2019), or to even have the budget to support full-time security personnel, increases the necessity for outsourcing. While many colleges and universities are rapidly developing cybersecurity programs, many of the first graduates are still several years from the workforce. After graduation, many students will continue to require time and experience before acquiring the skills necessary to operate independently within an organization. Thus, cybersecurity professionals need a means of bridging the gap between institutional learning and experience, and the traditional cyber range offers a way to fulfill this requirement.

The concept of a cyber range has been around since about 2006 when companies like Cyberbit and Metova CyberCENTS began offering range platforms to customers. According to Techopedia.com, “A cyber range is a virtual environment that is used for cyberwarfare training and cybertechnology development. It provides tools that help strengthen the stability, security, and performance of cyberinfrastructures and IT systems used by government and military agencies” (Techopedia, 2019). Over the years, these platforms have evolved from on-premise hardware and software solutions to combinations of cloud-hosted and on-premise platforms provided by a host of companies, universities, and government organizations. There have even been free and open source (FOSS) versions such as Japan’s Cybersecurity Training and Operation Network Environment (CyTrONE), and the University of Rhode Island’s Open Cyber Challenge Platform (OCCP). Where these ranges fall short, however, is in both cost and complexity.

Bryan Scarbrough, bryan.scarbrough@gmail.com

Even though CyTrONE and OCCP are free and open-source solutions (FOSS), the hardware required to run them can be quite substantial depending on the network fidelity desired. The hardware requirement is a common issue with cyber ranges where realism is often beneficial to training, skills validation, or systems analysis. For each of these systems to convincingly represent real-world network environments and Internet services, they have traditionally been built using virtual machines (VM). Even though a VM is more portable than individual hardware platforms, it consumes significant resources, and only so much can be done to reduce its CPU, RAM, and hard disk utilization. Using containers, however, opens another world of possibilities given even the most modest hardware and brings the cyber range within the grasp of both the low-budget company and frugal individual.

2. Containers, Clustering, and Container Networking

2.1 Containers

Containers are a method of virtualization that allows applications to share the same Operating System (OS) kernel. [Containers] allow a developer to package up an application with all of the artifacts it needs, such as libraries and other dependencies and ship it out as one package (Shanmugam, 2018). This feature of containers provides a consistent experience for developers and system administrators responsible for moving applications into production quickly and efficiently. When compared to the traditional VM method of portability, containers are more efficient in their use of shared kernel space, reducing the additional overhead of multiple virtual OS kernels operating on a single system (Figure 1).

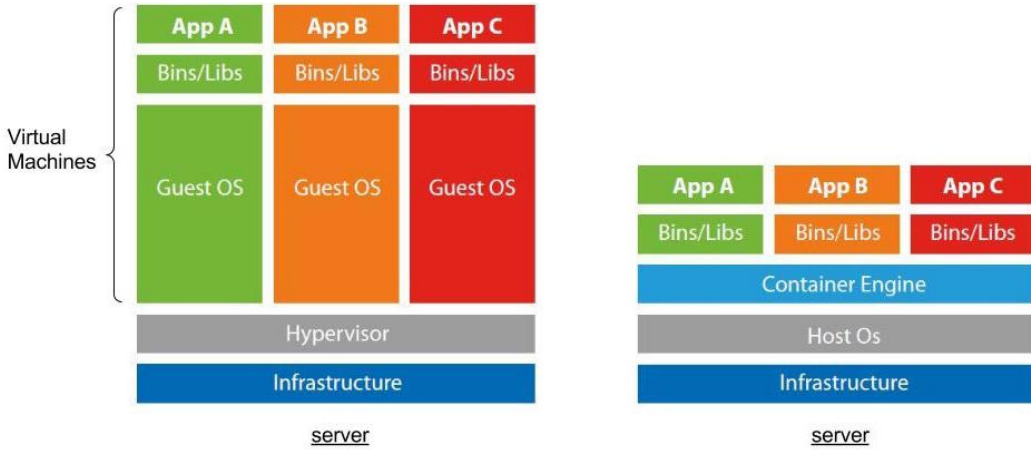

Figure 1: Virtual Machines vs. Containers (Vadgama, 2016)

Bryan Scarbrough, bryan.scarbrough@gmail.com

There are many container runtimes such as Docker, LXC (Linux Containers), rkt (pronounced *rocket*), containerd (pronounced *container-d*), and more. These perform the same type of kernel abstraction providing near-native application performance and portability. All of this is made possible through the use of kernel technologies such as control groups, known as cgroups, and namespace isolation. Both techniques allow containers to be separate from one another in the same manner, just as standard process isolation, network namespaces, mount namespaces, and user namespaces do in modern OS kernels (Wang, 2018). The difference, however, lies in the packaging. This packaging allows containers to be portable from one system to another regardless of the underlying kernel[1]. Due to the abundance of container architectures available, and the many fundamental differences between them, this research will focus, specifically, on the Docker container manager.

While container technology has been around for some time and there are several technologies available for use, the most popular, by far, is Docker. Docker originated as a wrapper around LXC containers, then eventually became a standalone platform. This transition was driven, in part, by a fundamental difference between how Docker and LXC operate. LXC follows the virtual machine model where packages and applications are installed and run in the same fashion as a traditional VM. Thus, LXC natively supports common NodeJS or PHP web applications which require a database backend all running in a single container. Docker, however, expects a separate container for each component of the application. The Docker process manager, which functions similarly to the Linux init or systemd service managers, is the reason for this difference.

When Docker starts a container, it expects an entrypoint which determines what service to start and monitor for container status and operational information. If the entrypoint process stops for any reason, Docker stops or attempts to restart the associated container depending upon the administrator's discretion at deployment. If a container requires multiple processes to function, then a wrapper script must be written to handle the necessary startup and shutdown functionality of each of those services. The wrapper script then becomes the entrypoint for

[1] This is only true of *NIX- based containers. Microsoft "supports process grouping and resource controls (similar to cgroups in Linux)" (Brown, 2017), but they are not portable to other kernel architectures, even having issues being compatible with previous Microsoft OS versions; i.e. a Microsoft container built for Windows Server 2019 builds 17763.* is not compatible with Windows 10 version 1803 builds 17134.* (Lang et al., 2019).

Docker to manage and monitor the container status. An example of a multi-process Docker image is the router container used in this research.

The container-based router used in this research is Cumulus Networks' Free Range Routing (FRR) software. This fork of the Quagga project still operates similarly to its progenitor by running separate processes for each of its various routing functions. Thus, there are individual processes for things like the virtual terminal (VTY), which is used to configure the router, static routing, and each of the various dynamic routing protocols such as BGP, OSPF, RIP, etc. To easily manage all of these daemons, the FRR developers added a custom program called *watchfrr* (Figure 2). According to the code comments, *watchfrr* monitors the frr daemons and attempts to restart them if they become unresponsive. "It determines whether a daemon is up based on whether it can connect to the daemon's VTY Unix stream socket. It then repeatedly sends echo commands over that socket to determine whether the daemon is responsive" ("FRRouting/watchfrr," 2019).

Using the *watchfrr* application, Docker can maintain its built-in process monitoring functionality and run this multi-daemon router without violating its one process per container paradigm.

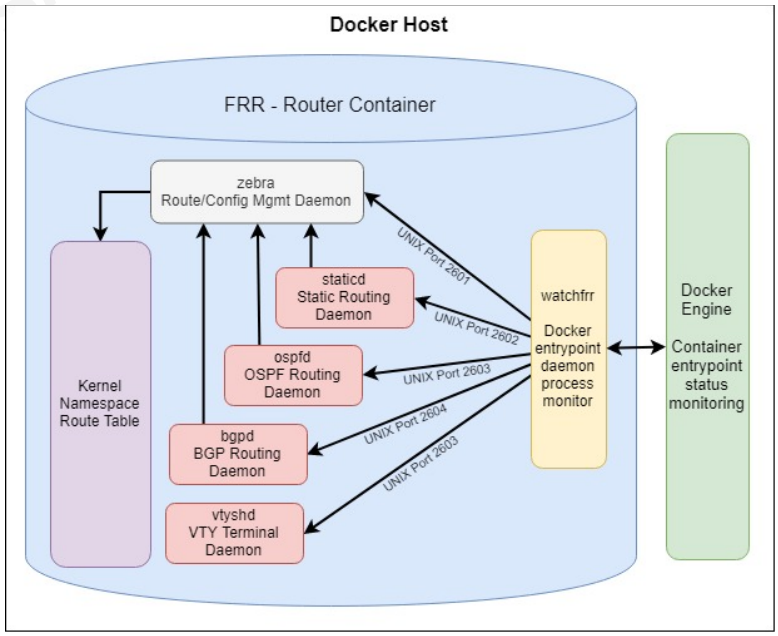

Figure 2: FRR watchfrr - Multi-daemon process monitor for Docker Container management

Container efficiency, portability, and ease of implementation make them a clear benefit to an enterprise. While some aspects, such as the one process per container rule of Docker, may

require developers to view and write their applications differently, it has not slowed their adoption. Another area of consideration in a container deployment model is how containers communicate both with one another and with external networks.

2.2 Docker Container Networking

Docker networking is designed to be relatively simple. It supports five primary models: the standard bridge, user-defined bridge, host networking, Media Access Control Virtual Local Area Network (MACVLAN), and none. Displayed in Figure 3 are the first four models, and the fifth is a standalone container with no network connectivity. The standard bridge network is the default model configured on Docker installation and is used at container runtime if no other network settings are defined.

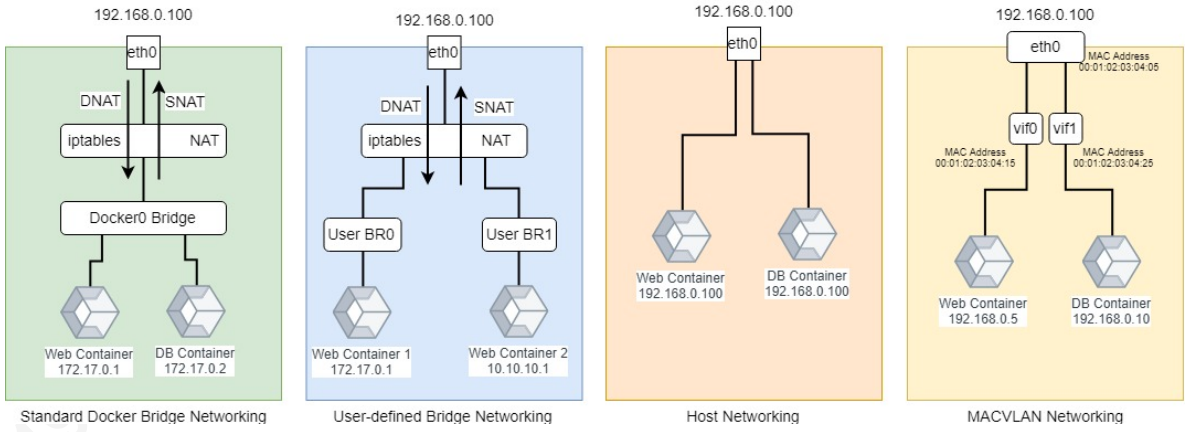

Figure 3: Comparison of Docker Network Models

The standard bridge network is Docker's default configuration (Figure 3). This network is visible on the system as the *docker0* interface and abstracts layer-3 routing from the user. The standard bridge uses a pre-configured network for container-to-container communication and Dynamic Host Configuration Protocol (DHCP) for IP allocation. Next, Source-based Network Address Translation (SNAT) and Destination-based Network Address Translation (DNAT) in the host-based firewall, known as iptables by default on Linux systems, manage container connections to the rest of the network. Containers connect to external systems through SNAT using a shared host IP address in the same manner that a home router shares a single public IP address with all of the devices connected to the network. To expose a container service to the local network, Docker uses DNAT to map host ports to container ports, which is similar to port

forwarding on a home router. The standard bridge network simplifies basic container network connectivity, but sometimes the user requires a more customized configuration.

User-defined bridge networks, signified by the blue block in Figure 3, grant the user more granular control over the network configuration. At creation, the user specifies the subnet parameters for the network and the bridge interface name and Docker manages building the required bridging, firewalling, and DHCP pool. This network functions similar to the standard bridge by using SNAT and DNAT for container network access but differs by isolating containers on one bridge from containers on another bridge. Thus, any containers connected to the same user-defined bridge are in the same network and can communicate directly with one another. However, containers connected to one user-defined bridge will not be able to communicate with containers connected to a different bridge interface due to the NAT configuration. Using NAT to connect containers to the network and to segregate containers from one another can be a useful deployment method, but this may not be suitable for situations where the container requires direct host network access.

In Docker host networking, all containers share the host's default network namespace (Figure 3). The container shares the same physical network interfaces and configurations as the host; it shares the host's IP, and directly exposes any daemon ports on the host. This model is similar to installing server software locally but without the hassle of configuring the environment, letting the administrator run the pre-configured container without concern for particular dependencies. Since the container shares the host network namespace, its ports and protocols are shared directly by the host system. Thus, a web server running with host networking has the same IP as the host and opens port 80 on the host by default.

Finally, Docker uses the kernel's built-in MACVLAN driver for networking indicated by the yellow block in Figure 3. MACVLAN assigns a MAC address to each container, and subsequently a unique IP address. It is similar to host networking because containers share the host's default network namespace. It differs, though, in that the containers do not share the host's IP or MAC address. The kernel divides the host interface into virtual sub-interfaces that are assigned a unique MAC address and then attached to the designated container. Due to the way the OS kernel handles MACVLAN interfaces, the host cannot communicate directly with the container's virtual interface over the network. Thus, MACVLAN interfaces are useful in

situations where an application requires direct network connectivity, but the host requires a degree of isolation from the container. Service providers often support this method when offering container-based services, yet require segregation between those services and their underlying hosts.

Docker provides multiple means of connecting containers to the physical network, and the ones listed here are merely the most common approaches. The flexibility offered by the various Docker network models grants a level of diversity that prepares environments to grow and expand along with the organization. With growth comes increased complexity, and with increased complexity arises the need for automation and orchestration. As container environments swell and expand the purview of the datacenter, continued availability, performance, and scale require some form of clustering and orchestration.

2.3 Container Clustering with Kubernetes

Given the light-weight portability of containers, developers quickly understood their inherent advantages and sought methods of scalability across the datacenter. The result is a series of cluster orchestration platforms such as Kubernetes, Docker Swarm, Apache Mesos, OpenShift, and many more. These platforms act similarly to services such as VMware's vCenter server which abstracts low-level operations and management from the administrator, focusing on optimization of computing resources. Each platform offers on-premise, or cloud-hosted orchestration of containers and centrally manages scale, replication, fault tolerance, and high availability, granting a high level of control.

This research will focus on the Kubernetes management engine using Docker containers to virtualize the cyber range environment. The original intent was to use Docker Swarm because it is simpler to use and implement, and more light-weight than other cluster managers. However, at the time of this writing, Swarm does not support privileged containers (containers that run with host OS privileged access). While privileged containers are not preferred because they grant the container direct access to host resources such as network and disk as the host's root user, running router containers requires privileged access to the kernel route table for the container's associated network namespace. Thus, the research focuses on running these containers using Google's Kubernetes cluster manager.

According to the official documentation, “Kubernetes is a portable, extensible open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation” ("What is Kubernetes," n.d.). A declarative configuration abstracts the underlying flow or logic of a configuration action and instead expects a declaration of the desired state or outcome. Thus, the configuration files define the desired state of an application, and Kubernetes manages the declared state. It does this regardless of container architecture allowing developers and administrators to focus more on managing, developing, and deploying applications and less on the underlying architecture.

Kubernetes (K8s) uses a resource called a master to abstract functionality away from the administrator and use a common language across platforms. The master acts as the cluster controller containing the API server, resource scheduler, and controller manager services (Figure 4). K8s administrators interface with the master to coordinate cluster activities, and to deploy, test, scale, and troubleshoot applications and their states. Masters control cluster operations and scheduling using resources called nodes[2], also known as workers or minions.

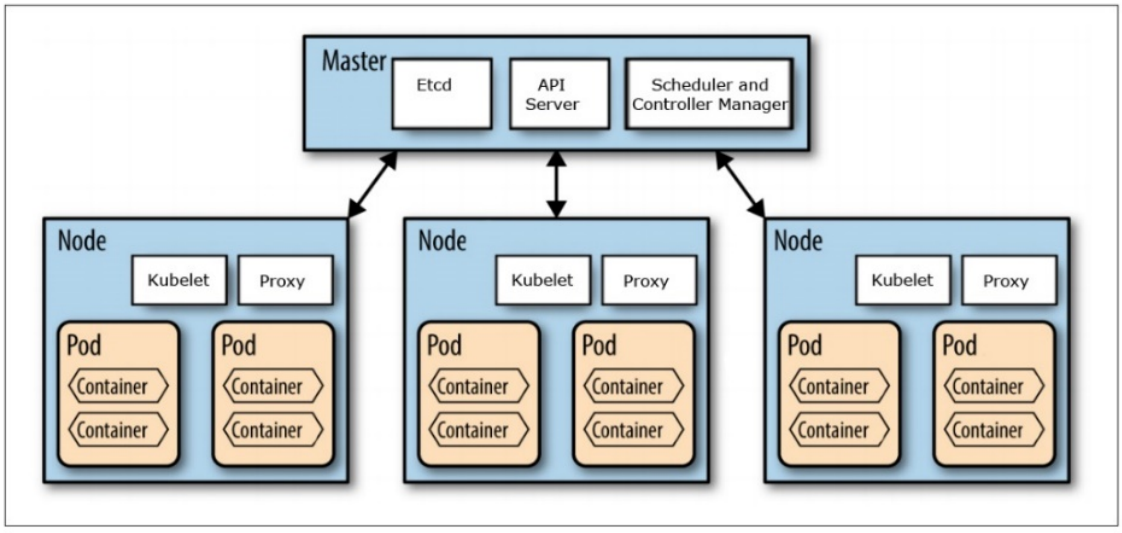

Figure 4: Kubernetes Cluster Architecture (Larsson, 2017)

The node is the compute resource for a cluster providing CPU, RAM, network, and storage on which to run a Pod, which is a container or a group of containers with tightly integrated functionality. Nodes communicate directly with the master’s API through the kubelet

[2] There are instances where the master is allowed to act as a node, in single system configurations such as Minikube, but it is recommended to use these for testing or development and not production.

process (Figure 4), which acts as an arbitrator between the master controller and the node's underlying container manager. Using the kubelet process feedback loop, the master ensures state compliance for all currently running Pods and attempts to balance cluster resource utilization for any newly scheduled Pods.

The most common Pod implementation is the "one-container-per-Pod" configuration (Figure 5). This employment is the smallest schedulable resource in a cluster and is the basic building block of a K8s configuration. It functions similar to running a single container in Docker but is managed and monitored by the master. While the "one-container-per-Pod" model is the most common, there are situations where multiple containers must coexist within the same Pod.

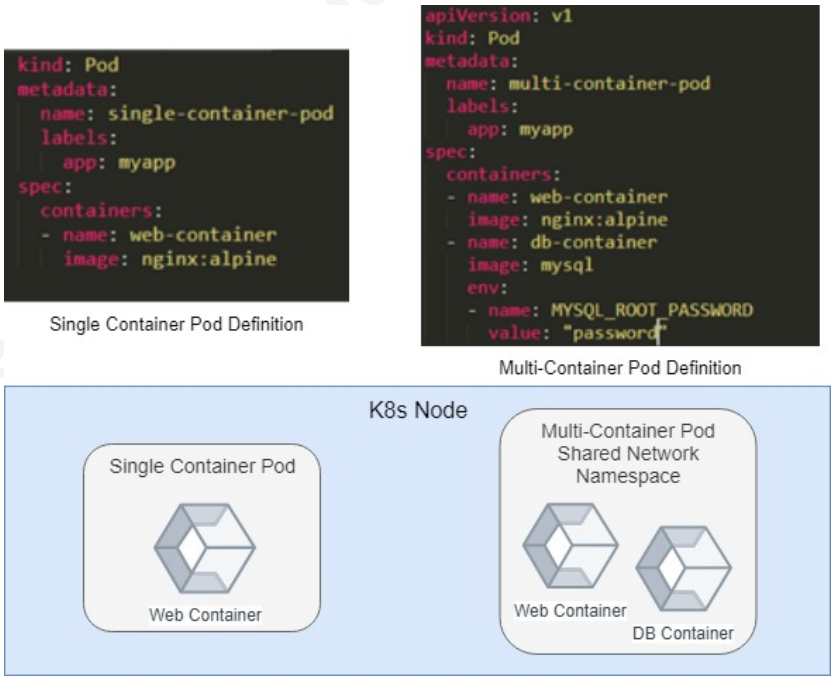

Figure 5: Single Container vs. Multi-Container Pod Deployment Examples

Any application where multiple components work together to serve a single function could be a multi-container Pod (Figure 5). Multi-container Pods are necessary for some applications which require helper programs, such as content management systems, proxies, or log and checkpoint backup packages. This integration allows K8s to manage the operational state of the group of containers as a single object. Doing so ensures that when one container malfunctions, the entire application is unable to consume additional system resources and is inaccessible to an end-user. It also ensures that all the containers in a given Pod run on the same

node, and all resources consumed by the Pod, including storage, or network connections, exist for the same lifetime. Since Pods are low-level implementations of containers, they have very little in the way of state management or monitoring functionality.

One of Kubernetes' advantages is its declarative management of an application's state across a cluster, and Pods by themselves are stateless. Thus, K8s documentation recommends managing Pods through higher-level schemas such as ReplicaSets, Deployments, StatefulSets, or DaemonSets. For this research, Deployments and DaemonSets are discussed and used in this configuration.

At its simplest, a Deployment describes a set of identical Pods (Figure 6). While the Pod is the smallest object in Kubernetes, the Deployment abstracts the Pod's low-level management and defines the number of Pods and the desired state of the application. The Deployment can monitor status, health, and liveness of an application and if a Pod or container within a Pod fails, the scheduler can automatically restart it to maintain the Deployment's state. In this way, the administrator does not have to be concerned with individual Pods but can manage the application itself and focus more on its stability and availability.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

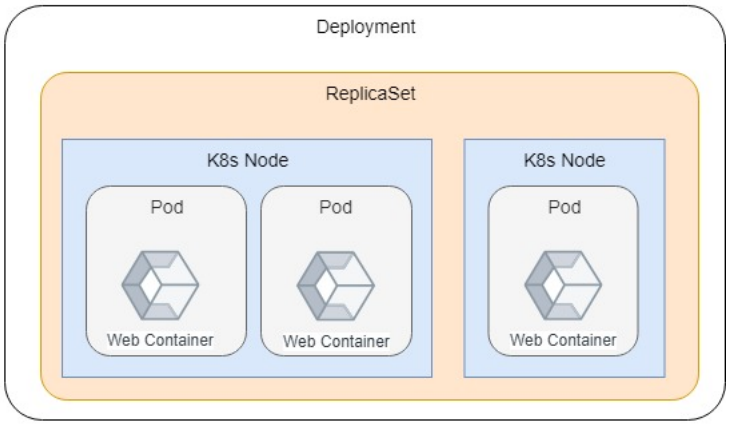

Figure 6: Kubernetes Deployment Specification Example

The DaemonSet specification declares the acceptable pod-to-node relationship for services (Figure 7). The declarative state of a DaemonSet ensures each designated cluster node runs an identical copy of the Pod. DaemonSets are standard with entities such as Container Network Interfaces (CNI), container-based storage like GlusterFS, or log mechanisms like Filebeat where an administrator may benefit from running a pod instance on each node in a cluster to access local log files.

Bryan Scarbrough, bryan.scarbrough@gmail.com

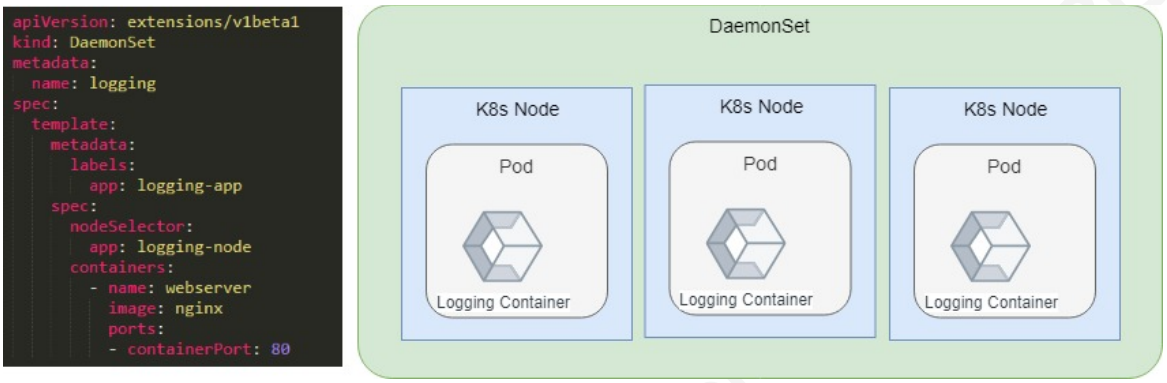

Figure 7: Kubernetes DaemonSet Specification Example

2.4 Kubernetes Overlay Networking

Kubernetes strives to abstract networking from the administrator in the same manner as other underlying container management mechanisms. It manages networking through the use of the Container Network Interface (CNI) which is a DaemonSet that governs Pod-to-Pod communication on each node. Each CNI strives to abide by Kubernetes' two networking tenets: Pods should be able to communicate with each other without NAT, regardless of the node; and, any node's system daemons should be able to natively interact with all of the Pods running on that node ("Cluster Networking," n.d.). To accomplish the former, K8s introduces the concept of the overlay network (Figure 8) to container communications.

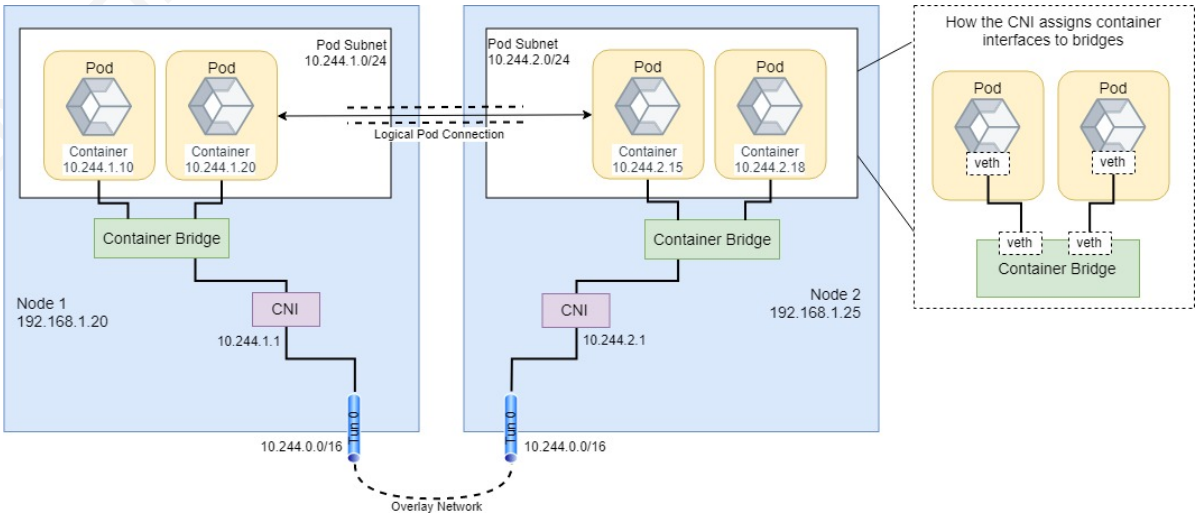

Figure 8: Kubernetes CNIs and Overlay Networking

Overlay networks interconnect disparate network environments transparent to the network underlay, which includes the routers and switches making up end-to-end transport.

Overlay networks allow two systems connected across it to appear as though they are on the same Local Area Network (LAN), and in some cases, even the same Layer-2 switched network. This type of connectivity is possible using several technologies such as Multiprotocol Label Switching (MPLS), Virtual Extensible LAN (VXLAN), or Generic Routing Encapsulation (GRE), to name a few. The most often used in container networking are GRE and VXLAN, and VXLAN is the referenced overlay network described in this research unless otherwise specified.

Kubernetes combines all of these concepts from container orchestration to high-level declarative deployment definitions, and overlay networks to provide a service discovery model that is shaping the cloud landscape. K8s shines in Layer-4 to Layer-7 service discovery and mapping, allowing administrators to easily manage applications and integrate load balancing, replication, and service presentation in a transparent way. Presenting applications in this way is perfect for Infrastructure-as-a-Service (IaaS) providers like Amazon Web Services (AWS), and Google Compute Engine (GCE) who wish to offer a simple means of offloading compute and storage for customer applications. Where it becomes much more challenging, however, is when the intent is to use K8s clustering capabilities to build and deploy a realistic underlay network.

3. System Design Hurdles for a Containerized Cyber Range

3.1 Overcoming Container Multi-Interface Concerns

Due to its service-oriented design model, Kubernetes excels in high-level service presentation and discovery, but working through its abstractions to build realistic underlay networks presents a series of challenges. Arbitrary IP addressing and multiple interfaces per Pod are among those challenges. Typically, Kubernetes' CNIs manage the overlay network requirements and automatically assign IP addresses, handle routing, DNS, and switching for the Pods. The network architect, however, must be able to configure all of these settings when designing and building underlay networks, and K8s does not natively support multiple network interfaces per Pod. This research chose to use the Flannel CNI to meet K8s primary networking tenants, and Intel's Multus CNI ("intel/multus-cni," 2019), OpenVswitch, and Kubevirt's OpenVswitch CNI (OVS-CNI) ("kubevirt/ovs-cni," 2019) for the container range Pod integration.

The primary CNI used for this deployment is CoreOS' (now Red Hat) Flannel CNI. Flannel is a simple, light-weight, VXLAN-based overlay network that satisfies the standard networking requirements of Kubernetes by allowing all Pods to communicate with one another regardless of the node, and allowing node resources to communicate with running pods. The purpose of this CNI is purely for Pod management and exposing services for the logging, analysis, and management of containers. It also opens the internal Kibana server for administration and interfacing with the Bro IDS log data captured on the internal OpenVswitch underlay network interfaces.

OpenVswitch (OVS) is a program that enables configurable Layer-2 switching for Linux hosts. It is a mature project backed by the Linux Foundation and companies such as Red Hat and is used extensively in the OpenStack virtualization platform. This software has robust support for underlay and overlay technologies, allowing Virtual LAN segmentation of the host if desired, and VXLAN overlay networking. This research implements OVS to create the desired Layer-2 underlay infrastructure and uses its VXLAN capability for Pod-to-Pod communication across nodes. Integrated with Kubevirt's OVS-CNI, which allows Pods to connect to OVS bridges, these features make it the ideal platform for building arbitrary underlay networks.

Configurability and manageability are OVS' primary advantages. Using a package like OVS gives a network administrator low-level control over the Layer-2 architecture bringing a degree of familiarity to Software Defined Networking (SDN). OVS supports many standard Layer-2 protocols and Layer-3 management protocols granting fine-grained control over network device interconnection similar to physical networks. After building the Layer-2 underlay network and any required overlay, the OVS-CNI abstracts all of those details from the K8s administrator. This abstraction facilitates separation of duties so that a single administrator does not need to be both a network engineer and a container infrastructure manager. The integration of OVS and OVS-CNI, however, does add a level of complication to network design.

The choice of OVS introduces complexity to network design by making virtual switch creation and Kubernetes integration a two-step process. The first step requires creating the same named bridge interface on every K8s node, then adding the VXLAN overlay to each bridge. The second step creates the K8s network definition used by Pods to connect to the OVS bridge. This two-step process introduces the potential for errors in the creation, configuration, and integration

of each bridge interface. Management through scripting or automation frameworks such as Ansible is possible, but it is still an additional step requiring execution, monitoring, and troubleshooting outside of a pure K8s deployment. With the K8s' networking requirements and Layer-2 underlay requirements met through Flannel, OVS, and OVS-CNI, the next concern is assigning multiple interfaces to a single Pod, and this is where the Multus CNI is useful.

Multus is a Kubernetes CNI plugin that attaches multiple interfaces to a single Pod. This plugin sits across other CNIs handling interconnection and allowing Pods to simultaneously connect to different CNIs, or even connect multiple times to a single CNI if that plugin supports such a configuration. One of the advantages of the OVS-CNI is that it readily supports more than one interface for a single Pod. Thus, using Multus, each Pod is allowed to have a Flannel CNI connection and satisfy the default network requirements for K8s, and it is also assigned all of the OVS-CNI network interfaces necessary for building the underlay container-based range network. Multus resolved the Pod multi-interface concern, but using OVS-CNI created issues surrounding the VXLAN overlay network.

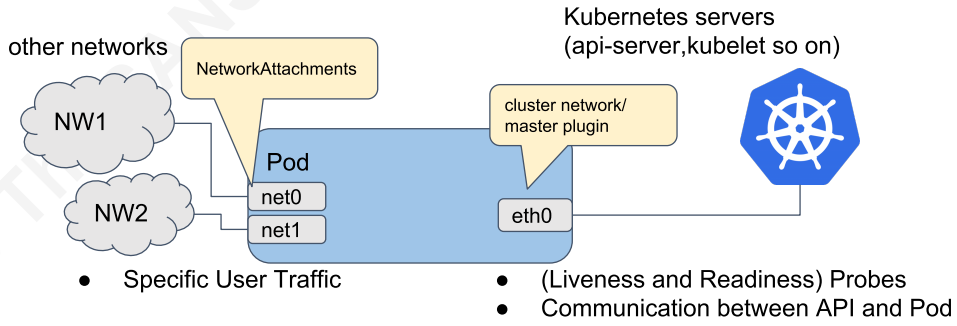

Figure 9: Multus CNI Multi-Homed Pod Connections ("intel/multus-cni," 2019)

VXLAN is a Layer-4 technology that encapsulates each frame with a 50-byte UDP header. This header includes the VXLAN Tunnel End Point (VTEP) IP address, referred to as the Outer IP in Figure 10. The Outer UDP block represents the VXLAN port, which is UDP port 4789 by default. Next, the VXLAN Header block contains the VXLAN Network Identifier (VNI), which functions similarly to a VLAN ID for standard Layer-2 networks. Finally, each block is overlaid onto the original ethernet frame to extend the network's Layer-2 connection across the underlying network infrastructure and adding 50-bytes to the original ethernet frame size. While 50-bytes may not seem like a significant size increase, unless it is compensated for in the configuration, it can wreak havoc on the rest of the network.

Bryan Scarbrough, bryan.scarbrough@gmail.com

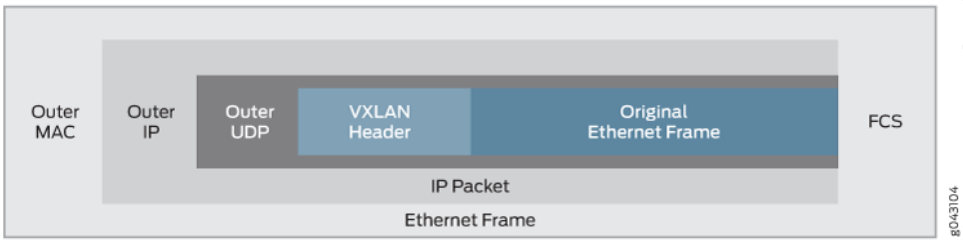

Figure 10: VXLAN Packet Format ("Understanding VXLANs," n.d.)

Using an overlay network simplifies connectivity between Pods and separates the range's underlay network from the environment's underlay network, but it only works by compensating for the increased frame size. OVS-CNI manages Pod connections to OVS bridges, but it is unaware of user-defined configuration settings such as VXLAN or VLAN identifiers and is unable to configure those settings itself. Without compensating for the increased frame size on the underlay network, fragmentation occurs, rendering TCP connection attempts useless across the range network. To resolve this issue, the user must reduce the maximum transmission unit (MTU) by 50-bytes to balance each frame's size on the network. The MTU issue does not affect external connections because Layer-3 endpoints negotiate MTU. Thus, external systems connect to the container-based network and mediate an adjusted MTU with the nearest Layer-3 device to communicate with the rest of the range environment. However, since OVS-CNI is unaware of the VXLAN implementation manual container MTU adjustment falls to the administrator.

Resolving the issues surrounding multiple interfaces, underlay creation, overlay management, and MTU were only the first steps in the range design process. Using Multus CNI, OVS and the OVS-CNI, and Flannel in concert satisfies the K8s network requirements and provides the granularity of control of both underlay and overlay environments necessary to build the range. OVS-CNI's support of multiple interfaces per Pod also assures that router containers can support the number of links needed to design complex network environments. Even though there are now many interfaces per Pod, there is still the concern of custom IP addressing necessary to design and build the range infrastructure.

3.2 Custom Container IP Addressing

Since the intent of this design is to create underlay network environments on top of Kubernetes, the network needs to be configurable and customizable. Network customization is not a normal part of container networking and is counter-intuitive to a standard K8s design

where Pods are automatically assigned addresses through the use of an IP Address Management (IPAM) tool, typically a part of the CNI. Overcoming this design limitation required customizations to each container to make declarative IP address assignment and network configuration possible. This research used a custom Docker entrypoint script, and environment variables to assign IP addresses.

As discussed previously, Docker's containers rely on entrypoint scripts or processes that run on startup to manage any underlying container operations. Docker, then, monitors container health and status based on that entrypoint or process. Thus, to configure each Pod's secondary interface IP, a custom Linux Shell script sets the IP and executes the primary container process. This new script uses environment variables to set the IP address values and is copied into each container to run as the default entrypoint. Using this method, the K8s Deployment can remain fully declarative, easing readability of deployments, templating, and manageability of the environment. Once IPs are assigned, and the container-based underlay network is functional, the challenge becomes integrating external systems.

Working through the multiple interface-per-Pod concern and arbitrary IP addressing concerns were among the most difficult challenges when building this environment. Resolving each only seemed to lead to additional problems culminating in the manual MTU assignment, which mitigated the last of the interconnectivity concerns. Once established, throughput and stress tests were performed on the network to ensure it could support multiple services and multiple users accessing resources simultaneously (discussed in Section 5). The last component of range operations and troubleshooting focused on physical-to-virtual interconnection.

3.3 Physical-to-Virtual Interconnection

Connecting external systems into a cyber range or lab environment is a vital function that enables users to extend existing capabilities and enables the range to meet changing requirements. External systems can be anything from virtual machines running on the same or a different hypervisor, physical systems, or even other containers. Several potential pitfalls arise when working through interconnection scenarios due to factors such as MAC address changes, routing concerns, and physical- to- container interface mapping.

The standard method of overcoming these limitations is to modify the hypervisor configuration ports to a less secure state that allows network traffic to flow. The changes require enabling promiscuous mode and forged transmits, on the VMware hypervisor, and promiscuous mode on Xenserver and Microsoft Hyper-V. Promiscuous mode creates a potential security concern by allowing traffic originating from an unknown system to traverse the network or to capture network traffic. This same method is used for Intrusion Detection Systems (IDS) to intercept packets not explicitly addressed to them. This research decided to integrate host local network namespaces with the existing container network to manage the connections and external routing into the environment.

A separate network namespace functions similarly to Virtual Routing and Forwarding (VRF) by segmenting kernel networking and managing IP addresses and routes independently from other namespaces. The range routes external connections through the virtual machine's ethernet port and all physical and virtual underlay systems do not have issues with unknown MAC address changes (Figure 11). This method also creates additional routing possibilities not only by resolving the security concerns of the underlay network but also by adding more hops inline in the network for packet processing and setting the stage for custom routing to other environments.

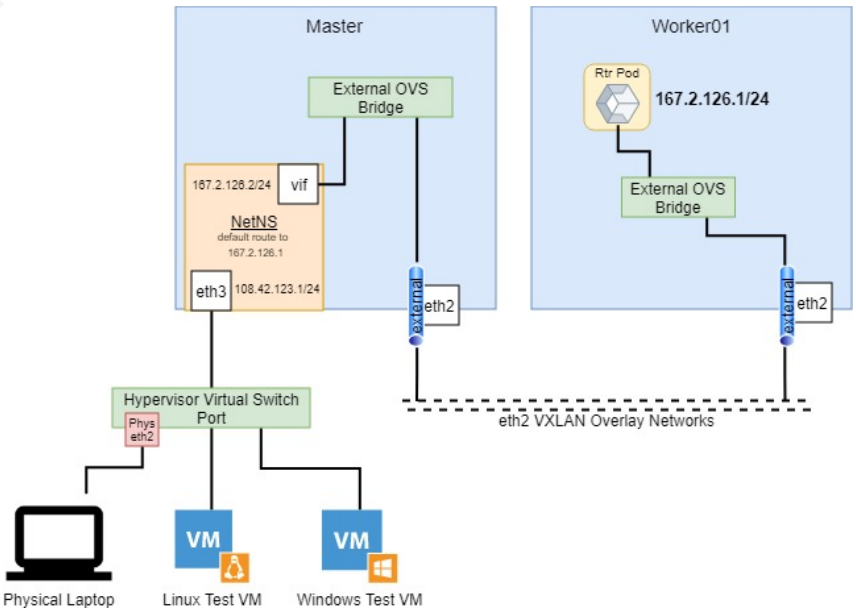

Figure 11: Network Namespace for External to Internal Routing

4. Understanding the Environment

4.1 Network Fidelity Considerations

As with any project, determining the correct scope is critical to ensuring success, and building a containerized cyber range is no different. The purpose of a cyber range is to test, validate, evaluate, or strengthen the user's skillset in a controlled environment. Based upon the end user's current skill level, however, there is a variety of capabilities that are required to fulfill any one of those goals. This research focused on three main objectives to support the range's intended purpose; a small-scale Internet using real-world IPs and protocols, make the network ephemeral, and give the user control of and visibility into the environment.

Routed networks form the backbone of any range, and this system attempts to balance realism and simplicity in the design of its routed network. The range consists of seven routers representing the dominant geographic areas of North America, Europe, Latin America, Oceania, Asia, and Africa. The routers use the Border Gateway Protocol (BGP) to exchange routing information because is the backbone protocol of the Internet, so it scales to a large number of routes allowing the network to grow if the user desires. BGP is also a point-to-point protocol which simplifies the OpenVswitch configuration requiring only one bridge interface instead of separate interfaces for each router link as would be necessary for multicast-based protocols such as Open Shortest Path First (OSPF). Finally, since the router interconnection only requires a single bridge interface, a port mirror enables Bro IDS logging so that the user can analyze all data traversing the network.

After selecting the routing protocol and interconnection design, the focus became replicating geographically relevant device configurations based on the areas represented. Since each router represents a geographic region, IP addressing follows Regional Internet Registry (RIR) assignments based on class-A IP reservations. Furthermore, BGP Autonomous System Numbers (ASN) also display loose geographic fidelity. Geospatial representation of the network using tools like Splunk or Elasticsearch drove the decision to use real-world addressing. Thus, the user can emulate events originating from different areas of the world and use standard monitoring tools to visualize the activities.

Lastly, transience is critical to reducing the cost of destruction and recreation of a range. Anyone who has spent any time in cybersecurity knows that part of the job description includes breaking things. There are few things in life more painful than spending hours, days, or even weeks on a project only to watch it fall apart after you, or someone else broke it to pieces. Thus, the cyber range must be able to easily roll back to a known good state without much difficulty. A simple, short-lived range is the best way of supporting training or skills validation by adding repeatability and sustainability. Currently, VM snapshots are the easiest method for managing this capability until a fully automated build process can be derived.

4.2 Service Fidelity Considerations

Several standard Internet services were layered onto the network for a realistic look and feel and to increase the potential attack surface (Figure 12). The services include DNS, HTTP(S), NTP, SMTP, FTP, a PKI Certificate Authority, and even a couple of intentionally vulnerable services like the Metasploit vulnerable service emulator, OWASP's Mutillidae, and a vulnerable Wordpress server. As with the routed network, each of these services represents a real-world service, using real-world IP addressing. For example, Google's public DNS service has become very popular, and many people use 8.8.8.8 as their DNS service of choice. To keep things simple, that same recursive DNS server IP address is used inside the range along with Quad9's 9.9.9.9.

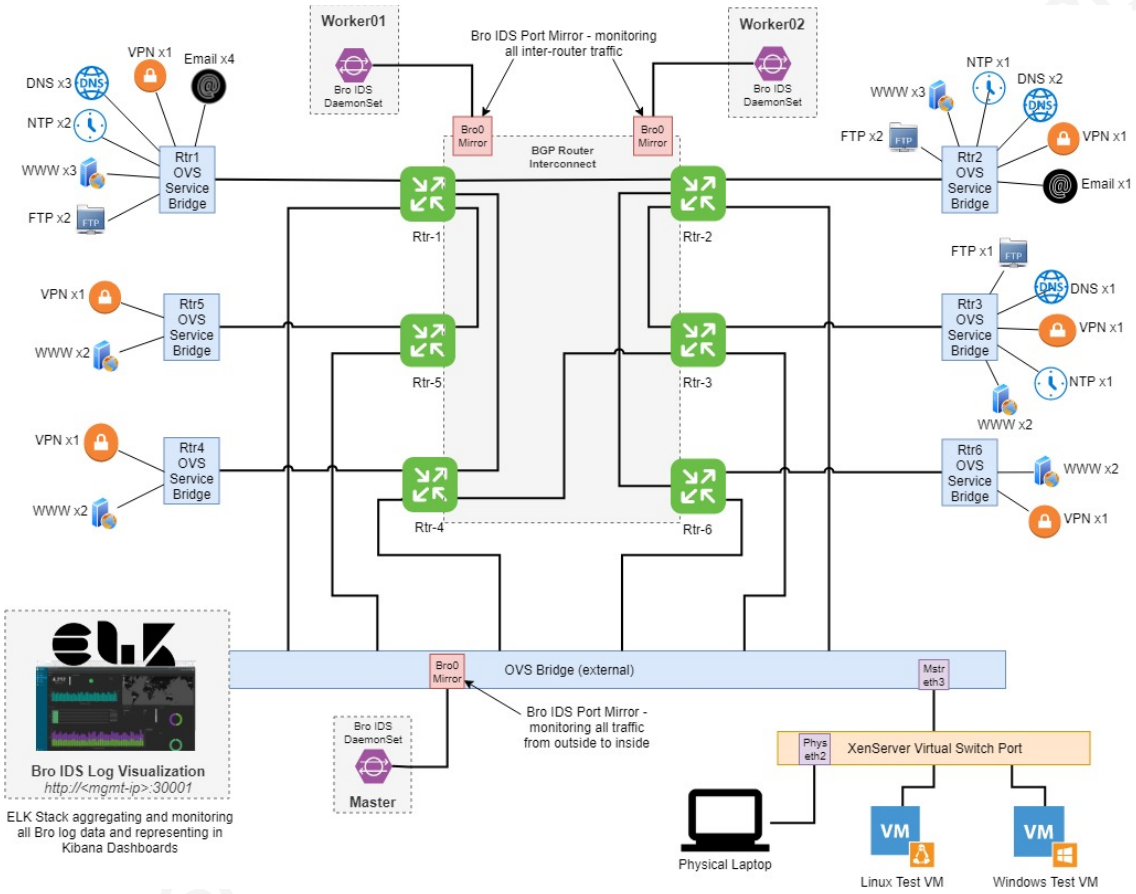

Figure 12: Cyber Range – Network Topology

A cyber range is not complete without services useful in training or learning. For instance, each router has a pre-configured OpenVPN server attached to it. OpenVPN adds intriguing possibilities by allowing the user to connect to a remote VPN using geographically relevant IP addressing, and analyze the network traffic from beginning to end. This scenario could help the user understand how tunneling works, learn how encrypted packets look traversing a network, or what information is evident about particular encryption algorithms by capturing the data and analyzing it using various tools.

Finally, on the service side, there are multiple Metasploit and Kali Linux machines located throughout the network and an integrated Bro IDS for logging. With these tools, the user does not need to install a local instance of Kali or Metasploit unless they require tools beyond the Kali Linux Top 10 metapackage. Users can connect to the consoles of these systems and initiate attacks or scans across the environment. Using the integrated logging is possible with a pre-configured Bro IDS sensor along with an Elasticsearch, Logstash, Kibana (ELK) stack for full

visibility inside the environment. Figure 13 shows a packet's path through the network and indicates the various log capture stages for attack traffic analysis. With this information, the user can capture data as it traverses the network analyzing not only the outcome of the event but also the log data generated leading up to the scan or attack.

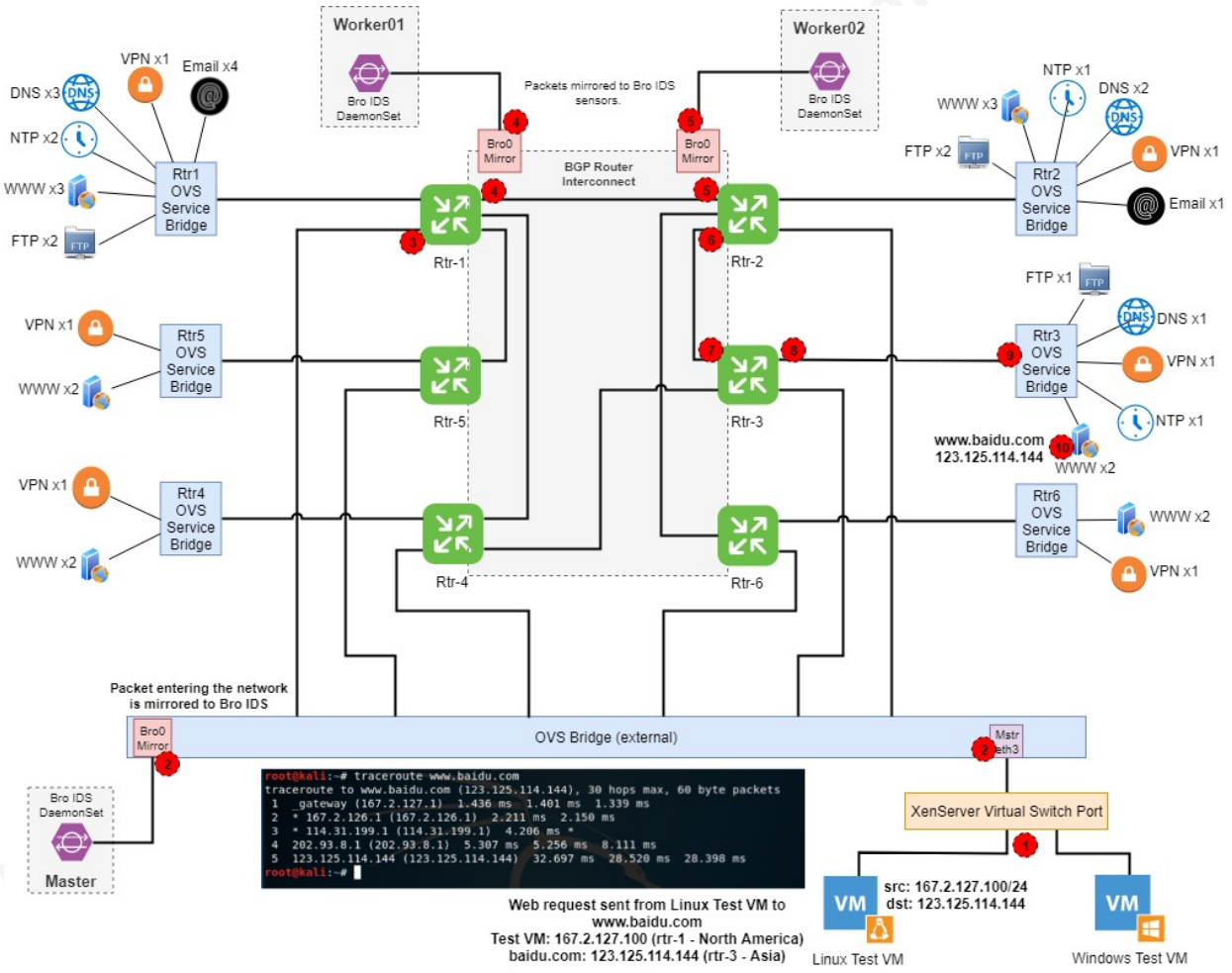

Figure 13: Cyber Range – Network Traffic Flow and Logging

4.3 Management Considerations

Kubernetes is known for being difficult to build, configure, and manage; a tool called Rancher helps separate the user from this underlying complexity. Rancher Labs (rancher.com) builds a software product called Rancher, which is a web-based management platform for Kubernetes or other container orchestrators. To help keep track of cluster operations and health, Rancher also has a built-in means of deploying additional logging utilities such as Prometheus and Grafana. For testing and analysis of this environment and a more sustained logging

mechanism, this research set up these tools to help visualize performance and scalability[3]. With this software, a user can manage and monitor the K8s cluster without the burden of learning all the commands. It keeps the focus of the range on learning and refining cybersecurity skills, not on maintaining a complex Kubernetes cluster or managing the range's complicated network infrastructure.

5. Analysis and Testing

5.1 Network Throughput Testing

Initial throughput tests of the range used iPerf3 and no services other than the container routers to reduce memory and CPU utilization and to evaluate network performance accurately. The first discovery is the maximum throughput of the underlay network is approximately 200 MB/s with routers distributed equally across the cluster. 200 MB/s may not seem like a very robust network, but the combined network processing of the range peaked at around 3.7 GB/s due to additional logging and processing overhead. The disparity between the underlay and the total cluster processing is due to the replication of network traffic from each of the CNIs and the number of interfaces, both physical and virtual, processing each packet.

Based on the way the underlay network functions within the cluster, multiple interfaces and Pods process a single packet. Figure 14 indicates that the physical interface, the Multus and OVS CNIs, the container bridge interface, then the Pod handle each packet. Thus, each time a packet moves from node to node, its processing requirement increases five-fold, and from Pod to Pod on a single node, it doubles. Following a packet through the network from Pod one to Pod two, then to Pod three, 200 MB/s of data generates 2.6 GB/s across the nodes (Figure 14). Add to this the iPerf3 Pods used in testing, and there is an additional 400 MB/s per node of data being processed (200 MB/s by the iPerf3 Pod and another 200 MB/s by the container bridge). The additional data raises the total throughput to 3.4 GB/s on the overlay network, while the underlay network processes 200 MB/s. To further exacerbate the issue, the way that Prometheus' node exporters log, they send network metadata from each packet such that if five separate entities process a packet, then the node exporter will also handle the same metadata five times. Grafana

[3] Due to the logging resource utilization overhead, it is not recommended to install Prometheus and Grafana unless there is a specific reason to do so.

log details indicated the Prometheus node exporter generated an additional 300 MB/s of data bringing the total to 3.7 GB/s processed by the overlay network for 200 MB/s on the underlay network.

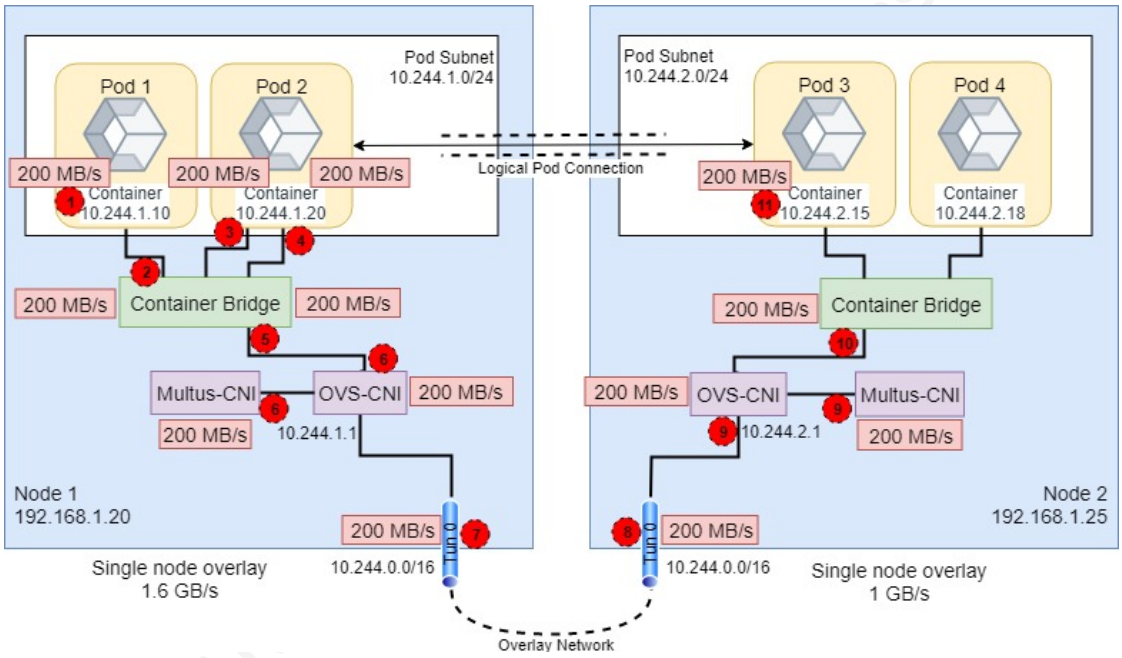

Figure 14: Multi-Hop Packet Processing for the Overlay Network

Kubernetes' packet processing in a multi-hop configuration reveals two essential facts about a containerized range. The first is that the standard Kubernetes logging mechanisms, while helpful in monitoring cluster health and status, do not scale very well for an environment such as this where a single packet may frequently move between cluster nodes. Second, the overall scalability of underlay networks supportable across a Kubernetes cluster is limited given current CNI methods of network processing. Based on throughput testing results, the only way to mitigate the packet processing concern was to place the entire range environment on a single worker node. Using only a single node removed the overlay network requirement and significantly reduced network processing overhead. With this method, the same throughput test that maxed out at 200 MB/s across cluster nodes, reached almost 1.15 GB/s for a multi-router network (Figure 15) and reducing network size to a single router increased throughput to approximately 1.5 GB/s.

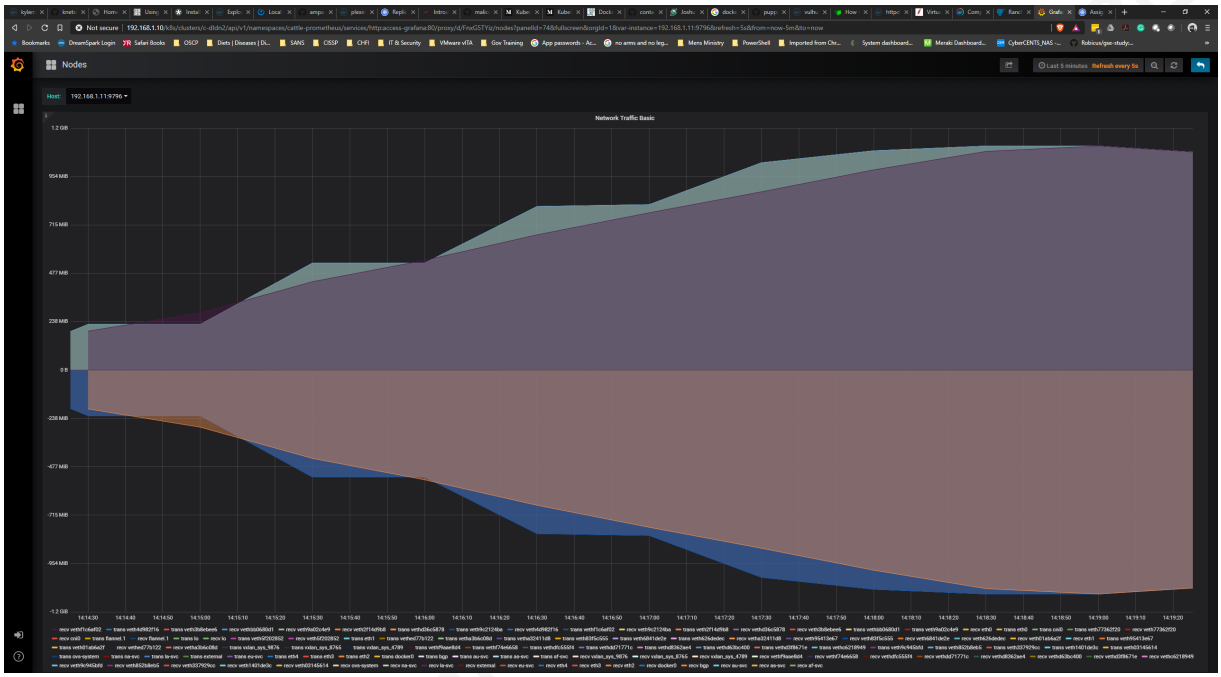

Figure 15: Single Host, Multi-Hop Network Throughput Test – 1.15 GB/s

5.2 Cluster Performance Analysis

Cluster performance is the primary limiting factor in overall network health and capabilities. This research sought to build an environment that operated well, given modest system requirements (See Appendix A). The cluster included three nodes, each with 4 vCPUs, 6 GB of memory, and a 60 GB of disk space. The goal was to create a range that could run on a modern laptop or desktop, or even across a couple of systems if the user did not have a single machine suitable for running all three nodes. As the environment scaled out to its final incarnation, the most significant limiting factor was CPU.

Containers are treated similarly to processes on an operating system, and on each assigned node, CPU contention becomes imminent as the environment scales out. In its tested incarnation, there were 96 Pods distributed across the three-node cluster; 68 Pods for the range network, 19 Pods for Kubernetes cluster management, eight for Prometheus and Grafana logging, and four Pods for Rancher management. In a static state with no outside network traffic, the cluster hovered steadily between 60-80% CPU utilization (Figure 16). It became evident this was a concern because the standard amount of time to start a single Pod increased over time from

a few seconds at the beginning with no additional Pods running, to almost one minute for the same process.

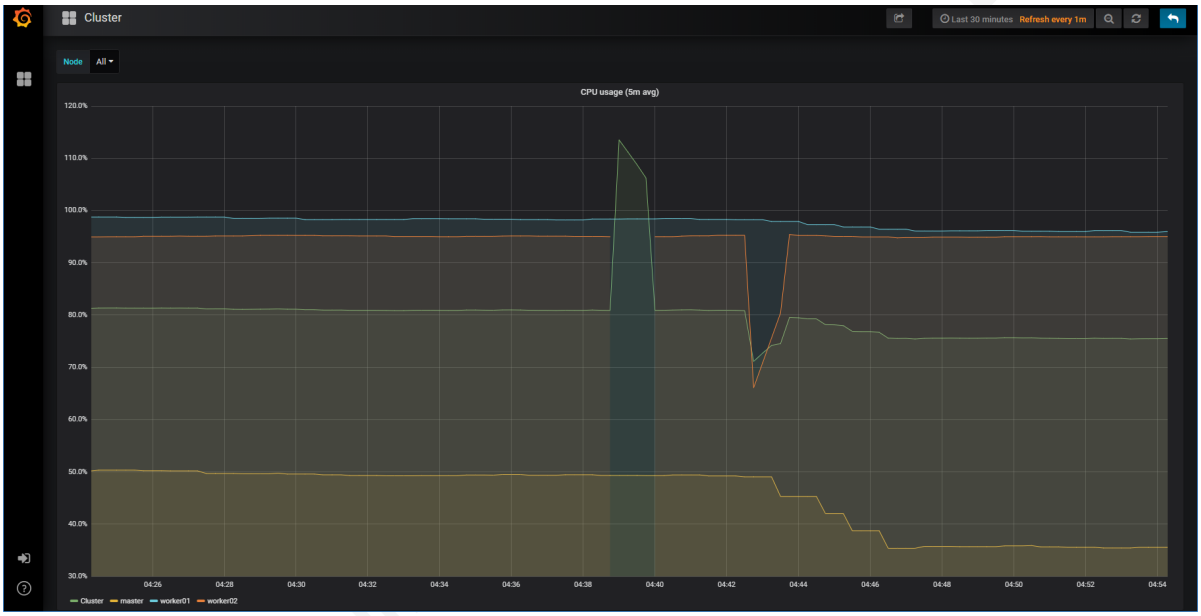

Figure 16: Cluster CPU Utilization – No Network Load

The sustained, increased CPU utilization also impacted the responsiveness of the network. As service fidelity grew, and CPU utilization climbed, the range started to feel sluggish. DNS queries were almost always quick to reply, but initial ICMP requests to hosts would often take anywhere from 1-5 seconds to start responding, and initial connections to websites also required the same 1-5 seconds to begin acknowledging. Further research is necessary to determine the exact cause of the environment's delayed responsiveness, but in its current incarnation, it is more of a slight inconvenience at times rather than a hindrance to platform operations.

6. Conclusion

The modern-day cyber range is often a behemoth of a platform requiring multiple personnel and a significant amount of time, money, and hardware to operate and maintain. Given the ever-shrinking world around us, it is about time that our expectations of cyberoperations and cyber training become more realistic for the individual. The job gap in the security field is not shrinking, and due to the pain of entry, it is not likely to change anytime soon. Research such as

this paper and containerization as a whole offer a means to bring about the change necessary to help bridge that gap and bring sophisticated cyber range platforms to the average or aspiring cybersecurity professionals.

The initial intent of this research was to build a realistic cyber range environment that could run on an affordable system to make it a viable training option for the fiscally challenged. Containerization certainly offers a means to that end, and the scalability, centralized management, and extendibility of Kubernetes make it a suitable candidate to facilitate this capability. Kubernetes and containerization are not without their limitations, however. The entire environment is abstracted in a way that generates significant management overhead, and without the necessary resources, it can become challenging to operate and maintain. Furthermore, building complex network infrastructures on top of a severely abstracted environment such as K8s introduces significant complexity and difficulty scaling.

Further research is required to determine methods of mitigating some of the determined limitations of this platform. Additional hardware should resolve the CPU contention problems, but network throughput will continue to be an issue until a CNI is written to manage underlay networks in a container cluster. The container-based range is useable in its current state and has much to offer concerning learning about how containers work, how networking works, and more information about the various network protocols and analyzing their traffic. It is a step in the right direction of building a platform capable of offering a wide array of capabilities without requiring deep pockets or significant staffing to construct and manage. As containerization and container orchestration continues to evolve, and as training services begin to adopt these platforms, the total cost of ownership of the cyber range will be poised to decrease as well.

Bryan Scarbrough, bryan.scarbrough@gmail.com

References

Brown, T. (2017, April). Bringing Docker To Windows Developers with Windows Server Containers. Retrieved from https://msdn.microsoft.com/en-us/magazine/mt797649.aspx

Cluster Networking. (n.d.). Retrieved June 22, 2019, from https://kubernetes.io/docs/concepts/cluster-administration/networking/

FRRouting/watchfrr. (2019, January 24). Retrieved from https://github.com/FRRouting/frr/blob/master/watchfrr

Hildred, T. (2015, August 28). The History of Containers. Retrieved from https://www.redhat.com/en/blog/history-containers

intel/multus-cni. (2019, 14). Retrieved June 22, 2019, from https://github.com/intel/multus-cni

kubevirt/ovs-cni. (2019, March 10). Retrieved June 22, 2019, from https://github.com/kubevirt/ovs-cni

Lang, P., Cooley, S., Wilhite, C., Wenzel, M., Nikolic, A., & Mezgebu, M. (2019, May 18). Windows Container Version Compatibility. Retrieved from https://docs.microsoft.com/en-us/virtualization/windowscontainers/deploy-containers/version-compatibility

Larsson, M. (2017, December 20). Setting up a Kubernetes cluster using Docker in Docker | Callista Enterprise. Retrieved from https://callistaenterprise.se/blogg/teknik/2017/12/20/kubernetes-on-docker-in-docker/

Security Team. (2019, January 15). 2019 Cybersecurity Trends to Watch: CISOs, the Skills Gap, and State Sponsored Attacks. Retrieved from https://modernciso.com/2019/01/15/2019-cybersecurity-trends-to-watch-cisos-the-skills-gap-and-state-sponsored-attacks/

Shanmugam, K. (2018, November 29). What Are Containers and Why Do We Need Them? Retrieved from https://containerjournal.com/2018/11/30/what-are-containers-and-why-do-we-need-them/

Techopedia. (2019). What is a Cyber Range? - Definition from Techopedia. Retrieved from https://www.techopedia.com/definition/28613/cyber-range

Understanding VXLANs. (n.d.). Retrieved July 3, 2019, from https://www.juniper.net/documentation/en_US/junos/topics/topic-map/sdn-vxlan.html

Vadgama, D. (2016, September 18). Microservices, Docker and buzz words. Retrieved from https://deepakvadgama.com/blog/microservices-and-buzz-words/

Bryan Scarbrough, bryan.scarbrough@gmail.com

Van Camp, E. (2019). ISACA's State of Cybersecurity 2019 Survey: Retaining Qualified Cybersecurity Professionals Increasingly Challenging for Organizations. Retrieved from http://www.isaca.org/About-ISACA/Press-room/News-Releases/2019/Pages/ISACA-State-of-Cybersecurity-2019-Survey.aspx

Wang, W. (2018, October 15). Demystifying Containers 101: A Deep Dive Into Container Technology for Beginners. Retrieved from https://www.freecodecamp.org/news/demystifying-containers-101-a-deep-dive-into-container-technology-for-beginners-d7b60d8511c1/

What is Kubernetes. (2019, April 26). Retrieved from https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix A
Container-Based Cyber Range Build Procedures

This range is maintained and regularly updated on Github at https://github.com/1computerguy/cbcr. The setup and installation procedures for the Github instance are slightly different since a `git clone` will automatically download all of the files and folders in the proper structure. The procedures outlined in this Appendix use the scripts and processes developed at the time of this research.

1) Prep the Infrastructure environment:
   a) Requires 4 virtual switch port groups (can segregate these with VLANs, but not a requirement) – *NOTE*: If using a product like VMware Workstation, LAN Segments are useful for the cluster storage and cluster overlay interfaces if the cluster is running on a single computer. Use bridged networking for external connections into the range. If using something like VirtualBox, then connect each NIC to a separate network, and ensure the management network has Internet connectivity.
      i) 1 – Management connection with Internet connectivity
      ii) 1 – Cluster storage network for NFS share for container data distribution
      iii) 1 – Cluster overlay network for OVS VXLAN network across Pods
      iv) 1 – External port group for connections from outside the cluster
2) Create 3 Ubuntu 18.04 Server VMs (NOTE: An Internet connection is required for the initial build of the range. Once the range is set up, this can, and probably should operate disconnected from Internet.):
   a) Hostnames: master, worker01, worker02
      i) 4 CPU (the more, the better)
      ii) 6GB RAM (the more, the better)
      iii) 60GB Disk
      iv) 4 vNIC
         (1) mgmt (has Internet)
         (2) storage
         (3) internal overlay
         (4) external connect

Bryan Scarbrough, bryan.scarbrough@gmail.com

3) Install Ubuntu 18.04 Server, using the following
   a) Configure static IP for the first NIC on all 3 nodes
      i) Common Ubuntu NIC names for different hypervisors
         (1) eth0 for Xenserver
         (2) ens160 for VMware esxi
         (3) ens33 for VMware Workstation
         (4) enp0s3 for VirtualBox
      ii) Remember the address and gateway used as they will be used in a configuration file later
      iii) Set hostnames according to step 2a
      iv) Create the same username and password across all three nodes
      v) Select defaults for all other configuration settings
      vi) Install OpenSSH Server
      vii) Do not install any metapackages (the options at the end)
4) Install VM Tools
   a) After reboot, install VM tools. Follow instructions for your designated hypervisor. The instructions below are for XenServer.
      i) In the XenCenter management console select the desired VM's *console* tab
      ii) Select VM > Install XenServer Tools...
      iii) Click inside the VM console
      iv) Run the "`lsblk -f`" command and look for the device with Xenserver Tools iso mounted (mine was sr0)
      v) Mount sr0
         (1) `sudo mount /dev/sr0 /mnt`
      vi) Navigate to the /mnt/Linux directory
         (1) `cd /mnt/Linux`
      vii) Run the installer
         (1) `sudo ./install.sh`
      viii) Unmount disk
         (1) `cd ~/ && sudo umount /mnt`
      ix) Power off after installing tools and take a snapshot
         (1) `sudo poweroff`

x) Take a snapshot of all 3 VMs while they are powered off

(1) VERY helpful in the event something goes wrong with the deployment scripts

(2) Often easier to troubleshoot by starting over

(3) Kubernetes can be finicky at times and redeployment can sometimes work better than trying to troubleshoot endlessly

5) Build Cluster

a) From Master VM, after reboot:

i) Create the folder structure from the user's home directory. Run the command below to create the folders, or manually create them like Figure 17.

(1) `mkdir -p ~/cbcr/{build/{cluster,management/pki,range/scraper},resources/{dns/{auth/reverse-zones,recursive},k8s_deployments,kibana_dashboard,monitor/{bro/{config,data},elasticsearch/data,kibana,logstash/{config,patterns,pipeline}},router-configs/{rtr-01,rtr-02,rtr-03,rtr-04,rtr-05,rtr-06}},testing/{iperf,k8s_configs}}`

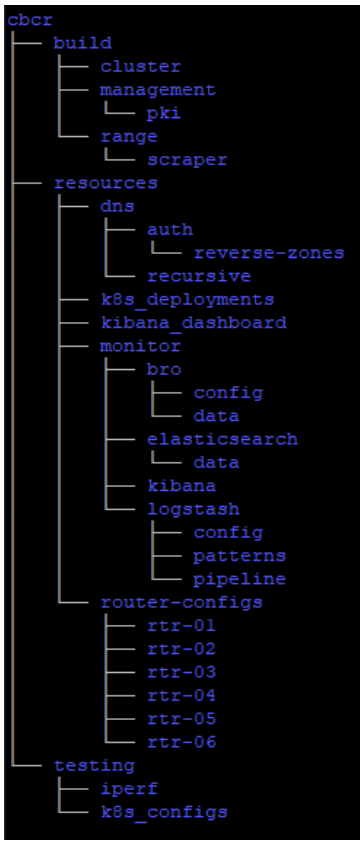

Figure 17: Range Deployment Folder Structure

ii) Next, copy and paste files from the Appendicies below into their appropriate folder.

(1) `~/cbcr/build/cluster`

(a) `build_cluster.sh` (See Appendix B)

(b) `cluster_cfg.csv` (See Appendix C)

(2) `~/cbcr/build/management/pki`

(a) `build_pki.sh` (See Appendix D)

(b) `intermed-config.conf.tmpl` (See Appendix E)

(c) `root-config.conf.tmpl` (See Appendix F)

(3) `~/cbcr/build/management`

(a) `.env` (See Appendix G)

(b) `build_management.sh` (See Appendix H)

(c) `pre-build_management.sh` (See Appendix I)

(d) `docker-compose.yml` (See Appendix J)

(4) `~/cbcr/build/range/scraper`

(a) `scrape.js` (See Appendix K)

(5) `~/cbcr/build/range`

(a) `01-bridge-nets.yml` (See Appendix L)

(b) `build_dns.py` (See Appendix M)

(c) `build_mirror.sh` (See Appendix N)

(d) `build_ovs_links.sh` (See Appendix O)

(e) `build_range_all.sh` (See Appendix P)

(f) `build_web_pki.py` (See Appendix Q)

(g) `build_web.py` (See Appendix R)

(h) `sign_cert.sh` (See Appendix S)

(i) `range_services.csv` (See Appendix T)

(6) `~/cbcr/resources/dns`

(a) `bind.keys` (See Appendix U)

(b) `db.0` (See Appendix V)

(c) `db.127` (See Appendix W)

(d) `db.255` (See Appendix X)

(e) `db.empty` (See Appendix Y)

(f) `db.local` (See Appendix Z)

Bryan Scarbrough, bryan.scarbrough@gmail.com

(g) `db.root` (See Appendix AA)

(h) `rndc.key` (See Appendix BB)

(i) `zones.rfc1918` (See Appendix CC)

(7) `~/cbcr/resources/dns/recursive`

(a) `named.conf` (See Appendix DD)

(b) `named.conf.default-zones` (See Appendix EE)

(c) `named.conf.local` (See Appendix FF)

(d) `named.conf.options` (See Appendix GG)

(8) `~/cbcr/resources/k8s_deployments`

(a) `attack.yml` (See Appendix HH)

(b) `bro-elk.yml` (See Appendix II)

(c) `dns.yml` (See Appendix JJ)

(d) `ftp.yml` (See Appendix KK)

(e) `webmail.yml` (See Appendix LL)

(f) `media.yml` (See Appendix MM)

(g) `ntp.yml` (See Appendix NN)

(h) `router.yml` (See Appendix OO)

(i) `smtp.yml` (See Appendix PP)

(j) `vpn.yml` (See Appendix QQ)

(k) `vuln.yml` (See Appendix RR)

(l) `web.yml` (See Appendix SS)

(9) `~/cbcr/resources/kibana_dashboard`

(a) `cbcr-kibana-export.json` (See Appendix TT)

(10) `~/cbcr/resources/monitor/bro/config`

(a) `local.bro` (See Appendix UU)

(11) `~/cbcr/resources/monitor/elasticsearch`

(a) `elasticsearch.yml` (See Appendix VV)

(12) `~/cbcr/resources/monitor/kibana`

(a) `kibana.yml` (See Appendix WW)

(13) `~/cbcr/resources/monitor/logstash/config`

(a) `logstash.yml` (See Appendix XX)

(14) `~/cbcr/resources/monitor/logstash/pipeline`

(a) `logstash.conf` (See Appendix YY)

(15) `~/cbcr/router-configs`

(a) `daemons` (See Appendix ZZ)

(b) `daemons.conf` (See Appendix AAA)

(16) `~/cbcr/router-configs/rtr-01`

(a) `bgpd.conf` (See Appendix BBB)

(b) `staticd.conf` (See Appendix CCC)

(c) `vtysh.conf` (See Appendix DDD)

(d) `zebra.conf` (See Appendix EEE)

(17) `~/cbcr/router-configs/rtr-02`

(a) `bgpd.conf` (See Appendix FFF)

(b) `staticd.conf` (See Appendix GGG)

(c) `vtysh.conf` (See Appendix HHH)

(d) `zebra.conf` (See Appendix III)

(18) `~/cbcr/router-configs/rtr-03`

(a) `bgpd.conf` (See Appendix JJJ)

(b) `staticd.conf` (See Appendix KKK)

(c) `vtysh.conf` (See Appendix LLL)

(d) `zebra.conf` (See Appendix MMM)

(19) `~/cbcr/router-configs/rtr-04`

(a) `bgpd.conf` (See Appendix NNN)

(b) `staticd.conf` (See Appendix OOO)

(c) `vtysh.conf` (See Appendix PPP)

(d) `zebra.conf` (See Appendix QQQ)

(20) `~/cbcr/router-configs/rtr-05`

(a) `bgpd.conf` (See Appendix RRR)

(b) `staticd.conf` (See Appendix SSS)

(c) `vtysh.conf` (See Appendix TTT)

(d) `zebra.conf` (See Appendix UUU)

(21) `~/cbcr/router-configs/rtr-06`

(a) `bgpd.conf` (See Appendix VVV)

(b) `staticd.conf` (See Appendix WWW)

(c) `vtysh.conf` (See Appendix XXX)

(d) `zebra.conf` (See Appendix YYY)

iii) The file and folder structure should mirror Figure 18.

Figure 18: Range File and Folder Structure After Copy/Paste Operations

iv) Copy and paste the command below to mark all the shell scripts (files ending in .sh) as executable.

(1) `for file in \`find ~/cbcr -name *.sh\`; do chmod +x $file; done`

v) Once all files match Figure 19, and shell scripts are executable, navigate to `~/cbcr/build/cluster` and modify the `cluster_cfg.csv` file to match your network IP/hostname configuration.

(1) Csv file structure: `hostname,mgmt_ip,mgmt_gateway,storage_ip,overlay_ip`

(a) Hostname – for compatibility, do not change hostnames from those configured in step 3a(iii)

(b) Management IP – static address configured in OS install step 3a

(c) Management Gateway – address used in OS install step 3a(ii)

(i) For Internet access during setup

(ii) Recommended to remove this when using the range, or better yet, disconnect it from the Internet completely after setup

(d) Storage IP

(i) Static address with no gateway for NFS connections

(e) Internal overlay network IP

(i) Static address with no gateway for the container-based network interconnections

NOTE: Packages are updated and installed during cluster initialization. If a prompt appears asking "`Restart services during package upgrades without asking?`", select "*`Yes`*".

(2) Run the `build_cluster.sh` file on the MASTER node (if your password has special characters, make sure to enclose it in quotes like the below command).

(a) `sudo ./build_cluster.sh -u <current username> -p "<worker ssh password>" -f <cluster config file>`

(b) If you see the error like the below, wait 3-5 minutes and try again. It seems that Ubuntu checks for updates on boot and locks the update file. Or, you can manually stop the process by using `ps aux | grep apt`, then use a `sudo pkill -9 <proc id number>`

- `E: Could not get lock /var/lib/dpkg/lock-frontend – open (11: Resource temporarily unavailable)`
- `E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), is another process using it?`

(c) Once the script gets to the "...Waiting for all Worker nodes to report a ready status..." prompt:

(3) Open a terminal to worker01

(a) Run the `worker01.sh` script in the home folder

(i) `sudo ./worker01.sh`

(b) Log out of the node and log back in – (this is necessary to populate environment variables for use in other deployment scripts)

(4) Open a terminal to worker02

(a) Run the `worker02.sh` script in the home folder

(i) `sudo ./worker02.sh`

(b) Log out of the node and log back in – (this is necessary to populate environment variables for use in other deployment scripts)

vi) After running worker scripts on each node, go back to the master and, check the status of the "...Waiting..." Prompt – should be finishing up soon.

(1) If the Warning prompt that "`It has been 15 min. since the workerXX setup started…`" pops up, check out the deployment's status manually

(2) Below are a couple of commands to run from the master node to help troubleshoot:

(a) `kubectl get nodes`

(i) Use this to get the high-level view of node status

(ii) `kubectl get pods –all-namespaces`

1. Looking at pod status to see if all are in a "Running" state
2. You can also run:

a. `kubectl describe pod <pod name> -n kube-system`

(iii) `kubectl describe nodes`

1. You're looking for any errors in node status or deployment

(iv) To help troubleshoot from the worker nodes:

1. `sudo systemctl status docker`
   a. Verify the docker service is running
2. `sudo systemctl status kubelet`
   a. Verify the Kubernetes service is running properly
3. `journalctl -xe`
   a. Run this if the kubelet service is not running

b. Look for kubelet start processes and error messages to Google for assistance

4. If the kubelet service is not running and there is an error concerning the cgroup driver
   a. Modify kubeadm cgroup driver in `/var/lib/kubelet/kubeadm-flags.env`
   b. `sudo nano /var/lib/kubelet/kubeadm-flags.env`
   c. Change `cgroup` driver to `39gmt.39`, save and exit
   d. restart kubelet service
   e. `sudo systemctl restart kubelet`

(3) If none of that works, sometimes, it is easiest to revert to snapshot (assuming you took one in the pre-setup stage)

vii) Now, with a working cluster, navigate to the `~/cbcr/build/management` directory:

(1) `cd ~/cbcr/build/management`

(2) FIRST, run `pre_build_management.sh`

(a) `./pre_build_management.sh`

(3) SECOND, Log out of the system, and log back in

(a) this process sets environment variables used throughout the remainder of set up

(b) If you skip this step, things will not work correctly, and you will spend a LOT of time troubleshooting...

(4) THIRD, after you log back in, navigate back to the `$REPO_HOME/build/management` directory and run the following commands

(a) `cd $REPO_HOME/build/management`

(b) `./build_management.sh -u <username>`

(i) This name is the username setup for each node during install

(c) Log into each worker and update mount by running the below command:

(i) `sudo mount -a`

(5) Once management completes, go to the `$REPO_HOME/build/range` directory and run the following:

(a) `cd ../range`

(b) `./build_range_all.sh -u <username> -p <password>`

(i) Enter the username and password of your nodes

(6) Once complete, log into each worker and run the following commands to verify completion

(a) `sudo ovs-vsctl show`

(b) You are looking for a list of OpenVswitch bridges, and, inspect the bridge named *bgp* for the *bro0* internal interface (this is the Bro IDS mirror interface)

6) Once the environment is built, and the underlying structure is in place, you are ready to run your range environment. It is currently static, but I am working toward a dynamic configuration model.

a) To startup all the range services run the following:

i) `kubectl create -f $K8S_CONFIGS`

b) This process will start the range with the services outlined in the range_services.csv file and the router network described in the `range_network.csv` file

c) NOTE: Initial creation of this environment will take approximately 10-15 minutes depending on hardware, and the VPN servers can take up to 20+ minutes depending on how long it takes the init-containers to generate secure keys.

d) To access the environment

i) Build a Virtual machine and connect it to the port group created for the "external" NIC on the Master node

ii) Configure it with an IP in the 167.2.127.xxx/24, a gateway of 167.2.127.1, and DNS of 8.8.8.8 or 9.9.9.9

e) To access the Kibana dashboard for Bro log information navigate to the management IP address port 30001.

i) http://<mgmt ip>:30001

ii) Select *Management* in the left-hand navigation pane

iii) Select *Index Patterns*

(1) If it tells you "No data found" or "No data captured" or something along those lines, wait a little bit, or generate some data from your externally connected VM and refresh the page

(2) In the *Index Pattern* section type "*logstash-**" and select *Next Step*

(3)

f) If you wish to use the pre-configured dashboard, follow the steps below:

i) Copy the contents of the `cbcr-kibana-export.json` (See Appendix TT) and save it to your local disk.

ii) Using a web browser on a system with access to the cluster management IP, navigate to `http://<mgmt ip>:30001`

iii) Select *Management* in the left-hand navigation pane

iv) Select *Saved Objects*

v) In the top right of the window, select *Import*

vi) In the window that opens, select the `cbcr-kibana-export.json` file saved previously

vii) Select *Yes, overwrite all*

viii) After import:

(1) In the left-hand pane, select *Dashboard*

(2) Then select the item called *Dashboard* in the center of the window

(3) If performed correctly, you should see something like the below window (Figure 19)

(4) NOTE: If you just set up your range, it may not look like the one below. This environment has been running for some time.

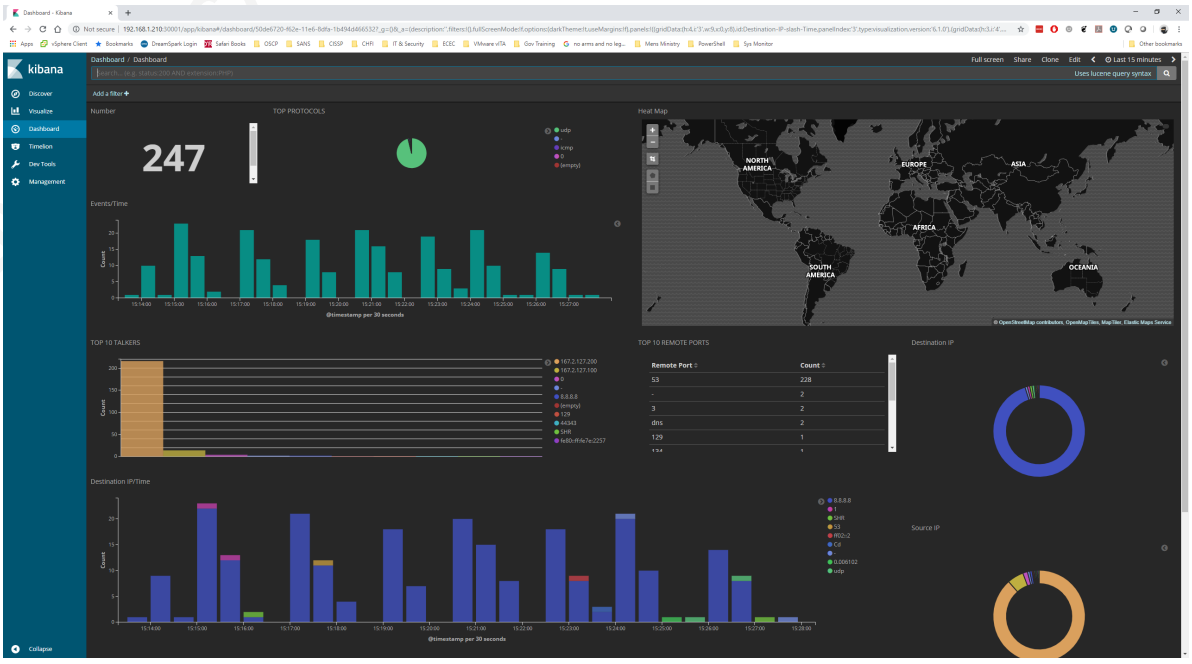

Figure 19: Pre-built Kibana Dashboard

Appendix B

build_cluster.sh

```
#!/bin/bash
set -e

usage() {
    echo ""
    echo "Please make sure all the systems you are configuring have a common username and
password."
    echo "Add the username, password, and the csv configuration file to build the cluster."
    echo ""
    echo " ----"
    echo -e "    Example: $0 -u <username> -p <password> -f <configfile.csv>"
    echo " ----"
    echo ""
    echo "CSV file contains the following columns:"
    echo -e "   - hostname"
    echo -e "   - management IP"
    echo -e "   - default gateway IP"
    echo -e "   - storage IP"
    echo -e "   - OVS Overlay IP"
    echo ""
    echo "----"
    echo "Example of configfile.csv file:"
    echo -e "    master,192.168.1.10,192.168.1.1,10.10.1.10,10.10.2.10"
    echo -e "    worker01,192.168.1.11,192.168.1.1,10.10.1.11,10.10.2.11"
    echo -e "    worker02,192.168.1.12,192.168.1.1,10.10.1.12,10.10.2.12"
    echo "----"
    echo ""
}

SSHPASS=""
user=""
password=""

# Set arguments as IP variables
if [ $# -gt 1 ]
then
    count=0
    while [ "$1" != "" ]
    do
        case $1 in
            -p | --password )
                shift
                SSHPASS="$1"
                export SSHPASS=$SSHPASS
                ;;
            -f | --filename )
                shift
                filename="$1"
                ;;
            -u | --username )
                shift
                user="$1"
                ;;
            -h | --help )
                usage
                exit
                ;;
            * )
                usage
                exit 1
                ;;
        esac
        shift
    done
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
else
    usage
    exit 1
fi

if [ "$filename" == "" ] || [ "$SSHPASS" == "" ] || [ "$user" == "" ]
then
  usage
  exit 1
fi



echo "------------------------------------------------------------"
echo "|  Importing CSV data into local variables                 |"
echo "------------------------------------------------------------"
echo ""
# Parse input CSV and create master, worker01, and worker02 arrays
OLDIFS="$IFS"
IFS=','
while read hostname mgmtip gateway storageip overlayip
do
  if [ "$hostname" ]
  then
    eval "$hostname=($mgmtip $gateway $storageip $overlayip)"
  fi
done < "$filename"
IFS="$OLDIFS"

echo "------------------------------------------------------------"
echo "| Pulling the latest K8s repo data for installation    |"
echo "------------------------------------------------------------"
echo ""
# Add K8s key and repo
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

echo "------------------------------------------------------------"
echo "| Installing packages...                                   |"
echo "------------------------------------------------------------"
echo ""
# Install necessary packages
DEBIAN_FRONTEND=noninteractive apt update && apt upgrade -y
DEBIAN_FRONTEND=noninteractive apt install -y docker.io python-docker pv python-pip kubeadm
kubelet kubectl \
                    geoipupdate docker-compose openvswitch-switch nfs-common \
                    python3-pip sshpass expect

# Disable rpcbind (service added from nfs-common install) so we can share nfs from our nfs
container
systemctl stop rpcbind
systemctl disable rpcbind

echo "------------------------------------------------------------"
echo "| Installing some Python pip packages for use later    |"
echo "------------------------------------------------------------"
echo ""
sudo -u $user pip install dnspython
sudo -u $user pip install geoip2
sudo -u $user pip3 install kuku

echo "------------------------------------------------------------"
echo "|  Writing and applying network configuration              |"
echo "------------------------------------------------------------"
echo ""
# Create static netplan configuration file for static IP addresses
cat > /etc/netplan/50-cloud-init.yaml <<NET
# This file is generated from information provided by
# the datasource.  Changes to it will not persist across an instance.
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
```

```
# network: {config: disabled}
network:
  ethernets:
    eth0:
      addresses: [ ${master[0]}/24 ]
      gateway4: ${master[1]}
      nameservers:
        addresses:
        - 8.8.8.8
      dhcp4: false
    eth1:
      addresses: [ ${master[2]}/24 ]
      dhcp4: false
    eth2:
      addresses: [ ${master[3]}/24 ]
      dhcp4: false
    eth3:
      dhcp4: false
NET

netplan apply

echo "------------------------------------------------------------"
echo "| Enabling IP forwarding for the router containers         |"
echo "------------------------------------------------------------"
echo ""
# Enable IP forwarding for some later functionality
echo "net.ipv4.ip_forward=1" | tee -a /etc/sysctl.conf
echo "net.ipv6.conf.all.forwarding=1" | tee -a /etc/sysctl.conf
sysctl -p

echo "------------------------------------------------------------"
echo "| Disable swap and enable NFS                              |"
echo "------------------------------------------------------------"
echo ""
swapoff -a
sed -i '$s|/swap|\#/swap|' /etc/fstab

modprobe nfs
modprobe nfsd

tee -a /etc/modules <<MOD
nfs
nfsd
MOD

echo "------------------------------------------------------------"
echo "| Writing /etc/hosts file for named access to resoruces    |"
echo "------------------------------------------------------------"
echo ""
cat >> /etc/hosts <<EOF
${master[0]}      master
${master[2]}      storage
${master[3]}      network
${worker01[0]}      worker01
${worker02[0]}      worker02
${worker01[3]}      network01
${worker02[3]}      network02
EOF

echo "------------------------------------------------------------"
echo "| Configuring systemd cgroupdriver for Docker              |"
echo "------------------------------------------------------------"
echo ""
# Change Docker manager to systemd, not cgroup
tee /etc/docker/daemon.json <<DOC
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
```

```
  },
  "storage-driver": "overlay2"
}
DOC

mkdir -p /etc/systemd/system/docker.service.d
systemctl daemon-reload
systemctl enable docker
systemctl restart docker

usermod -aG docker $user

echo "------------------------------------------------------"
echo "| Initializing Kubernetes Cluster!                   |"
echo "|  I can take a couple of minutes, please be patient...|"
echo "------------------------------------------------------"
echo ""

home_dir="/home/$user/"
kubeadm init --pod-network-cidr=10.244.0.0/16 > cluster-init
sudo -u $user mkdir -p $home_dir/.kube
cp -i /etc/kubernetes/admin.conf $home_dir/.kube/config
chown $user:$user $home_dir/.kube/config

sudo -u $user kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/a70459be0084506e4ec919aa1c114638878db11b/Documen
tation/kube-flannel.yml

while [ "$(kubectl get nodes | awk '{if (NR!=1) {print $2}}')" != "Ready" ]
do
    echo "...Waiting for Primary CNI to load and cluster node to be Ready..."
    sleep 2
done

echo "Installing Multus-CNI..."
sudo -u $user kubectl apply -f https://raw.githubusercontent.com/intel/multus-
cni/master/images/multus-daemonset.yml

while [ "$(kubectl get pods --all-namespaces | grep multus | awk '{print $4}')" != "Running" ]
do
    echo "...Waiting for Multus CNI to load properly..."
    sleep 2
done

echo "Now installing OVS CNI"
sudo -u $user kubectl apply -f https://raw.githubusercontent.com/kubevirt/ovs-
cni/master/examples/kubernetes-ovs-cni.yml

while [ "$(kubectl get pods --all-namespaces | grep ovs | awk '{print $4}')" != "Running" ]
do
    echo "...Waiting for OVS CNI to load properly..."
    sleep 2
done

echo "*--------------------------------"
echo "*"
echo -e "*  Writing Worker01 configuration to worker01.sh..."
echo "*"
echo "*--------------------------------"

hosts=`cat /etc/hosts`

sudo -u $user cat > worker01.sh <<WK1
#!/bin/bash
set -e

echo "------------------------------------------------------"
echo "| Installing packages...                             |"
echo "------------------------------------------------------"
echo ""
# Add kubernetes repo
```

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add
apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

# Install Updates
DEBIAN_FRONTEND=noninteractive apt update && apt upgrade -y

# Install packages
DEBIAN_FRONTEND=noninteractive apt install -y sshpass docker.io kubeadm kubelet kubectl
openvswitch-switch curl nfs-common

echo "------------------------------------------------------------"
echo "|  Writing and applying network configuration              |"
echo "------------------------------------------------------------"
echo ""
cat > /etc/netplan/50-cloud-init.yaml <<NET
# This file is generated from information provided by
# the datasource.  Changes to it will not persist across an instance.
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  ethernets:
    eth0:
      addresses: [ ${worker01[0]}/24 ]
      gateway4: ${worker01[1]}
      nameservers:
        addresses:
        - 8.8.8.8
      dhcp4: false
    eth1:
      addresses: [ ${worker01[2]}/24 ]
      dhcp4: false
    eth2:
      addresses: [ ${worker01[3]}/24 ]
      dhcp4: false
    eth3:
      dhcp4: false
NET

netplan apply

cat > /etc/hosts <<EOF
$hosts
EOF

cat > ~/.env <<EOF
WORK01_NET=`host network01 | grep "has address" | awk '{print $4}'`
WORK02_NET=`host network02 | grep "has address" | awk '{print $4}'`
MSTR_NET=`host network | grep "has address" | awk '{print $4}'`
EOF

cat >> ~/.bashrc <<ENV
set -a
    [ -f ~/.env ] && . ~/.env
set +a
ENV

# Disable swap (incompatible with K8s 1.7+) and add NFS share mount for internal Docker registry
swapoff -a
sed -i '\$s|/swap|\#/swap|' /etc/fstab
echo "storage:/certs       /etc/docker/certs.d      nfs
auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0" >> /etc/fstab

# Enable IP forwarding for some later functionality
echo "net.ipv4.ip_forward=1" | tee -a /etc/sysctl.conf
echo "net.ipv6.conf.all.forwarding=1" | tee -a /etc/sysctl.conf

# Reload
sysctl -p /etc/sysctl.conf

# Change Docker manager to systemd, not cgroup
```

```
echo "------------------------------------------------------"
echo "| Configuring systemd cgroupdriver for Docker          |"
echo "------------------------------------------------------"
echo ""
# Change Docker manager to systemd, not cgroup
tee /etc/docker/daemon.json <<DOC
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
DOC

mkdir -p /etc/systemd/system/docker.service.d
systemctl daemon-reload
systemctl enable docker
systemctl restart docker

mkdir -p /etc/docker/certs.d

# Sleep to ensure docker process is started fully and that cgroup driver is loaded properly
echo -e "Please wait while we do some things..."
sleep 15

usermod -aG docker greyadmin

echo "------------------------------------------------------"
echo "| Initializing Kubernetes Cluster!                   |"
echo "|  I can take a couple of minutes, please be patient...|"
echo "------------------------------------------------------"
echo ""
`grep -A 2 "kubeadm join" cluster-init`

WK1


echo "*---------------------------------"
echo "*"
echo -e "*  Writing Worker02 configuration to worker02.sh..."
echo "*"
echo "*---------------------------------"

sudo -u $user cat > worker02.sh <<WK2
#!/bin/bash
set -e

echo "------------------------------------------------------"
echo "| Installing packages...                             |"
echo "------------------------------------------------------"
echo ""
# Add kubernetes repo
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add
apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

# Install Updates
DEBIAN_FRONTEND=noninteractive apt update && apt upgrade -y

# Install packages
DEBIAN_FRONTEND=noninteractive apt install -y sshpass docker.io kubeadm kubelet kubectl
openvswitch-switch curl nfs-common

echo "------------------------------------------------------------"
echo "|  Writing and applying network configuration              |"
echo "------------------------------------------------------------"
echo ""
cat > /etc/netplan/50-cloud-init.yaml <<NET
# This file is generated from information provided by
# the datasource.  Changes to it will not persist across an instance.
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  ethernets:
    eth0:
      addresses: [ ${worker02[0]}/24 ]
      gateway4: ${worker02[1]}
      nameservers:
        addresses:
        - 8.8.8.8
      dhcp4: false
    eth1:
      addresses: [ ${worker02[2]}/24 ]
      dhcp4: false
    eth2:
      addresses: [ ${worker02[3]}/24 ]
      dhcp4: false
    eth3:
      dhcp4: false
NET

netplan apply

cat > /etc/hosts <<EOF
$hosts
EOF

cat > ~/.env <<EOF
WORK01_NET=`host network01 | grep "has address" | awk '{print $4}'`
WORK02_NET=`host network02 | grep "has address" | awk '{print $4}'`
MSTR_NET=`host network | grep "has address" | awk '{print $4}'`
EOF

cat >> ~/.bashrc <<ENV
set -a
    [ -f ~/.env ] && . ~/.env
set +a
ENV

# Disable swap (incompatible with K8s 1.7+) and add NFS share mount for internal Docker registry
swapoff -a
sed -i '\$s|/swap|\#/swap|' /etc/fstab
echo "storage:/certs       /etc/docker/certs.d      nfs
auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0" >> /etc/fstab

# Enable IP forwarding for some later functionality
echo "net.ipv4.ip_forward=1" | tee -a /etc/sysctl.conf
echo "net.ipv6.conf.all.forwarding=1" | tee -a /etc/sysctl.conf

# Reload
sysctl -p /etc/sysctl.conf

echo "------------------------------------------------------"
echo "| Configuring systemd cgroupdriver for Docker          |"
echo "------------------------------------------------------"
echo ""
# Change Docker manager to systemd, not cgroup
tee /etc/docker/daemon.json <<DOC
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
DOC

mkdir -p /etc/systemd/system/docker.service.d
systemctl daemon-reload
```

```
systemctl enable docker
systemctl restart docker

mkdir -p /etc/docker/certs.d

# Sleep to ensure docker process is started fully and that cgroup driver is loaded properly
echo -e "Please wait while we do some things..."
sleep 15

usermod -aG docker greyadmin

echo "------------------------------------------------------------"
echo "| Initializing Kubernetes Cluster!                         |"
echo "|  I can take a couple of minutes, please be patient...    |"
echo "------------------------------------------------------------"
echo ""
`grep -A 2 "kubeadm join" cluster-init`

WK2


chmod +x worker01.sh
chmod +x worker02.sh

sshpass -p "$SSHPASS" scp -o StrictHostKeyChecking=no worker01.sh $user@worker01:~/worker01.sh
#echo $SSHPASS | sshpass -p "$SSHPASS" ssh -o StrictHostKeyChecking=no $user@worker01 cat \| sudo
--prompt="" -S -- ./worker01.sh

sshpass -p "$SSHPASS" scp -o StrictHostKeyChecking=no worker02.sh $user@worker02:~/worker02.sh
#echo $SSHPASS | sshpass -p "$SSHPASS" ssh -o StrictHostKeyChecking=no $user@worker02 cat \| sudo
--prompt="" -S -- ./worker02.sh

echo "Waiting for Worker01 to report a ready status..."
echo -e "  - This process will wait 15 minutes if worker01 does not report"
echo -e "    successfully, you can look to the install docs for troubleshooting"
echo -e "    steps."

testcount=0
while [ "$(kubectl get nodes | grep worker01 | awk '{print $2}')" != "Ready" ]
do
    sleep 30
    let testcount=$testcount+1
    if [ $testcount -eq 30 ]
    then
                echo "|-------------------------------------------------------------------------------
----"
                echo "|"
                echo "| It has been 15 min. since the worker01 setup started..."
                echo "|"
                echo "| There may be a problem with your cluster. Please log into the worker nodes"
                echo "| manually to check the status of the kubelet and/or docker processes and
the"
                echo "| output of the /var/log/syslog file to troubleshoot this potential issue."
                echo "| NOTE: here may not be any problems other than network delays in pulling
the"
                echo "| docker containers required to run the cluster."
                echo "|"
                echo "| You can feel free to wait a little longer, or begin troubleshooting now,
it's up"
                echo "| to you..."
                echo "|"
                echo "| To check the status of the nodes use the commands below"
                echo "|        - kubectl get nodes"
                echo "|        - kubectl describe nodes <node name>"
      echo "|"
      echo "| If you have any questions, comments, or concerns, about this range, please"
      echo "| feel free to raise an Issue on my github:"
      echo "|"
      echo "|     https://github.com/1computerguy/cbcr"
      echo "|"
```

```
            echo "|-------------------------------------------------------------------------
-----"

            exit 1
     else
      echo "...Still waiting, please be patient"
    fi
done

echo ""
echo "Worker01 reported ready!"
echo ""
echo ""
echo "Waiting for Worker02 to report a ready status..."
echo -e "  - This process will wait 15 minutes if worker02 does not report"
echo -e "    successfully, you can look to the install docs for troubleshooting"
echo -e "    steps."

testcount=0
while [ "$(kubectl get nodes | grep worker02 | awk '{print $2}')" != "Ready" ]
do
    sleep 30
    let testcount=$testcount+1
    if [ $testcount -eq 30 ]
    then
            echo "|-------------------------------------------------------------------------
----"
            echo "|"
            echo "| It has been 15 min. since worker02 setup started..."
            echo "|"
            echo "| There may be a problem with your cluster. Please log into the worker nodes"
            echo "| manually to check the status of the kubelet and/or docker processes and
the"
            echo "| output of the /var/log/syslog file to troubleshoot this potential issue."
            echo "| NOTE: here may not be any problems other than network delays in pulling
the"
            echo "| docker containers required to run the cluster."
            echo "|"
            echo "| You can feel free to wait a little longer, or begin troubleshooting now,
it's up"
            echo "| to you..."
            echo "|"
            echo "| To check the status of the nodes use the commands below"
            echo "|        - kubectl get nodes"
            echo "|        - kubectl describe nodes <node name>"
      echo "|"
      echo "| If you have any questions, comments, or concerns, about this range, please"
      echo "| feel free to raise an Issue on my github:"
      echo "|"
      echo "|     https://github.com/1computerguy/cbcr"
      echo "|"
            echo "|-------------------------------------------------------------------------
-----"

            exit 1
     else
      echo "...Still waiting, please be patient"
    fi
done

echo ""
echo "Worker02 reported ready!"
echo ""
echo ""
echo ""
echo "|-------------------------------------------------------------------------"
echo "|"
echo "| Your Kubernetes cluster is now configured and ready to use."
echo "|"
echo "| You are ready to move to the next stage of your Cyber Range setup."
echo "|"
```

```
echo "| Please proceed to the management directory cd ../management to continue."
echo "| If you have any questions, comments, or concerns, please feel free"
echo "| to raise an Issue on my github:"
echo "|"
echo "|     https://github.com/1computerguy/cbcr"
echo "|"
echo "|--------------------------------------------------------------------------"

exit 0
```

Appendix C

cluster_cfg.csv

```
master,192.168.1.210,192.168.1.1,10.10.0.210,10.10.1.210
worker01,192.168.1.211,192.168.1.1,10.10.0.211,10.10.1.211
worker02,192.168.1.212,192.168.1.1,10.10.0.212,10.10.1.212
```

Appendix D

build_pki.sh

```
#!/bin/bash

root_dir="$PKI_HOME/root-ca"
intermed_dir="$PKI_HOME/intermed-ca"
dom_name="range.com"
alt_dom_name="range.net"
start_date=`date +%y%m%d000000Z -u -d -1day`
end_date=`date +%y%m%d000000Z -u -d +10years+1day`

# Build base folder/file structure and set some permissions
mkdir -p {$root_dir,$intermed_dir}/{certreqs,certs,crl,newcerts,private}
chmod 700 {$root_dir,$intermed_dir}/private
touch $root_dir/{root-ca.index,root-ca.index.attr,root-ca.serial}
echo 00 > $root_dir/root-ca.crlnum
touch $intermed_dir/{intermed-ca.index,intermed-ca.index.attr,intermed-ca.serial}
echo 00 > $intermed_dir/intermed-ca.crlnum


sed "s|{{PKI_HOME}}|$PKI_HOME|g" root-config.cnf.tmpl | sed "s/{{DOM_NAME}}/$dom_name/g" | sed
"s/{{ALT_DOM_NAME}}/$alt_dom_name/g" > $root_dir/root-ca.cnf

sed "s|{{PKI_HOME}}|$PKI_HOME|g" intermed-config.cnf.tmpl | sed "s/{{DOM_NAME}}/$dom_name/g" |
sed "s/{{ALT_DOM_NAME}}/$alt_dom_name/g" > $intermed_dir/intermed-ca.cnf

# Setup Root CA
cd $root_dir
export OPENSSL_CONF=$root_dir/root-ca.cnf
# Generate secret passphrase that can be used to automate signing
openssl rand -base64 48 > .passphrase

# Generate initial Root CA Key
openssl req -new -passout file:.passphrase -out root-ca.req.pem

# Set permissions, and generate remaining Root keys
chmod 400 private/root-ca.key.pem
openssl req -new -passin file:.passphrase -key private/root-ca.key.pem -out root-ca.req.pem
openssl rand -hex 16 > root-ca.serial
expect -c "
spawn openssl ca -selfsign -in root-ca.req.pem -passin file:.passphrase -out root-ca.cert.pem -
extensions root-ca_ext -startdate $start_date -enddate $end_date
expect \"ign the certificate\" {
  send \"y\r\"

  expect \"1 out of 1 certificate requests certified, commit\"
  send \"y\r\"
  exp_continue
  }
"

openssl ca -gencrl -passin file:.passphrase -out crl/root-ca.crl

# Setup Intermediate CA
cd $intermed_dir
export OPENSSL_CONF=$intermed_dir/intermed-ca.cnf

# Generate secret passphrase that can be used to automate signing
openssl rand -base64 48 > .passphrase

# Generate initial Root CA Key
openssl req -new -passout file:.passphrase -out intermed-ca.req.pem

chmod 400 private/intermed-ca.key.pem
openssl rand -hex 16 > intermed-ca.serial
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
openssl req -new -passin file:.passphrase -key private/intermed-ca.key.pem -out intermed-
ca.req.pem
cp intermed-ca.req.pem $root_dir/certreqs

cd $root_dir
export OPENSSL_CONF=$root_dir/root-ca.cnf
openssl rand -hex 16 > root-ca.serial
expect -c "
spawn openssl ca -in certreqs/intermed-ca.req.pem -passin file:.passphrase -out certs/intermed-
ca.cert.pem -extensions intermed-ca_ext -startdate $start_date -enddate $end_date
expect \"ign the certificate\" {
  send \"y\r\"

  expect \"1 out of 1 certificate requests certified, commit\"
  send \"y\r\"
  exp_continue
  }
"
cp certs/intermed-ca.cert.pem $intermed_dir/
cd $intermed_dir
export OPENSSL_CONF=$intermed_dir/intermed-ca.cnf
openssl ca -gencrl -passin file:.passphrase -out crl/intermed-ca.crl

exit 0
```

Appendix E

intermed-config.cnf.tmpl

```
#
# OpenSSL configuration for the Intermediate Certification Authority.
#

#
# This definition doesn't work if HOME isn't defined.
CA_HOME                 = {{PKI_HOME}}/intermed-ca
RANDFILE                = $ENV::CA_HOME/private/.rnd
oid_section             = new_oids

#
# XMPP address Support
[ new_oids ]
xmppAddr          = 1.3.6.1.5.5.7.8.5
dnsSRV            = 1.3.6.1.5.5.7.8.7

#
# Default Certification Authority
[ ca ]
default_ca              = intermed_ca

#
# Intermediate Certification Authority
[ intermed_ca ]
dir                     = $ENV::CA_HOME
certs                   = $dir/certs
serial                  = $dir/intermed-ca.serial
database                = $dir/intermed-ca.index
new_certs_dir           = $dir/newcerts
certificate             = $dir/intermed-ca.cert.pem
private_key             = $dir/private/intermed-ca.key.pem
default_days            = 730 # Two years
crl                     = $dir/crl/intermed-ca.crl
crl_dir                 = $dir/crl
crlnumber               = $dir/intermed-ca.crlnum
name_opt                = multiline, align
cert_opt                = no_pubkey
copy_extensions         = copy
crl_extensions          = crl_ext
default_crl_days        = 30
default_md              = sha256
preserve                = no
email_in_dn             = no
policy                  = policy
unique_subject          = no

#
# Distinguished Name Policy
[ policy ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied

#
# Distinguished Name Policy for Personal Certificates
[ user_policy ]
countryName             = supplied
stateOrProvinceName     = optional
localityName            = supplied
organizationName        = optional
organizationalUnitName  = optional
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
commonName              = supplied
emailAddress            = supplied
#xmppAddr               = optional # Added to SubjAltName by req

#
# Intermediate CA request options
[ req ]
default_bits            = 3072
default_keyfile         = private/intermed-ca.key.pem
encrypt_key             = yes
default_md              = sha256
string_mask             = utf8only
utf8                    = yes
prompt                  = no
req_extensions          = req_ext
distinguished_name      = distinguished_name
subjectAltName          = subject_alt_name

#
# Intermediate CA Request Extensions
[ req_ext ]
subjectKeyIdentifier    = hash
subjectAltName          = @subject_alt_name

#
# Distinguished Name (DN)
[ distinguished_name ]
organizationName        = {{ALT_DOM_NAME}}
commonName              = {{ALT_DOM_NAME}} Intermediate Certification Authority

#
# Server Certificate Extensions
[ server_ext ]
basicConstraints        = CA:FALSE
keyUsage                = critical, digitalSignature, keyEncipherment
extendedKeyUsage        = critical, serverAuth, clientAuth
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
authorityInfoAccess     = @auth_info_access
crlDistributionPoints   = crl_dist

#
# Client Certificate Extensions
[ client_ext ]
basicConstraints        = CA:FALSE
keyUsage                = critical, digitalSignature
extendedKeyUsage        = critical, clientAuth
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
authorityInfoAccess     = @auth_info_access
crlDistributionPoints   = crl_dist

#
# User Certificate Extensions
[ user_ext ]
basicConstraints        = CA:FALSE
keyUsage                = critical, digitalSignature
extendedKeyUsage        = critical, clientAuth, emailProtection
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
authorityInfoAccess     = @auth_info_access
crlDistributionPoints   = crl_dist

#
# CRL Certificate Extensions
[ crl_ext ]
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
```

```
#
# Certificate Authorities Alternative Names
[ subject_alt_name ]
URI                     = http://ca.{{ALT_DOM_NAME}}/
email                   = certmaster@{{ALT_DOM_NAME}}

#
# Certificate download addresses for the intermediate CA
[ auth_info_access ]
caIssuers;URI           =
http://ca.{{ALT_DOM_NAME}}/certs/{{ALT_DOM_NAME}}_Intermediate_Certification_Authority.cert.pem

#
# CRL Download address for the intermediate CA
[ crl_dist ]
fullname                =
URI:http://ca.{{ALT_DOM_NAME}}/crl/{{ALT_DOM_NAME}}_Intermediate_Certification_Authority.crl
```

Appendix F

root-config.cnf.tmpl

```
#
# OpenSSL configuration for the Root Certification Authority.
#

#
# This definition doesn't work if HOME isn't defined.
CA_HOME                 = {{PKI_HOME}}/root-ca
RANDFILE                = $ENV::CA_HOME/private/.rnd

#
# Default Certification Authority
[ ca ]
default_ca              = root_ca

#
# Root Certification Authority
[ root_ca ]
dir                     = $ENV::CA_HOME
certs                   = $dir/certs
serial                  = $dir/root-ca.serial
database                = $dir/root-ca.index
new_certs_dir           = $dir/newcerts
certificate             = $dir/root-ca.cert.pem
private_key             = $dir/private/root-ca.key.pem
default_days            = 1826 # Five years
crl                     = $dir/root-ca.crl
crl_dir                 = $dir/crl
crlnumber               = $dir/root-ca.crlnum
name_opt                = multiline, align
cert_opt                = no_pubkey
copy_extensions         = copy
crl_extensions          = crl_ext
default_crl_days        = 180
default_md              = sha256
preserve                = no
email_in_dn             = no
policy                  = policy
unique_subject          = no

#
# Distinguished Name Policy for CAs
[ policy ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = supplied
organizationalUnitName  = optional
commonName              = supplied

#
# Root CA Request Options
[ req ]
default_bits            = 4096
default_keyfile         = private/root-ca.key.pem
encrypt_key             = yes
default_md              = sha256
string_mask             = utf8only
utf8                    = yes
prompt                  = no
req_extensions          = root-ca_req_ext
distinguished_name      = distinguished_name
subjectAltName          = @subject_alt_name

#
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
# Root CA Request Extensions
[ root-ca_req_ext ]
subjectKeyIdentifier    = hash
subjectAltName          = @subject_alt_name

#
# Distinguished Name (DN)
[ distinguished_name ]
organizationName        = {{ALT_DOM_NAME}}
commonName              = {{ALT_DOM_NAME}} Root Certification Authority

#
# Root CA Certificate Extensions
[ root-ca_ext ]
basicConstraints        = critical, CA:true
keyUsage                = critical, keyCertSign, cRLSign
nameConstraints         = critical, @name_constraints
subjectKeyIdentifier    = hash
subjectAltName          = @subject_alt_name
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
authorityInfoAccess     = @auth_info_access
crlDistributionPoints   = crl_dist

#
# Intermediate CA Certificate Extensions
[ intermed-ca_ext ]
basicConstraints        = critical, CA:true, pathlen:0
keyUsage                = critical, keyCertSign, cRLSign
subjectKeyIdentifier    = hash
subjectAltName          = @subject_alt_name
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy
authorityInfoAccess     = @auth_info_access
crlDistributionPoints   = crl_dist

#
# CRL Certificate Extensions
[ crl_ext ]
authorityKeyIdentifier  = keyid:always
issuerAltName           = issuer:copy

#
# Certificate Authorities Alternative Names
[ subject_alt_name ]
URI                     = http://ca.{{DOM_NAME}}/
email                   = certmaster@{{DOM_NAME}}

#
# Name Constraints
[ name_constraints ]
permitted;DNS.1         = {{DOM_NAME}}
permitted;DNS.2         = {{ALT_DOM_NAME}}
permitted;DNS.3         = master
permitted;DNS.4         = lan
permitted;email.1       = {{DOM_NAME}}
permitted;email.2       = {{ALT_DOM_NAME}}

#
# Certificate download addresses for the root CA
[ auth_info_access ]
caIssuers;URI           =
http://ca.{{DOM_NAME}}/certs/{{ALT_DOM_NAME}}_Root_Certification_Authority.cert.pem

#
# CRL Download address for the root CA
[ crl_dist ]
fullname                =
URI:http://ca.{{ALT_DOM_NAME}}/crl/{{ALT_DOM_NAME}}_Root_Certification_Authority.crl
```

Appendix G

.env

```
RANGE_HOME=/range
REPO_HOME=~/cbcr
MGMT_HOME=/range/mgmt
CONFIG_HOME=/range/configs
PKI_HOME=/range/mgmt/pki
TEMPLATE_DIR=$REPO_HOME/build/range/deployments
K8S_CONFIGS=$REPO_HOME/resources/k8s_deployments
NFS_ROOT=/configs
WORK01_NET=`host network01 | grep "has address" | awk '{print $4}'`
WORK02_NET=`host network02 | grep "has address" | awk '{print $4}'`
MSTR_NET=`host network | grep "has address" | awk '{print $4}'`
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix H

build_management.sh

```
#!/bin/bash

set -e

if [ "$(id -u)" == "0" ]; then
          echo "Please run this as a normal user, not root"
          exit 1
fi

source ~/.env

usage() {
    echo ""
    echo "Use this script to build the backend management containers; NFS, Local Registry,"
    echo " and the Rancher manager (this one is for optional use, but can be quite helpful"
    echo " especially if you setup the Prometheus and Grafana logging for troubleshooting and"
    echo " monitoring cluster health and performance). If you anticipate this being a purely"
    echo " ephemeral range, then you don't have to worry about Rancher and it's logging."
    echo ""
    echo " ----"
    echo -e "    Example: $0 -u <username>"
    echo " ----"
    echo ""
    echo ""
}

# Set arguments as variables
if [ $# -gt 1 ]
then
    count=0
    while [ "$1" != "" ]
    do
        case $1 in
            -u | --username )
                shift
                user="$1"
                ;;
            -h | --help )
                usage
                exit
                ;;
            * )
                usage
                exit 1
                ;;
        esac
        shift
    done
else
    usage
    exit 1
fi


echo "------------------------------------------------"
echo "Building directory structure..."
echo "------------------------------------------------"
echo ""
# Create default range directories
# TODO: Make this user configurable
sudo mkdir -p /range/{mgmt/{rancher,registry,pki/{root-ca,intermed-ca}},configs}
sudo mkdir -p /range/configs/attack/{rtr1-attacker,rtr2-attacker,rtr3-attacker,rtr4-
attacker,rtr5-attacker,rtr6-attacker}
sudo mkdir -p /range/configs/smtp/webmail
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
sudo mkdir -p /range/configs/ftp/{ftp.adobe.com,ftp.cisco.com,ftp.malwr.cn,ftp.music.ru}
sudo mkdir -p /range/configs/{web,ftp,network,webmail}
sudo mkdir -p /range/configs/media/stream.com
sudo mkdir -p /range/configs/vuln/{metasploit-vuln,mutillidae,wordpress}
sudo mkdir -p
/range/configs/vpn/{ausvpn.com,expressvpn.com,ipvanish.com,privatevpn.com,vpn.com,nordvpn.com}

sudo cp -r $REPO_HOME/resources/dns $RANGE_HOME/configs/
sudo chown -R 100:101 $RANGE_HOME/configs/dns/auth
sudo cp -r $REPO_HOME/resources/monitor $RANGE_HOME/configs/

sudo mkdir -p $RANGE_HOME/configs/network
sudo cp -r $REPO_HOME/resources/router-configs/* $RANGE_HOME/configs/network

sudo chown -R $user:$user /range

echo "------------------------------------------------"
echo "Generating cert for Docker Registry"
echo "------------------------------------------------"
echo ""
registry_dir="$MGMT_HOME/registry"

openssl req  -newkey rsa:4096 -nodes -sha256 -keyout $registry_dir/domain.key  -x509 -days 365 -
out $registry_dir/domain.crt -subj "/CN=master"
sudo mkdir -p /etc/docker/certs.d/master:5000
sudo cp $registry_dir/domain.crt /etc/docker/certs.d/master:5000/ca.crt

echo "------------------------------------------------"
echo "Building PKI infrastructure..."
echo "------------------------------------------------"
echo ""
cd pki
./build_pki.sh

cd ../
echo "------------------------------------------------"
echo "Building and deploying Management Containers..."
echo "------------------------------------------------"
echo ""
docker-compose up -d

exit 0
```

Appendix I

pre-build_management.sh

```
#!/bin/bash

echo "-------------------------------------------------"
echo "Setting environment variables and making sure they persist..."
echo "-------------------------------------------------"
echo ""
# Set Environment Variables for use with other scripts and environment validation
export RANGE_HOME=/range
export REPO_HOME=~/cbcr
export MGMT_HOME=/range/mgmt
export CONFIG_HOME=/range/configs
export PKI_HOME=/range/mgmt/pki
export TEMPLATE_DIR=$REPO_HOME/build/range/deployments
export K8S_CONFIGS=$REPO_HOME/k8s_configs

# Copy .env file to home directory for use with .bashrc to load variables on login
cp .env ~/.env

cat >> ~/.bashrc <<ENV
set -a
    [ -f ~/.env ] && . ~/.env
set +a
ENV

echo "-------------------------------------------------------------------"
echo "!       Please log out of the terminal and log back in            !"
echo "!    BEFORE continuing. These settings must be in effect          !"
echo "!             for the setup to work properly.                     !"
echo "!                                                                 !"
echo "!   When you log back in, come back to this directory and run     !"
echo "!                 the build_managment.sh script                   !"
echo "-------------------------------------------------------------------"

exit 0
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix J

docker-compose.yml

```
version: "3"
services:
  registry:
    image: registry:2
    restart: always
    environment:
      - REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt
      - REGISTRY_HTTP_TLS_KEY=/certs/domain.key
    volumes:
      - /range/mgmt/registry:/certs
    network_mode: "host"

  rancher:
    image: rancher/rancher
    restart: unless-stopped
    volumes:
      - /range/mgmt/rancher:/var/lib/rancher
    ports:
      - "80:80"
      - "443:443"

  nfs:
    image: erichough/nfs-server:latest
    restart: always
    privileged: true
    network_mode: "host"
    environment:
      - NFS_EXPORT_0=/configs                 *(rw,no_root_squash,no_subtree_check)
      - NFS_EXPORT_1=/certs                 *(rw,no_root_squash,no_subtree_check)
    volumes:
      - /etc/docker/certs.d:/certs
      - /range/configs:/configs
```

Appendix K

scrape.js

```
const scrape = require('website-scraper');
const PuppeteerPlugin = require('website-scraper-puppeteer');
scrapeUrl = process.env.URL;

// Set default save location for web scrapes
if(process.env.SAVE_LOCATION) {
  saveLocation = process.env.SAVE_LOCATION;
} else {
  saveLocation = '/web/output';
}

// Default protocol to https, but allow user to pass environment variable to change it
if(process.env.PROTOCOL) {
  protocol = process.env.PROTOCOL;
} else {
  protocol = 'https';
}

class PuppeteerAction {
  apply(registerAction) {
    registerAction('getReference', ({resource, parentResource, originalReference}) => {
      if (!resource) {
        return { reference: parentResource.url + originalReference };
      }
      return { reference: utils.getRelativePath(parentResource.filename, resource.filename) };
    });
  }
}

scrape({
    urls: [ protocol + '://' + scrapeUrl],
    directory: saveLocation + '/' + scrapeUrl,
    recursive: true,
    maxRecursiveDepth: 1,
    //filenameGenerator: 'bySiteStructure',
    ignoreErrors: false,
    request: {
      headers: {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.157 Safari/537.36'
      }
    },
    plugins: [ new PuppeteerPlugin() ],
    plugins: [ new PuppeteerAction() ]
});
```

Appendix L

01-bridge-nets.yml

```
---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: rtr1-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr1-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: rtr2-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr2-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: rtr5-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr5-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: rtr6-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr6-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: rtr3-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr3-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
```

```
  name: rtr4-svc
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "rtr4-svc"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: bgp
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "bgp"
    }'

---
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: external
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "ovs",
      "bridge": "external"
    }'
```

Appendix M

build_dns.py

```
#!/usr/bin/python3

import csv
import datetime
import os
import glob
import sys
from shutil import copyfile, copytree

def forward_dns(domain, dict_list, auth_named, timestamp):
    # Generate DNS file

    # Header
    dns = """$TTL  86400
@       IN      SOA     ns1.{arg1}. dns-admin.{arg1}. (
                        {arg2}          ; Serial
                        604800          ; Refresh
                        86400          ; Retry
                        2419200          ; Expire
                        604800 )        ; Negative Cache TTL
""".format(arg1=domain, arg2=timestamp)

    # Begin generation of Bind9 db file content
    for vals in dict_list:
        # If the full domain name is the same as the base domain name then we just
        # need to generate a standard A record.
        if vals["full_domain_name"] == domain:
            dns += "{arg1}.      IN      A       {arg2}\n".format(arg1=domain,
arg2=vals["ip_address"])

        # If service type is not web server, just add A record. This is used for ftp, smtp, and
other
        # services that don't use the www CNAME entry.
        elif vals["svc_type"] != "web":
            dns += "{arg1}.      IN      A       {arg2}\n".format(arg1=vals["full_domain_name"],
arg2=vals["ip_address"])

        # For all other entries, add A records for the full domain as well base domain. Since
this is a
        # relatively simple network, both of these are the same.
        else:
            dns += "{arg1}.      IN      A       {arg2}\n".format(arg1=vals["full_domain_name"],
arg2=vals["ip_address"])
            dns += "{arg1}.      IN      A       {arg2}\n".format(arg1=domain,
arg2=vals["ip_address"])

        # If the service is a web server, add the "www" CNAME entry to redirect to the
        # full domain name. Makes both www.google.com and google.com valid DNS entries
        # NO, this is not automatic...
        if vals["svc_type"] == "web":
            dns += "www.{arg1}.     IN      CNAME       {arg2}.\n".format(arg1=domain,
arg2=domain)

    # Add MX records
    for vals3 in dict_list:
        if vals3["svc_type"] == "smtp":
            dns += "{arg1}.      IN      MX      10  {arg2}.\n".format(arg1=domain,
arg2=vals3["full_domain_name"])

    # Add nameservers
    for server in auth_named:
        dns += "{arg1}.     IN    NS    {arg2}.\n".format(arg1=domain, arg2=server)

    return dns
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
def reverse_dns(dns_ips, auth_named, timestamp):
    # Generate output content for reverse dns records. Used formatted strings to
    # create the output. This is written to a series of files in the ./reverse
    # folder. These files are then added to the named.conf file in the bind9 root
    # directory.
    #
    # Generate DNS file header
    dns = """$TTL  86400
@       IN      SOA         ns1.{arg1}. dns-admin.{arg1}. (
                            {arg2}         ; Serial
                            604800         ; Refresh
                            86400         ; Retry
                            2419200         ; Expire
                            604800 )       ; Negative Cache TTL
""".format(arg1=domain, arg2=timestamp)

    # Add reverse pointer records
    for vals in entries:
        dns += "{arg1}.      IN      PTR       {arg2}\n".format(arg1=vals["reverse_address"],
arg2=vals["full_domain_name"])

    # Add nameservers
    for ns in auth_named:
        dns += "                IN      NS      {arg1}\n".format(arg1=ns)

    return dns


def named_dns(dns_files_dir):
    named = """// BIND Configuration
include "/etc/bind/rndc.key";
controls {
    inet 127.0.0.1 port 953 allow { 127.0.0.1; };
};
logging {
    category default { null; };
};
zone "." {
    type hint;
    file "/etc/bind/db.root";
};
zone "localhost" {
    type master;
    file "/etc/bind/db.local";
    allow-update { none; };
};
zone "0.0.127.in-addr.arpa" {
    type master;
    file "/etc/bind/db.127";
    allow-update { none; };
};
zone "255.in-addr.arpa" {
    type master;
    file "/etc/bind/db.255";
    allow-update { none; };
};
zone "0.in-addr.arpa" {
    type master;
    file "/etc/bind/db.0";
    allow-update { none; };
};
options {
    directory "/etc/bind";
    dump-file "/etc/bind/data/cache_dump.db";
    statistics-file "/etc/bind/data/named_stats.txt";
    recursion no;
    allow-recursion { none; };
    allow-query { any; };
    allow-query-cache { any; };
```

```
    listen-on-v6 { none; };
    listen-on { any; };
};
"""
    # Generate named.conf output for forward-lookup zones
    named_files = dns_files_dir + "/db.*"
    for filename in glob.glob(named_files):
        forward_file = os.path.basename(filename)
        forward_domain = '.'.join(forward_file.split('.')[1:])
        named += """zone "{domain_name}" {{
    type master;
    file "/etc/bind/{domain_file}";
    allow-update {{ none; }};
}};
""".format(domain_name=forward_domain, domain_file=forward_file)

    # Generate named.conf output for reverse-lookup zones
    reverse_files = dns_files_dir + "/reverse-zones/db.*"
    for filename in glob.glob(reverse_files):
        reverse_file = os.path.basename(filename)
        reverse_domain = '.'.join(reverse_file.split('.')[1:])
        named += """zone "{zone_number}.in-addr.arpa" {{
    type master;
    file "/etc/bind/reverse_zones/{domain_file}";
}};
""".format(zone_number=reverse_domain, domain_file=reverse_file)

    return named


# Grab save directory from command line
build_file = sys.argv[1]
copy_directory = os.environ["CONFIG_HOME"] + "/dns"
write_directory = copy_directory + "/auth"
reverse_directory = write_directory + "/reverse-zones"
dns_build_dir = os.environ["REPO_HOME"] + "/resources/dns"

# Make required directories for DNS servers if they don't already exist
#if not os.path.exists(write_directory):
#    os.makedirs(write_directory)

#if not os.path.exists(reverse_directory):
#    os.makedirs(reverse_directory)

# import range_services.csv file - maybe accept this as an argument - consider...
services_reader = csv.reader(open(build_file))

# Generate timestamp for DNS serial - useful if changes are made so Bind will
# automatically update - so long as the serial is updated
time = '{0:%Y%m%d%H}'.format(datetime.datetime.now())

forward_svcs = {}
reverse_ips = {}
nameservers = []

# Iterate over the imported csv file and get the necessary data
#    - Needed for dns from the range_services.csv file:
#       - full domain name
#       - base domain name (this will be the Bind db file created)
#       - IP address
#       - Service type - to know what type of records to generate
#       - Sub service type - necessary for some MX record and other processing
#
#   - Add something for the "flat" DNS configuration that selects the root
#       servers in lieu of the tiered DNS and the auth servers
#       May have to rewrite portions to change nameserver addresses...
for row in services_reader:
    if row[0] != 'svc_category':
        if row[6] != '':
            if row[6] in forward_svcs:
```

```
                forward_svcs[row[6]].append({'full_domain_name':row[1],
                                             'ip_address':row[2],
                                             'svc_type':row[4],
                                             'svc_sub_type':row[5]})
            else:
                forward_svcs[row[6]] = [{'full_domain_name':row[1],
                                         'ip_address':row[2],
                                         'svc_type':row[4],
                                         'svc_sub_type':row[5]}]

            # Create dictionary of dictionaries keyed on the last octet of the IP
            # The last-octet will be used to create the necessary Bind9 reverse
            # lookup files.
            last_octet = row[2].split('.')[3]
            first_three = '.'.join((row[2].split('.')[:3][::-1]))
            if last_octet in reverse_ips:
                reverse_ips[last_octet].append({'full_domain_name':row[1],
                                                'reverse_address':first_three})
            else:
                reverse_ips[last_octet] = [{'full_domain_name':row[1],
                                            'reverse_address':first_three}]

        # Generate list of authoritative nameservers
        if row[5] == 'authoritative':
            nameservers.append(row[1])

# write forward zone db files
for domain, dict_list in forward_svcs.items():
    forward_file = write_directory + "/db." + domain
    with open(forward_file, 'w') as forward_write:
        forward_write.write(forward_dns(domain, dict_list, nameservers, time))

# write reverse-zone db files
for domain, entries in reverse_ips.items():
    reverse_file = reverse_directory + "/db." + domain
    with open(reverse_file, 'w') as reverse_write:
        reverse_write.write(reverse_dns(reverse_ips, nameservers, time))

named_file = write_directory + "/named.conf"
# Write named.conf to file
with open(named_file, 'w') as named_write:
    named_write.write(named_dns(write_directory))

for full_filename in glob.glob(dns_build_dir + "/auth/*"):
    filename = os.path.basename(full_filename)
    copyfile(full_filename, write_directory + "/" + filename)

#dirs = ["root", "recursive"]

#for path in dirs:
#    copytree(dns_build_dir + "/" + path , copy_directory + "/" + path)
```

Appendix N

build_mirror.sh

```
#!/bin/bash

if [ $# -ne 2 ]; then
    echo "Add a mirror to OVS bridge"
    echo "NOTE: a tap interface is added first, then that tap is configured as"
    echo "the mirror interface below. It is then used for network monitoring"
    echo "using the Bro IDS"
    echo "Usage: $0 <bridge> <output port>"
    echo "Example: $0 bgp bro0"
    exit 1
fi

br=$1
output_port=$2

ovs-vsctl add-port $br $output_port \
  -- set interface $output_port type=internal \
  -- --id=@p get port $output_port \
  -- --id=@m create mirror name=m0 \
  select-all=true output-port=@p \
  -- set bridge $br mirrors=@m

ip link set up dev $output_port
```

Appendix O

build_ovs_links.sh

```
#!/bin/bash
#######
# This script must be run as root or with sudo!
#######

source ~/.env

role=""

usage() {
    echo "------------------------------------------------------------"
    echo "* This script creates OVS bridges for network"
    echo "* It requires a role variable be passed to setup"
    echo "* everything properly. See example below"
    echo "*"
    echo "* $0 -r <role> -l br1 br2 br3 brN"
    echo "*"
    echo "* Acceptable roles: master, worker01, worker02"
    echo ""
    echo " NOTE: if role is left blank, system hostname"
    echo "  will be used. If hostname does not match the"
    echo "  above roles, the script will exit and do nothing"
    echo "------------------------------------------------------------"
    exit 0
}

create_bridges() {
    #bridges = (bgp rtr1-svc rtr2-svc rtr3-svc rtr4-svc rtr5-svc rtr6-svc external)
    br="$1"
    count="$2"
    role="$3"

    ovs-vsctl add-br $br
    ifconfig $br up
    ovs-vsctl set int $br mtu_request=1450
    case "$role" in
        master )
            ovs-vsctl add-port $br vx01-${br} -- set interface vx01-${br} type=vxlan
options:remote_ip=$WORK01_NET options:key=111${count} options:dst_port=4789
            ovs-vsctl add-port $br vx02-${br} -- set interface vx02-${br} type=vxlan
options:remote_ip=$WORK02_NET options:key=222${count} options:dst_port=4789
            ;;
        worker01 )
            ovs-vsctl add-port $br vx-${br} -- set interface vx-${br} type=vxlan
options:remote_ip=$MSTR_NET options:key=111${count} options:dst_port=4789
            ;;
        worker02 )
            ovs-vsctl add-port $br vx-${br} -- set interface vx-${br} type=vxlan
options:remote_ip=$MSTR_NET options:key=222${count} options:dst_port=4789
            ;;
    esac
}

if [ $# -gt 0 ]
then
    while [ "$1" != "" ]
    do
        case "$1" in
            -r | --role )
                shift
                role="$1"
                ;;
            -l | --list )
                shift
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
                count=1
                if [ "$role" == "" ]
                then
                    role=`cat /etc/hostname`
                fi
                while [ "$1" != "" ]
                do
                    create_bridges "$1" "$count" "$role"
                    shift
                    let count+=1
                done
                ;;
            -h | --help )
                usage
                ;;
        esac
        shift
    done
else
    usage
fi
```

Appendix P

build_range_all.sh

```
#!/bin/bash

usage() {
    echo ""
    echo "Please make sure all the systems you are configuring have a common username and
password."
    echo "Add the password to the command to automate the deployment of OVS to the worker nodes."
    echo ""
    echo " ----"
    echo -e "    Example: $0 -p <password> -u <username>"
    echo " ----"
    echo ""
    echo ""
}

# Set arguments as IP variables
if [ $# -gt 1 ]
then
    while [ "$1" != "" ]
    do
        case $1 in
            -u | --username )
                shift
                user="$1"
                ;;
            -p | --password )
                shift
                export SSHPASS="$1"
                ;;
            -h | --help )
                usage
                exit
                ;;
            * )
                usage
                exit 1
                ;;
        esac
        shift
    done
else
    usage
    exit 1
fi

# Build website scraper and start scraping sites
#echo "------------------------------------------------------------"
#echo "|  Building the website scraper. This will be used to   |"
#echo "|  pull download web content and populate sites for     |"
#echo "|  deployment to the nginx servers"
#echo "------------------------------------------------------------"
#echo ""
#cd scraper
#./build.sh
#cd ..

echo "------------------------------------------------------------"
echo "|  Scraping / Downloading websites listed in the          |"
echo "|  range_services.csv file. This process will run in the  |"
echo "|  backbround and can take between 30-45 min.             |"
echo "|                                                         |"
echo "|  Use the docker ps -a command to see if there are any   |"
echo "|    remaining web scrapers after about 30 or so minutes  |"
echo "|    I have found that any still running after this amount|"
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
echo "|    of time can be safely stopped without causing major  |"
echo "|    problems."
echo "-------------------------------------------------------------"
echo ""
downloader="site-downloader"
docker pull bscarbrough/$downloader:latest
docker tag bscarbrough/$downloader:latest master:5000/$downloader:latest
docker push master:5000/$downloader:latest
docker rmi bscarbrough/$downloader:latest

python3 build_web.py

echo "-------------------------------------------------------------"
echo "|  Building all containers required for environment     |"
echo "|                                                       |"
echo "-------------------------------------------------------------"
echo ""
#########################################################################
# UNCOMMENT lines below if you wish to build all containers from scratch
# Otherwise, you can just pull them from my regularly updated Docker Hub
# Repository
#########################################################################
#cd $REPO_HOME/range_svcs
#./build_all.sh

images=(elasticsearch bro-ids kibana mysql postgres kali metasploit-vuln-svc-emu metasploit vuln-
wordpress tor-node openvpn ftpd media ntpd smtp nginx logstash frr webmail vuln-mutillidae bind)

for image in ${images[@]}
do
    docker pull bscarbrough/$image:latest
    docker tag bscarbrough/$image:latest master:5000/$image:latest
    docker push master:5000/$image:latest
    docker rmi bscarbrough/$image:latest
done

echo "-----------------------------------------------------------------"
echo "|                                                               |"
echo "| Copying necessary range configuration files from resources    |"
echo "| directory to the /range/configs directory                     |"
echo "|                                                               |"
echo "|---------------------------------------------------------------"
for folder in auth recursive
do
    cp
$REPO_HOME/resources/dns/{bind.keys,db.0,db.127,db.255,db.empty,db.local,db.root,rndc.key,zones.r
fc1918} $CONFIG_HOME/dns/$folder
done
cp $REPO_HOME/resources/dns/recursive/named* $CONFIG_HOME/dns/recursive
cp -R $REPO_HOME/resources/monitor $CONFIG_HOME
cp -R $REPO_HOME/resources/router-configs/* $CONFIG_HOME/network
for folder in rtr-01 rtr-02 rtr-03 rtr-04 rtr-05 rtr-06
do
    cp $REPO_HOME/resources/router-configs/daemons* $CONFIG_HOME/network/$folder
done

#cd $REPO_HOME/build/range
# Call build_range_helper.py script to ingest range_services.csv file and build and
# deploy the remaining configurations
#echo "-------------------------------------------------------------"
#echo "|  Building the Kubernetes yaml configuration files     |"
#echo "|  This process can take up to 3-5 minutes to complete  |"
#echo "-------------------------------------------------------------"
#echo ""
#python3 build_k8s_net.py
#python3 build_k8s_deps.py

# Build Router Configs
#echo "-------------------------------------------------------------"
#echo "|  Building the router configuration files zebra.conf and |"
#echo "|  bgpd.conf and copying necessary daemons files to the   |"
```

```
#echo "|  appropriate locations.                                    |"
#echo "------------------------------------------------------------"
#echo ""
#python3 build_rtr_cfgs.py

# Build DNS
echo "------------------------------------------------------------"
echo "|  Generating authoritative DNS entries based on           |"
echo "|  range_services.csv entries. These will serve as the     |"
echo "|  in-range DNS entries.                                   |"
echo "------------------------------------------------------------"
echo ""
python3 build_dns.py range_services.csv

# Build OVS Bridges for overlay network
echo "------------------------------------------------------------"
echo "|  Building OpenVswitch bridges for overlay network        |"
echo "|                                                          |"
echo "| This will need to be run on worker nodes too.            |"
echo "------------------------------------------------------------"
echo ""
#bridges=(bgp rtr1-svc rtr2-svc rtr3-svc rtr4-svc rtr5-svc rtr6-svc external)
sudo ./build_ovs_links.sh -l bgp rtr1-svc rtr2-svc rtr3-svc rtr4-svc rtr5-svc rtr6-svc external

sshpass -e scp -o StrictHostKeyChecking=no build_ovs_links.sh $user@worker01:~/build_ovs_links.sh
echo $SSHPASS | sshpass -e ssh -o StrictHostKeyChecking=no $user@worker01 cat \| sudo --prompt=""
-S -- ./build_ovs_links.sh -l bgp rtr1-svc rtr2-svc rtr3-svc rtr4-svc rtr5-svc rtr6-svc external

sshpass -e scp -o StrictHostKeyChecking=no build_ovs_links.sh $user@worker02:~/build_ovs_links.sh
echo $SSHPASS | sshpass -e ssh -o StrictHostKeyChecking=no $user@worker02 cat \| sudo --prompt=""
-S -- ./build_ovs_links.sh -l bgp rtr1-svc rtr2-svc rtr3-svc rtr4-svc rtr5-svc rtr6-svc external


# Build external access network namespace and configure for network
echo "------------------------------------------------------------"
echo "|  Building Mirror ports for Bro IDS Logging               |"
echo "|                                                          |"
echo "|  Use Kibana to view logged data at the master mgmt IP    |"
echo "|  port 30001                                              |"
echo "------------------------------------------------------------"
echo ""

sudo ./build_mirror.sh external bro0

sshpass -e scp -o StrictHostKeyChecking=no build_mirror.sh $user@worker01:~/build_mirror.sh
echo $SSHPASS | sshpass -e ssh -o StrictHostKeyChecking=no $user@worker01 cat \| sudo --prompt=""
-S -- ./build_mirror.sh bgp bro0

sshpass -e scp -o StrictHostKeyChecking=no build_mirror.sh $user@worker02:~/build_mirror.sh
echo $SSHPASS | sshpass -e ssh -o StrictHostKeyChecking=no $user@worker02 cat \| sudo --prompt=""
-S -- ./build_mirror.sh bgp bro0


# Build external access network namespace and configure for network
echo "------------------------------------------------------------"
echo "|  Creating/configuring external connection namespace      |"
echo "|                                                          |"
echo "------------------------------------------------------------"
echo ""
sudo ovs-vsctl add-port external ext-con -- set Interface ext-con type=internal
sudo ip netns add ext-con-ns
sudo ip link set ext-con netns ext-con-ns
sudo ip netns exec ext-con-ns ip addr add 167.2.126.2/24 dev ext-con
sudo ip netns exec ext-con-ns ip link set ext-con up
sudo ip link set eth3 netns ext-con-ns
sudo ip netns exec ext-con-ns ip addr add 167.2.127.1/24 dev eth3
sudo ip netns exec ext-con-ns ip link set eth3 up
sudo ip netns exec ext-con-ns ip route add default via 167.2.126.1

# Create Multus network attachments for container connections to OVS
kubectl create -f 01-bridge-nets.yml
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
# Build external access network namespace and configure for network
echo "------------------------------------------------------------"
echo "|  Signing website PKI certs. Had to wait this long      |"
echo "|  because if the web directories are created ouside of   |"
echo "|  the scraper, it will fail silently...                  |"
echo "|  That was NOT fun to troubleshoot...                    |"
echo "------------------------------------------------------------"
echo ""
sudo chown -R $user:$user $CONFIG_HOME/web
python3 build_web_pki.py


echo ""
echo "------------------------------------------------------------"
echo "|    ************  CONGRATULATIONS!!  *************        |"
echo "|                                                          |"
echo "|  Your environment is setup and almost ready to go!       |"
echo "|  You will need to wait until the web scraper is finished |"
echo "|  which can take quite a while. Go ahead, take a break    |"
echo "|  go get a nice cup of coffee, or spend some time with    |"
echo "|  friends or family while waiting. Once it is complete,   |"
echo "|  you can come back to it and play around. In the mean    |"
echo "|  you might as well have some fun.                        |"
echo "|                                                          |"
echo "|                                                          |"
echo "|  Once you come back, run the command below and all the   |"
echo "|  K8s systems will run and you can play in your new test  |"
echo "|  environment!                                            |"
echo "|                                                          |"
echo '|    kubectl create -f $K8s_CONFIGS                        |'
echo "|                                                          |"
echo "|  To interact with the environment, connect a VM to the   |"
echo "|  port-group you used for your 'External' bridge.         |"
echo "|  Use the IP information below to route into the range    |"
echo "|  and you can use any of the resources listed in the      |"
echo "|  range_services.csv file.                                |"
echo "|                                                          |"
echo "|   External Network: 167.2.127.0/24                       |"
echo "|                                                          |"
echo "|  OVPN config files are in the \$CONFIG_PATH/vpn. Use the  |"
echo "|  OpenVPN client to connect and VPN services within the   |"
echo "|  range for kicks and giggles - and to see VPN traffic    |"
echo "|  because it's realistic and fun to see if you can learn  |"
echo "|  about the encryption algorithms used, etc.              |"
echo "|                                                          |"
echo "------------------------------------------------------------"
echo ""
```

Appendix Q

build_web_pki.py

```
#!/usr/bin/python3

import csv
import os

svc_cfg_file = "range_services.csv"
web_reader = csv.reader(open(svc_cfg_file))

command = ''
cont_name = ''
for row in web_reader:
    if row[0] != 'svc_category':
        if row[4] == 'web':
            if row[5] == 'http' or row[5] == 'https':
                command = "./sign_cert.sh -s {dom}".format(dom=row[1])

                os.system(command)
```

Appendix R

build_web.py

```
#!/usr/bin/python3

import csv
import os

svc_cfg_file = "range_services.csv"
web_reader = csv.reader(open(svc_cfg_file))

command = ''
cont_name = ''
for row in web_reader:
    if row[0] != 'svc_category':
        if row[4] == 'web':
            if row[5] == 'http' or row[5] == 'https':
                cont_name = row[1].replace('.', '-')
                command = "docker run --rm -i --init -d -e "
                command += "URL={dom} -u root ".format(dom=row[1])
                command += "--cap-add=SYS_ADMIN -v {mnt}:/web/output
".format(mnt=os.environ["CONFIG_HOME"] + "/web")
                command += "-v {scrape_dir}:/data --name get-{dom} master:5000/site-downloader
".format(scrape_dir=os.environ["REPO_HOME"] + "/build/range/scraper",dom=cont_name)
                command += "node /data/scrape.js"

                os.system(command)
                #print(command)
```

Appendix S

sign_cert.sh

```
#!/bin/bash

usage() {
  echo ""
  echo "------------------------------------------------------------------------"
  echo " This script uses a Range generated PKI to sign domains for Range use"
  echo " It accepts 3 different inputs:"
  echo -e "     - Single domain <-s domain.com>"
  echo -e "     - List of domains <-l domain1.com domain2.net domain3.org>"
  echo -e "     - Newline delimted file list of domains <-f filename.txt>"
  echo ""
  echo " Example Usage:"
  echo -e "    $0 -s domain.com"
  echo -e "    $0 -l domain1.com domain2.net"
  echo -e "    $0 -f domainlist.txt"
  echo ""
  exit 1
}

sign-cert() {
  local ca_dir="$PKI_HOME/intermed-ca"
  local web_dir="$CONFIG_HOME/web"
  local domain="$1"
  local dom_ssl_dir="$web_dir/$domain"

  mkdir -p $dom_ssl_dir

  cd "$ca_dir"
  export OPENSSL_CONF="./intermed-ca.cnf"

  openssl req -new -passin file:.passphrase -newkey rsa:2048 -nodes -keyout "$dom_ssl_dir/server-
key.pem" -out certreqs/$domain.csr -subj "/C=US/ST=Georgia/O=Ranges-R-Us, Inc./CN=$domain"
  openssl rand -hex 16 > intermed-ca.serial
  expect -c "
    spawn openssl ca -passin file:.passphrase -in certreqs/${domain}.csr -out
certs/${domain}.cert.pem -extensions server_ext
    expect \"Sign the certificate\" {
    send \"y\r\"

    expect \"1 out of 1 certificate requests certified\"
    send \"y\r\"
    exp_continue
    }
  "
  cp "$ca_dir/certs/$domain.cert.pem" "$dom_ssl_dir/server.pem"
}

if [ $# -gt 1 ]
then
  case $1 in
    # Process newline delimited file of domains
    -f | --file )
      shift
      while read dom
      do
        sign-cert "$dom"
      done < "$1"
      ;;
    # Process single domain
    -s | --site )
      shift
      sign-cert "$1"
      ;;
    # Process list of domains from CLI
```

```
    -l | --list )
      shift
      while [ "$1" != "" ]
      do
        sign-cert "$1"
        shift
      done
      ;;
    -h | --help )
      usage
      ;;
  esac
else
  usage
fi

exit 0
```

Appendix T

range_services.csv

```
svc_category,domain_name,ip_addr,svc_bridge,svc_type,svc_sub_type,base_domain,k8s_name,container_
name
services,ac1.nstld.com,192.42.173.30,rtr1-svc,dns,authoritative,nstld.com,dns-ac1-nstld,bind
services,ac2.nstld.com,192.42.174.30,rtr1-svc,dns,authoritative,nstld.com,dns-ac2-nstld,bind
services,a.nic.dns-tld.site,37.209.192.9,rtr2-svc,dns,authoritative,dns-tld.site,dns-a-nic,bind
services,b.nic.dns-tld.site,37.209.194.9,rtr2-svc,dns,authoritative,dns-tld.site,dns-b-nic,bind
services,google-public-dns-a.google.com,8.8.8.8,rtr1-svc,dns,recursive,google.com,dns-google-
a,bind
services,google-public-dns-b.google.com,8.8.4.4,rtr1-svc,dns,recursive,google.com,dns-google-
b,bind
services,dns.quad9.net,9.9.9.9,rtr1-svc,dns,recursive,quad9.net,dns-quad9-01,bind
services,dns.quad9.net,149.112.112.112,rtr1-svc,dns,recursive,quad9.net,dns-quad9-02,bind
services,google.com,172.217.0.78,rtr1-svc,web,https,google.com,www-google,nginx
services,microsoft.com,40.76.4.15,rtr1-svc,web,https,microsoft.com,www-microsoft,nginx
services,wikipedia.org,208.80.154.224,rtr1-svc,web,https,wikipedia.org,www-wikipedia,nginx
services,elsalvador.com,179.249.122.52,rtr5-svc,web,https,elsalvador.com,www-elsalvador,nginx
services,elgenero.com,201.23.113.243,rtr5-svc,web,https,elgenero.com,www-elgenero,nginx
services,facebook.com,31.13.65.36,rtr2-svc,web,https,facebook.com,www-facebook,nginx
services,yandex.ru,77.88.55.55,rtr2-svc,web,https,yandex.ru,www-yandex,nginx
services,rtl.lu,81.92.237.71,rtr2-svc,web,https,rtl.lu,www-rtl,nginx
services,standardmedia.co.ke,105.20.205.11,rtr6-svc,web,https,standardmedia.co.ke,www-
standardmedia,nginx
services,fakaza.com,154.162.225.76,rtr6-svc,web,https,fakaza.com,www-fakaza,nginx
services,livedoor.com,202.104.153.16,rtr3-svc,web,https,livedoor.com,www-livedoor,nginx
services,sabq.org,105.16.44.64,rtr3-svc,web,https,sabq.org,www-sabaq,nginx
services,baidu.com,123.125.114.144,rtr3-svc,web,https,baidu.com,www-baidu,nginx
services,ebay.com.au,58.135.195.175,rtr4-svc,web,https,ebay.com.au,www-ebay,nginx
services,trademe.co.nz,203.162.72.2,rtr4-svc,web,https,trademe.co.nz,www-trademe,nginx
services,asia.pool.ntp.org,49.19.96.19,rtr3-svc,ntp,time,pool.ntp.org,ntp-asia,ntpd
services,europe.pool.ntp.org,89.120.166.8,rtr2-svc,ntp,time,pool.ntp.org,ntp-europe,ntpd
services,time.nist.gov,129.6.15.28,rtr1-svc,ntp,time,nist.gov,ntp-nist,ntpd
services,pool.ntp.org,107.15.121.121,rtr1-svc,ntp,time,pool.ntp.org,ntp-pool,ntpd
services,aspmx.l.google.com,74.125.21.26,rtr1-svc,smtp,mail,google.com,smtp-google,smtp
services,microsoft-com.mail.protection.outlook.com,104.47.54.36,rtr1-
svc,smtp,mail,outlook.com,smtp-microsoft,smtp
services,smtpin.vvv.facebook.com,176.252.127.251,rtr2-svc,smtp,mail,facebook.com,smtp-
facebook,smtp
services,mx.maillb.baidu.com,12.0.243.41,rtr1-svc,smtp,mail,baidu.com,smtp-baidu,smtp
services,mail.email.com,68.178.213.60,rtr1-svc,smtp,webmail,email.com,mail-webmail,webmail
obfuscation,vpn.com,34.202.89.42,rtr1-svc,vpn,vpn,vpn.com,vpn-vpn,openvpn
obfuscation,ausvpn.com,150.1.79.230,rtr4-svc,vpn,vpn,ausvpn.com,vpn-ausvpn,openvpn
obfuscation,expressvpn.com,31.32.247.106,rtr2-svc,vpn,vpn,expressvpn.com,vpn-expressvpn,openvpn
obfuscation,nordvpn.com,106.18.229.229,rtr3-svc,vpn,vpn,nordvpn.com,vpn-nordvpn,openvpn
obfuscation,privatevpn.com,196.24.4.13,rtr6-svc,vpn,vpn,privatevpn.com,vpn-privatevpn,openvpn
obfuscation,ipvanish.com,201.185.216.42,rtr3-svc,vpn,vpn,ipvanish.com,vpn-ipvanish,openvpn
attack,rtr1-attacker,73.103.29.155,rtr1-svc,attack,metasploit,,attack-rtr1,metasploit
attack,rtr6-attacker,105.114.32.193,rtr6-svc,attack,metasploit,,attack-rtr6,metasploit
attack,rtr5-attacker,187.10.12.64,rtr5-svc,attack,metasploit,,attack-rtr2,metasploit
attack,rtr4-attacker,114.21.3.146,rtr4-svc,attack,metasploit,,attack-rtr3,metasploit
attack,rtr3-attacker,211.121.232.14,rtr3-svc,attack,kali,,attack-rtr5,kali
attack,rtr2-attacker,145.123.12.133,rtr2-svc,attack,kali,,attack-rtr4,kali
vulnerable,wordpress.org,198.143.164.252,rtr1-svc,web,wordpress,wordpress.org,vuln-
wordpress,vuln-wordpress
vulnerable,vulnerable.org,128.143.18.14,rtr1-svc,web,metasploit-vuln-svc,vulnerable.org,vuln-
org,metasploit-vuln-svc-emu
vulnerable,hackme.co.uk,83.129.43.12,rtr2-svc,web,metasploit-vuln-svc,hackme.co.uk,vuln-hackme-
co-uk,metasploit-vuln-svc-emu
vulnerable,imvulnerable.net,103.112.1.3,rtr4-svc,web,metasploit-vuln-svc,imvulnerable.net,vuln-
imvulnerable-net,metasploit-vuln-svc-emu
vulnerable,hackattack.com,102.120.3.42,rtr6-svc,web,metasploit-vuln-svc,hackattack.com,vuln-
hackattack-com,metasploit-vuln-svc-emu
services,ftp.cisco.com,72.163.4.185,rtr1-svc,ftp,ftps,cisco.com,ftp-cisco,ftpd
services,ftp.adobe.com,193.104.215.58,rtr2-svc,ftp,ftps,adobe.com,ftp-adobe,ftpd
services,ftp.malwr.cn,61.62.202.101,rtr3-svc,ftp,ftp,malwr.cn,ftp-malwr,ftpd
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
services,ftp.music.ru,212.2.123.43,rtr2-svc,ftp,ftp,music.ru,ftp-music,ftpd
services,stream.com,184.29.214.127,rtr1-svc,web,media,stream.com,media-stream,media
vulnerable,muteme.com,41.0.12.15,rtr6-svc,web,mutillidae,muteme.com,vuln-mutillidae,vuln-
mutillidae
```

Appendix U

bind.keys

```
# The bind.keys file is used to override the built-in DNSSEC trust anchors
# which are included as part of BIND 9.  The only trust anchors it contains
# are for the DNS root zone (".").  Trust anchors for any other zones MUST
# be configured elsewhere; if they are configured here, they will not be
# recognized or used by named.
#
# The built-in trust anchors are provided for convenience of configuration.
# They are not activated within named.conf unless specifically switched on.
# To use the built-in key, use "dnssec-validation auto;" in the
# named.conf options.  Without this option being set, the keys in this
# file are ignored.
#
# This file is NOT expected to be user-configured.
#
# These keys are current as of October 2017.  If any key fails to
# initialize correctly, it may have expired.  In that event you should
# replace this file with a current version.  The latest version of
# bind.keys can always be obtained from ISC at https://www.isc.org/bind-keys.
#
# See https://data.iana.org/root-anchors/root-anchors.xml
# for current trust anchor information for the root zone.

managed-keys {
        # This key (19036) is to be phased out starting in 2017. It will
        # remain in the root zone for some time after its successor key
        # has been added. It will remain this file until it is removed from
        # the root zone.
        . initial-key 257 3 8 "AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQbSEW0O8gcCjF
                FVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh/RStIoO8g0NfnfL2MTJRkxoX
                bfDaUeVPQuYEhg37NZWAJQ9VnMVDxP/VHL496M/QZxkjf5/Efucp2gaD
                X6RS6CXpoY68LsvPVjR0ZSwzz1apAzvN9dlzEheX7ICJBBtuA6G3LQpz
                W5hOA2hzCTMjJPJ8LbqF6dsV6DoBQzgul0sGIcGOYl7OyQdXfZ57relS
                Qageu+ipAdTTJ25AsRTAoub8ONGcLmqrAmRLKBP1dfwhYB4N7knNnulq
                QxA+Uk1ihz0=";

        # This key (20326) was published in the root zone in 2017.
        # Servers which were already using the old key (19036) should
        # roll seamlessly to this new one via RFC 5011 rollover. Servers
        # being set up for the first time can use the contents of this
        # file as initializing keys; thereafter, the keys in the
        # managed key database will be trusted and maintained
        # automatically.
        . initial-key 257 3 8 "AwEAAaz/tAm8yTn4Mfeh5eyI96WSVexTBAvkMgJzkKTOiW1vkIbzxeF3
                +/4RgWOq7HrxRixHlFlExOLAJr5emLvN7SWXgnLh4+B5xQlNVz8Og8kv
                ArMtNROxVQuCaSnIDdD5LKyWbRd2n9WGe2R8PzgCmr3EgVLrjyBxWezF
                0jLHwVN8efS3rCj/EWgvIWgb9tarpVUDK/b58Da+sqqls3eNbuv7pr+e
                oZG+SrDK6nWeL3c6H5Apxz7LjVc1uTIdsIXxuOLYA4/ilBmSVIzuDWfd
                RUfhHdY6+cn8HFRm+2hM8AnXGXws9555KrUB5qihylGa8subX2Nn6UwN
                R1AkUTV74bU=";
};
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix V

db.0

```
;
; BIND reverse data file for broadcast zone
;
$TTL      604800
@         IN        SOA       localhost. root.localhost. (
                                      1            ; Serial
                                 604800            ; Refresh
                                  86400            ; Retry
                                2419200            ; Expire
                                 604800 ) ; Negative Cache TTL
;
@         IN        NS        localhost.
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix W

db.127

```
;
; BIND reverse data file for local loopback interface
;
$TTL      604800
@         IN        SOA       localhost. root.localhost. (
                                      1            ; Serial
                                 604800            ; Refresh
                                  86400            ; Retry
                                2419200            ; Expire
                                 604800 ) ; Negative Cache TTL
;
@         IN        NS        localhost.
1.0.0     IN        PTR       localhost.
```

Appendix X

db.255

```
;
; BIND reverse data file for broadcast zone
;
$TTL      604800
@         IN        SOA       localhost. root.localhost. (
                                      1            ; Serial
                                 604800            ; Refresh
                                  86400            ; Retry
                                2419200            ; Expire
                                 604800 ) ; Negative Cache TTL
;
@         IN        NS        localhost.
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix Y

db.empty

```
; BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL      86400
@         IN        SOA       localhost. root.localhost. (
                                      1            ; Serial
                                 604800            ; Refresh
                                  86400            ; Retry
                                2419200            ; Expire
                                  86400 ) ; Negative Cache TTL
;
@         IN        NS        localhost.
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix Z

db.local

```
;
; BIND data file for local loopback interface
;
$TTL      604800
@         IN        SOA       localhost. root.localhost. (
                                    2            ; Serial
                                604800            ; Refresh
                                 86400            ; Retry
                               2419200            ; Expire
                                604800 ) ; Negative Cache TTL
;
@         IN        NS        localhost.
@         IN        A         127.0.0.1
@         IN        AAAA      ::1
```

Appendix AA

db.root

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC
;       under anonymous FTP as
;           file                /domain/named.cache
;           on server           FTP.INTERNIC.NET
;       -OR-                    RS.INTERNIC.NET
;
;       last update:    February 17, 2016
;       related version of root zone:   2016021701
;
; formerly NS.INTERNIC.NET
;
.                        3600000      NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
A.ROOT-SERVERS.NET.      3600000      AAAA  2001:503:ba3e::2:30
;
; FORMERLY NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     192.228.79.201
B.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:84::b
;
; FORMERLY C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
C.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:2::c
;
; FORMERLY TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     199.7.91.13
D.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:2d::d
;
; FORMERLY NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; FORMERLY NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
F.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:2f::f
;
; FORMERLY NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; FORMERLY AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     198.97.190.53
H.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:1::53
;
; FORMERLY NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
I.ROOT-SERVERS.NET.      3600000      AAAA  2001:7fe::53
;
; OPERATED BY VERISIGN, INC.
;
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     192.58.128.30
J.ROOT-SERVERS.NET.      3600000      AAAA  2001:503:c27::2:30
;
; OPERATED BY RIPE NCC
;
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     193.0.14.129
K.ROOT-SERVERS.NET.      3600000      AAAA  2001:7fd::1
;
; OPERATED BY ICANN
;
.                        3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000      A     199.7.83.42
L.ROOT-SERVERS.NET.      3600000      AAAA  2001:500:3::42
;
; OPERATED BY WIDE
;
.                        3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A     202.12.27.33
M.ROOT-SERVERS.NET.      3600000      AAAA  2001:dc3::35
; End of file
```

Appendix BB

rndc.key

```
key "rndc-key" {
          algorithm hmac-md5;
          secret "ybPGtV7tLHyYS/WQ1Mbw/A==";
};
```

Appendix CC

zones.rfc1918

```
zone "10.in-addr.arpa"      { type master; file "/etc/bind/db.empty"; };

zone "16.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "17.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "18.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "19.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "20.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "21.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "22.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "23.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "24.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "25.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "26.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "27.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "28.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "29.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "30.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };
zone "31.172.in-addr.arpa"  { type master; file "/etc/bind/db.empty"; };

zone "168.192.in-addr.arpa" { type master; file "/etc/bind/db.empty"; };
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix DD

named.conf

```
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Appendix EE

named.conf.default-zones

```
// prime the server with knowledge of the root servers
zone "." {
          type hint;
          file "/etc/bind/db.root";
};

// be authoritative for the localhost forward and reverse zones, and for
// broadcast zones as per RFC 1912

zone "localhost" {
          type master;
          file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
          type master;
          file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
          type master;
          file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
          type master;
          file "/etc/bind/db.255";
};
```

Appendix FF

named.conf.local

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
```

Appendix GG

named.conf.options

```
options {
         directory "/var/cache/bind";

         // If there is a firewall between you and nameservers you want
         // to talk to, you may need to fix the firewall to allow multiple
         // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

         // If your ISP provided one or more IP addresses for stable
         // nameservers, you probably want to use them as forwarders.
         // Uncomment the following block, and insert the addresses replacing
         // the all-0's placeholder.
         listen-on port 53 { any; };
         recursion yes;
         allow-recursion { any; };
        forward only;
        forwarders {
                        37.209.192.9;
                        192.42.173.30;
                        37.209.194.9;
                        192.42.174.30;
        };

         //========================================================================
         // If BIND logs error messages about the root key being expired,
         // you will need to update your keys.  See https://www.isc.org/bind-keys
         //========================================================================
        dnssec-enable no;
        dnssec-validation no;
        dnssec-lookaside auto;

         auth-nxdomain no;    # conform to RFC1035
         listen-on-v6 { any; };
};
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix HH

attack.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr1
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: attack-rtr1
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 73.103.29.155
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 73.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit
        name: attack-rtr1
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/msf/.msf4
          name: rtr1-attacker
      initContainers: []
      volumes:
      - name: rtr1-attacker
        nfs:
          path: /configs/attack/rtr1-attacker
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr2
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr2
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr5-svc
      labels:
        app: attack-rtr2
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 187.10.12.64
        - name: LEN
```

```
          value: '8'
        - name: GATEWAY
          value: 187.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit
        name: attack-rtr2
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/msf/.msf4
          name: rtr5-attacker
      initContainers: []
      volumes:
      - name: rtr5-attacker
        nfs:
          path: /configs/attack/rtr5-attacker
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr3
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr3
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr4-svc
      labels:
        app: attack-rtr3
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 114.21.3.146
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 114.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit
        name: attack-rtr3
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/msf/.msf4
          name: rtr4-attacker
      initContainers: []
      volumes:
      - name: rtr4-attacker
        nfs:
          path: /configs/attack/rtr4-attacker
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr4
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr4
  template:
```

```
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: attack-rtr4
    spec:
      containers:
      - command:
        - /bin/bash
        - -c
        args:
        - |-
          sleep 100000
        env:
        - name: IP_ADDR
          value: 145.123.12.133
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 145.0.0.1
        - name: INT
          value: net1
        image: master:5000/kali
        name: attack-rtr4
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /root
          name: rtr2-attacker
      initContainers: []
      volumes:
      - name: rtr2-attacker
        nfs:
          path: /configs/attack/rtr2-attacker
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr5
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr5
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: attack-rtr5
    spec:
      containers:
      - command:
        - /bin/bash
        - -c
        args:
        - |-
          sleep 10000
        env:
        - name: IP_ADDR
          value: 211.121.232.14
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 211.0.0.1
        - name: INT
          value: net1
        image: master:5000/kali
        name: attack-rtr5
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /root
          name: rtr3-attacker
      initContainers: []
      volumes:
      - name: rtr3-attacker
        nfs:
          path: /configs/attack/rtr3-attacker
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: attack-rtr6
spec:
  replicas: 1
  selector:
    matchLabels:
      app: attack-rtr6
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: attack-rtr6
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 105.114.32.193
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 105.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit
        name: attack-rtr6
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/msf/.msf4
          name: rtr6-attacker
      initContainers: []
      volumes:
      - name: rtr6-attacker
        nfs:
          path: /configs/attack/rtr6-attacker
          server: storage
```

Appendix II

bro-elk.yml

```
---
apiVersion: v1
kind: Service
metadata:
  name: elasticsearch
spec:
  ports:
  - name: "9200"
    port: 9200
    targetPort: 9200
  - name: "9300"
    port: 9300
    targetPort: 9300
  selector:
    app: elasticsearch
---
apiVersion: v1
kind: Service
metadata:
  name: kibana
spec:
  ports:
  - name: "5601"
    port: 5601
    targetPort: 5601
    nodePort: 30001
  type: NodePort
  selector:
    app: kibana
---
apiVersion: v1
kind: Service
metadata:
  name: logstash
spec:
  ports:
  - name: "5000"
    port: 5000
    targetPort: 5000
  selector:
    app: logstash
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: bro-ids
spec:
  selector:
    matchLabels:
      name: bro-ids
  template:
    metadata:
      labels:
        name: bro-ids
    spec:
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      hostNetwork: true
      containers:
      - args:
         - -i
         - bro0
        image: blacktop/bro
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
        name: bro-ids
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /pcap
          name: bro-ids
          subPath: data
        - mountPath: /usr/local/share/bro/site
          name: bro-ids
          subPath: config
      restartPolicy: Always
      volumes:
      - name: bro-ids
        nfs:
          server: storage
          path: /configs/monitor/bro
---
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: elasticsearch
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: elasticsearch
    spec:
      containers:
      - env:
        - name: ES_JAVA_OPTS
          value: -Xmx256m -Xms256m
        image: master:5000/elasticsearch
        name: elasticsearch
        ports:
        - containerPort: 9200
        - containerPort: 9300
        volumeMounts:
        - mountPath: /usr/share/elasticsearch/config/elasticsearch.yml
          name: elasticsearch
          subPath: elasticsearch.yml
          readOnly: true
        - mountPath: /usr/share/elasticsearch/data
          name: elasticsearch
          subPath: data
      restartPolicy: Always
      volumes:
      - name: elasticsearch
        nfs:
          server: storage
          path: /configs/monitor/elasticsearch
---
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: kibana
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: kibana
    spec:
      containers:
      - image: master:5000/kibana
        name: kibana
        ports:
```

```
        - containerPort: 5601
        volumeMounts:
        - mountPath: /usr/share/kibana/config
          name: kibana-config
          readOnly: true
      restartPolicy: Always
      volumes:
      - name: kibana-config
        nfs:
          server: storage
          path: /configs/monitor/kibana
---
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: logstash
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: logstash
    spec:
      containers:
      - env:
        - name: LS_JAVA_OPTS
          value: -Xmx256m -Xms256m
        image: master:5000/logstash
        name: logstash
        ports:
        - containerPort: 5000
        volumeMounts:
        - mountPath: /usr/share/logstash/config/logstash.yml
          name: logstash
          subPath: config/logstash.yml
          readOnly: true
        - mountPath: /usr/share/logstash/config/geo-ip.json
          name: logstash
          subPath: config/geo-ip.json
          readOnly: true
        - mountPath: /usr/share/logstash/pipeline
          name: logstash
          subPath: pipeline
          readOnly: true
        - mountPath: /usr/share/logstash/bro
          name: logstash-bro
          readOnly: true
        - mountPath: /etc/logstash/custom_patterns
          name: logstash
          subPath: patterns
      restartPolicy: Always
      volumes:
      - name: logstash
        nfs:
          server: storage
          path: /configs/monitor/logstash
      - name: logstash-bro
        nfs:
          server: storage
          path: /configs/monitor/bro/data
```

Appendix JJ

dns.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-a-nic
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-a-nic
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: dns-a-nic
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 37.209.192.9
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 37.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-a-nic
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: a-nic-dns-tld-site
      initContainers: []
      volumes:
      - name: a-nic-dns-tld-site
        nfs:
          path: /configs/dns/auth
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-ac1-nstld
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-ac1-nstld
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-ac1-nstld
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 192.42.173.30
        - name: LEN
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
          value: '8'
        - name: GATEWAY
          value: 192.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-ac1-nstld
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: ac1-nstld-com
      initContainers: []
      volumes:
      - name: ac1-nstld-com
        nfs:
          path: /configs/dns/auth
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-ac2-nstld
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-ac2-nstld
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-ac2-nstld
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 192.42.174.30
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 192.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-ac2-nstld
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: ac2-nstld-com
      initContainers: []
      volumes:
      - name: ac2-nstld-com
        nfs:
          path: /configs/dns/auth
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-b-nic
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-b-nic
  template:
```

```
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: dns-b-nic
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 37.209.194.9
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 37.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-b-nic
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: b-nic-dns-tld-site
      initContainers: []
      volumes:
      - name: b-nic-dns-tld-site
        nfs:
          path: /configs/dns/auth
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-google-a
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-google-a
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-google-a
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 8.8.8.8
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 8.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-google-a
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: google-public-dns-a-google-com
      initContainers: []
      volumes:
      - name: google-public-dns-a-google-com
        nfs:
          path: /configs/dns/recursive
          server: storage
---
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-google-b
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-google-b
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-google-b
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 8.8.4.4
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 8.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-google-b
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: google-public-dns-b-google-com
      initContainers: []
      volumes:
      - name: google-public-dns-b-google-com
        nfs:
          path: /configs/dns/recursive
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-quad9-01
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-quad9-01
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-quad9-01
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 9.9.9.9
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 9.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-quad9-01
        securityContext:
```

```
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: dns-quad9-net
      initContainers: []
      volumes:
      - name: dns-quad9-net
        nfs:
          path: /configs/dns/recursive
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: dns-quad9-02
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dns-quad9-02
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: dns-quad9-02
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 149.112.112.112
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 149.0.0.1
        - name: INT
          value: net1
        image: master:5000/bind
        name: dns-quad9-02
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/bind
          name: dns-quad9-net
      initContainers: []
      volumes:
      - name: dns-quad9-net
        nfs:
          path: /configs/dns/recursive
          server: storage
```

Appendix KK

ftp.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ftp-adobe
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ftp-adobe
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: ftp-adobe
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 193.104.215.58
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 193.0.0.1
        - name: INT
          value: net1
        - name: PUBLICHOST
          value: ftp.adobe.com
        - name: FTP_USER_NAME
          value: ftpuser
        - name: FTP_USER_PASS
          value: ftp123
        - name: FTP_USER_HOME
          value: /home/ftpuser
        image: master:5000/ftpd
        name: ftp-adobe
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/ftpuser
          name: ftp-adobe-com
      initContainers: []
      volumes:
      - name: ftp-adobe-com
        nfs:
          path: /configs/ftp/ftp.adobe.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ftp-cisco
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ftp-cisco
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
      labels:
        app: ftp-cisco
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 72.163.4.185
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 72.0.0.1
        - name: INT
          value: net1
        - name: PUBLICHOST
          value: ftp.cisco.com
        - name: FTP_USER_NAME
          value: ftpuser
        - name: FTP_USER_PASS
          value: ftp123
        - name: FTP_USER_HOME
          value: /home/ftpuser
        image: master:5000/ftpd
        name: ftp-cisco
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/ftpuser
          name: ftp-cisco-com
      initContainers: []
      volumes:
      - name: ftp-cisco-com
        nfs:
          path: /configs/ftp/ftp.cisco.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ftp-malwr
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ftp-malwr
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: ftp-malwr
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 61.62.202.101
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 61.0.0.1
        - name: INT
          value: net1
        - name: PUBLICHOST
          value: ftp.malwr.cn
        - name: FTP_USER_NAME
          value: ftpuser
        - name: FTP_USER_PASS
          value: ftp123
        - name: FTP_USER_HOME
          value: /home/ftpuser
        image: master:5000/ftpd
```

```
        name: ftp-malwr
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/ftpuser
          name: ftp-malwr-cn
      initContainers: []
      volumes:
      - name: ftp-malwr-cn
        nfs:
          path: /configs/ftp/ftp.malwr.cn
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ftp-music
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ftp-music
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: ftp-music
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 212.2.123.43
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 212.0.0.1
        - name: INT
          value: net1
        - name: PUBLICHOST
          value: ftp.music.ru
        - name: FTP_USER_NAME
          value: ftpuser
        - name: FTP_USER_PASS
          value: ftp123
        - name: FTP_USER_HOME
          value: /home/ftpuser
        image: master:5000/ftpd
        name: ftp-music
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /home/ftpuser
          name: ftp-music-ru
      initContainers: []
      volumes:
      - name: ftp-music-ru
        nfs:
          path: /configs/ftp/ftp.music.ru
          server: storage
```

Appendix LL

webmail.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: mail-webmail
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mail-webmail
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: mail-webmail
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 68.178.213.60
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 68.0.0.1
        - name: INT
          value: net1
        image: master:5000/webmail
        name: mail-webmail
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /data
          name: mail-email-com
      initContainers: []
      volumes:
      - name: mail-email-com
        nfs:
          path: /configs/smtp/webmail
          server: storage
```

Appendix MM

media.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: media-stream
spec:
  replicas: 1
  selector:
    matchLabels:
      app: media-stream
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: media-stream
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 184.29.214.127
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 184.0.0.1
        - name: INT
          value: net1
        image: master:5000/media
        name: media-stream
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /media/serviio
          name: stream-com
      initContainers: []
      volumes:
      - name: stream-com
        nfs:
          path: /configs/media/stream.com
          server: storage
```

Appendix NN

ntp.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ntp-asia
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ntp-asia
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: ntp-asia
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 49.19.96.19
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 49.0.0.1
        - name: INT
          value: net1
        image: master:5000/ntpd
        name: ntp-asia
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ntp-europe
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ntp-europe
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: ntp-europe
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 89.120.166.8
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 89.0.0.1
        - name: INT
          value: net1
        image: master:5000/ntpd
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
        name: ntp-europe
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ntp-nist
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ntp-nist
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: ntp-nist
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 129.6.15.28
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 129.0.0.1
        - name: INT
          value: net1
        image: master:5000/ntpd
        name: ntp-nist
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: ntp-pool
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ntp-pool
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: ntp-pool
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 107.15.121.121
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 107.0.0.1
        - name: INT
          value: net1
        image: master:5000/ntpd
        name: ntp-pool
```

```
  securityContext:
    privileged: true
  volumeMounts: []
initContainers: []
volumes: []
```

Appendix OO

router.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rtr-01
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-01
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, bgp, bgp, rtr1-svc, external
      labels:
        app: rtr-01
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-01
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-01
      initContainers: []
      volumes:
      - name: rtr-01
        nfs:
          path: /configs/network/rtr-01
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rtr-02
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-02
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, bgp, bgp, rtr2-svc, external
      labels:
        app: rtr-02
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-02
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-02
      initContainers: []
      volumes:
      - name: rtr-02
        nfs:
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
          path: /configs/network/rtr-02
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rtr-03
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-03
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, bgp, rtr3-svc, external
      labels:
        app: rtr-03
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-03
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-03
      initContainers: []
      volumes:
      - name: rtr-03
        nfs:
          path: /configs/network/rtr-03
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rtr-04
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-04
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, bgp, rtr4-svc, external
      labels:
        app: rtr-04
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-04
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-04
      initContainers: []
      volumes:
      - name: rtr-04
        nfs:
          path: /configs/network/rtr-04
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
```

```
kind: Deployment
metadata:
  name: rtr-05
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-05
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, rtr5-svc, external
      labels:
        app: rtr-05
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-05
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-05
      initContainers: []
      volumes:
      - name: rtr-05
        nfs:
          path: /configs/network/rtr-05
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rtr-06
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtr-06
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: bgp, rtr6-svc, external
      labels:
        app: rtr-06
    spec:
      containers:
      - env: []
        image: master:5000/frr
        name: rtr-06
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/frr
          name: rtr-06
      initContainers: []
      volumes:
      - name: rtr-06
        nfs:
          path: /configs/network/rtr-06
          server: storage
```

Appendix PP

smtp.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: smtp-baidu
spec:
  replicas: 1
  selector:
    matchLabels:
      app: smtp-baidu
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: smtp-baidu
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 12.0.243.41
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 12.0.0.1
        - name: INT
          value: net1
        image: master:5000/smtp
        name: smtp-baidu
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: smtp-facebook
spec:
  replicas: 1
  selector:
    matchLabels:
      app: smtp-facebook
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: smtp-facebook
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 176.252.127.251
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 176.0.0.1
        - name: INT
          value: net1
        image: master:5000/smtp
```

```
        name: smtp-facebook
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: smtp-google
spec:
  replicas: 1
  selector:
    matchLabels:
      app: smtp-google
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: smtp-google
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 74.125.21.26
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 74.0.0.1
        - name: INT
          value: net1
        image: master:5000/smtp
        name: smtp-google
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: smtp-microsoft
spec:
  replicas: 1
  selector:
    matchLabels:
      app: smtp-microsoft
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: smtp-microsoft
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 104.47.54.36
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 104.0.0.1
        - name: INT
          value: net1
        image: master:5000/smtp
        name: smtp-microsoft
```

```
    securityContext:
      privileged: true
    volumeMounts: []
  initContainers: []
  volumes: []
```

Appendix QQ

vpn.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-ausvpn
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-ausvpn
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr4-svc
      labels:
        app: vpn-ausvpn
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 150.1.79.230
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 150.0.0.1
        - name: INT
          value: net1
        - name: OPEN_VPN_CMD
          value: ovpn_run
        image: master:5000/openvpn
        name: vpn-ausvpn
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ausvpn-com
      initContainers:
      - command:
        - ovpn_genconfig
        - -u
        - udp://ausvpn.com
        image: master:5000/openvpn
        name: vpn-ausvpn-ausvpn-genconfig
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ausvpn-com
      - command:
        - ovpn_initpki
        - nopass
        image: master:5000/openvpn
        name: vpn-ausvpn-ausvpn-initpki
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ausvpn-com
      - command:
        - easyrsa
        - build-client-full
        - ausvpn
        - nopass
        image: master:5000/openvpn
        name: vpn-ausvpn-ausvpn-build-client
        volumeMounts:
        - mountPath: /etc/openvpn
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
          name: ausvpn-com
      - command:
        - ovpn_getclient
        - ausvpn
        - '>'
        - /configs/vpn/ausvpn.ovpn
        image: master:5000/openvpn
        name: vpn-ausvpn-ausvpn-get-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ausvpn-com
      volumes:
      - name: ausvpn-com
        nfs:
          path: /configs/vpn/ausvpn.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-expressvpn
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-expressvpn
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: vpn-expressvpn
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 31.32.247.106
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 31.0.0.1
        - name: INT
          value: net1
        - name: OPEN_VPN_CMD
          value: ovpn_run
        image: master:5000/openvpn
        name: vpn-expressvpn
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/openvpn
          name: expressvpn-com
      initContainers:
      - command:
        - ovpn_genconfig
        - -u
        - udp://expressvpn.com
        image: master:5000/openvpn
        name: vpn-expressvpn-expressvpn-genconfig
        volumeMounts:
        - mountPath: /etc/openvpn
          name: expressvpn-com
      - command:
        - ovpn_initpki
        - nopass
        image: master:5000/openvpn
        name: vpn-expressvpn-expressvpn-initpki
        volumeMounts:
        - mountPath: /etc/openvpn
          name: expressvpn-com
```

```
      - command:
        - easyrsa
        - build-client-full
        - expressvpn
        - nopass
        image: master:5000/openvpn
        name: vpn-expressvpn-expressvpn-build-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: expressvpn-com
      - command:
        - ovpn_getclient
        - expressvpn
        - '>'
        - /configs/vpn/expressvpn.ovpn
        image: master:5000/openvpn
        name: vpn-expressvpn-expressvpn-get-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: expressvpn-com
      volumes:
      - name: expressvpn-com
        nfs:
          path: /configs/vpn/expressvpn.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-ipvanish
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-ipvanish
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: vpn-ipvanish
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 201.185.216.42
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 201.0.0.1
        - name: INT
          value: net1
        - name: OPEN_VPN_CMD
          value: ovpn_run
        image: master:5000/openvpn
        name: vpn-ipvanish
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ipvanish-com
      initContainers:
      - command:
        - ovpn_genconfig
        - -u
        - udp://ipvanish.com
        image: master:5000/openvpn
        name: vpn-ipvanish-ipvanish-genconfig
        volumeMounts:
        - mountPath: /etc/openvpn
```

```
          name: ipvanish-com
      - command:
        - ovpn_initpki
        - nopass
        image: master:5000/openvpn
        name: vpn-ipvanish-ipvanish-initpki
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ipvanish-com
      - command:
        - easyrsa
        - build-client-full
        - ipvanish
        - nopass
        image: master:5000/openvpn
        name: vpn-ipvanish-ipvanish-build-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ipvanish-com
      - command:
        - ovpn_getclient
        - ipvanish
        - '>'
        - /configs/vpn/ipvanish.ovpn
        image: master:5000/openvpn
        name: vpn-ipvanish-ipvanish-get-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: ipvanish-com
      volumes:
      - name: ipvanish-com
        nfs:
          path: /configs/vpn/ipvanish.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-nordvpn
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-nordvpn
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: vpn-nordvpn
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 106.18.229.229
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 106.0.0.1
        - name: INT
          value: net1
        - name: OPEN_VPN_CMD
          value: ovpn_run
        image: master:5000/openvpn
        name: vpn-nordvpn
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/openvpn
          name: nordvpn-com
```

```
      initContainers:
      - command:
        - ovpn_genconfig
        - -u
        - udp://nordvpn.com
        image: master:5000/openvpn
        name: vpn-nordvpn-nordvpn-genconfig
        volumeMounts:
        - mountPath: /etc/openvpn
          name: nordvpn-com
      - command:
        - ovpn_initpki
        - nopass
        image: master:5000/openvpn
        name: vpn-nordvpn-nordvpn-initpki
        volumeMounts:
        - mountPath: /etc/openvpn
          name: nordvpn-com
      - command:
        - easyrsa
        - build-client-full
        - nordvpn
        - nopass
        image: master:5000/openvpn
        name: vpn-nordvpn-nordvpn-build-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: nordvpn-com
      - command:
        - ovpn_getclient
        - nordvpn
        - '>'
        - /configs/vpn/nordvpn.ovpn
        image: master:5000/openvpn
        name: vpn-nordvpn-nordvpn-get-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: nordvpn-com
      volumes:
      - name: nordvpn-com
        nfs:
          path: /configs/vpn/nordvpn.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-privatevpn
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-privatevpn
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: vpn-privatevpn
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 196.24.4.13
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 196.0.0.1
        - name: INT
          value: net1
```

```
        - name: OPEN_VPN_CMD
          value: ovpn_run
        image: master:5000/openvpn
        name: vpn-privatevpn
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /etc/openvpn
          name: privatevpn-com
      initContainers:
      - command:
        - ovpn_genconfig
        - -u
        - udp://privatevpn.com
        image: master:5000/openvpn
        name: vpn-privatevpn-privatevpn-genconfig
        volumeMounts:
        - mountPath: /etc/openvpn
          name: privatevpn-com
      - command:
        - ovpn_initpki
        - nopass
        image: master:5000/openvpn
        name: vpn-privatevpn-privatevpn-initpki
        volumeMounts:
        - mountPath: /etc/openvpn
          name: privatevpn-com
      - command:
        - easyrsa
        - build-client-full
        - privatevpn
        - nopass
        image: master:5000/openvpn
        name: vpn-privatevpn-privatevpn-build-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: privatevpn-com
      - command:
        - ovpn_getclient
        - privatevpn
        - '>'
        - /configs/vpn/privatevpn.ovpn
        image: master:5000/openvpn
        name: vpn-privatevpn-privatevpn-get-client
        volumeMounts:
        - mountPath: /etc/openvpn
          name: privatevpn-com
      volumes:
      - name: privatevpn-com
        nfs:
          path: /configs/vpn/privatevpn.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vpn-vpn
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vpn-vpn
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: vpn-vpn
    spec:
      containers:
```

```
- env:
  - name: IP_ADDR
    value: 34.202.89.42
  - name: LEN
    value: '8'
  - name: GATEWAY
    value: 34.0.0.1
  - name: INT
    value: net1
  - name: OPEN_VPN_CMD
    value: ovpn_run
  image: master:5000/openvpn
  name: vpn-vpn
  securityContext:
    privileged: true
  volumeMounts:
  - mountPath: /etc/openvpn
    name: vpn-com
initContainers:
- command:
  - ovpn_genconfig
  - -u
  - udp://vpn.com
  image: master:5000/openvpn
  name: vpn-vpn-vpn-genconfig
  volumeMounts:
  - mountPath: /etc/openvpn
    name: vpn-com
- command:
  - ovpn_initpki
  - nopass
  image: master:5000/openvpn
  name: vpn-vpn-vpn-initpki
  volumeMounts:
  - mountPath: /etc/openvpn
    name: vpn-com
- command:
  - easyrsa
  - build-client-full
  - vpn
  - nopass
  image: master:5000/openvpn
  name: vpn-vpn-vpn-build-client
  volumeMounts:
  - mountPath: /etc/openvpn
    name: vpn-com
- command:
  - ovpn_getclient
  - vpn
  - '>'
  - /configs/vpn/vpn.ovpn
  image: master:5000/openvpn
  name: vpn-vpn-vpn-get-client
  volumeMounts:
  - mountPath: /etc/openvpn
    name: vpn-com
volumes:
- name: vpn-com
  nfs:
    path: /configs/vpn/vpn.com
    server: storage
```

Appendix RR

vuln.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-hackattack-com
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-hackattack-com
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: vuln-hackattack-com
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 102.120.3.42
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 102.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit-vuln-svc-emu
        name: vuln-hackattack-com
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-hackme-co-uk
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-hackme-co-uk
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: vuln-hackme-co-uk
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 83.129.43.12
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 83.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit-vuln-svc-emu
```

```
        name: vuln-hackme-co-uk
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-imvulnerable-net
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-imvulnerable-net
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr4-svc
      labels:
        app: vuln-imvulnerable-net
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 103.112.1.3
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 103.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit-vuln-svc-emu
        name: vuln-imvulnerable-net
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-mutillidae
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-mutillidae
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: vuln-mutillidae
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 41.0.12.15
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 41.0.0.1
        - name: INT
          value: net1
        image: master:5000/vuln-mutillidae
        name: vuln-mutillidae
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-org
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-org
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: vuln-org
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 128.143.18.14
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 128.0.0.1
        - name: INT
          value: net1
        image: master:5000/metasploit-vuln-svc-emu
        name: vuln-org
        securityContext:
          privileged: true
        volumeMounts: []
      initContainers: []
      volumes: []
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: vuln-wordpress
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vuln-wordpress
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: vuln-wordpress
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 198.143.164.252
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 198.0.0.1
        - name: INT
          value: net1
        image: master:5000/vuln-wordpress
        name: vuln-wordpress
        securityContext:
```

```
    privileged: true
  volumeMounts: []
initContainers: []
volumes: []
```

Appendix SS

web.yml

```
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-baidu
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-baidu
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: www-baidu
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 123.125.114.144
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 123.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-baidu
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: baidu-com
      initContainers: []
      volumes:
      - name: baidu-com
        nfs:
          path: /configs/web/baidu.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-ebay
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-ebay
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr4-svc
      labels:
        app: www-ebay
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 58.135.195.175
        - name: LEN
```

```
          value: '8'
        - name: GATEWAY
          value: 58.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-ebay
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: ebay-com-au
      initContainers: []
      volumes:
      - name: ebay-com-au
        nfs:
          path: /configs/web/ebay.com.au
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-elgenero
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-elgenero
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr5-svc
      labels:
        app: www-elgenero
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 201.23.113.243
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 201.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-elgenero
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: elgenero-com
      initContainers: []
      volumes:
      - name: elgenero-com
        nfs:
          path: /configs/web/elgenero.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-elsalvador
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-elsalvador
  template:
```

```
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr5-svc
      labels:
        app: www-elsalvador
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 179.249.122.52
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 179.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-elsalvador
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: elsalvador-com
      initContainers: []
      volumes:
      - name: elsalvador-com
        nfs:
          path: /configs/web/elsalvador.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-facebook
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-facebook
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: www-facebook
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 31.13.65.36
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 31.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-facebook
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: facebook-com
      initContainers: []
      volumes:
      - name: facebook-com
        nfs:
          path: /configs/web/facebook.com
          server: storage
---
```

```
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-fakaza
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-fakaza
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: www-fakaza
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 154.162.225.76
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 154.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-fakaza
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: fakaza-com
      initContainers: []
      volumes:
      - name: fakaza-com
        nfs:
          path: /configs/web/fakaza.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-google
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-google
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: www-google
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 172.217.0.78
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 172.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-google
        securityContext:
```

```
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: google-com
      initContainers: []
      volumes:
      - name: google-com
        nfs:
          path: /configs/web/google.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-livedoor
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-livedoor
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: www-livedoor
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 202.104.153.16
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 202.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-livedoor
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: livedoor-com
      initContainers: []
      volumes:
      - name: livedoor-com
        nfs:
          path: /configs/web/livedoor.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-microsoft
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-microsoft
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: www-microsoft
    spec:
      containers:
      - env:
```

```
        - name: IP_ADDR
          value: 40.76.4.15
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 40.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-microsoft
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: microsoft-com
      initContainers: []
      volumes:
      - name: microsoft-com
        nfs:
          path: /configs/web/microsoft.com
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-rtl
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-rtl
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: www-rtl
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 81.92.237.71
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 81.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-rtl
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: rtl-lu
      initContainers: []
      volumes:
      - name: rtl-lu
        nfs:
          path: /configs/web/rtl.lu
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-sabaq
spec:
  replicas: 1
  selector:
```

```
    matchLabels:
      app: www-sabaq
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr3-svc
      labels:
        app: www-sabaq
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 105.16.44.64
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 105.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-sabaq
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: sabq-org
      initContainers: []
      volumes:
      - name: sabq-org
        nfs:
          path: /configs/web/sabq.org
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-standardmedia
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-standardmedia
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr6-svc
      labels:
        app: www-standardmedia
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 105.20.205.11
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 105.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-standardmedia
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: standardmedia-co-ke
      initContainers: []
      volumes:
      - name: standardmedia-co-ke
        nfs:
```

```
          path: /configs/web/standardmedia.co.ke
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-trademe
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-trademe
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr4-svc
      labels:
        app: www-trademe
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 203.162.72.2
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 203.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-trademe
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: trademe-co-nz
      initContainers: []
      volumes:
      - name: trademe-co-nz
        nfs:
          path: /configs/web/trademe.co.nz
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-wikipedia
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-wikipedia
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr1-svc
      labels:
        app: www-wikipedia
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 208.80.154.224
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 208.0.0.1
        - name: INT
          value: net1
```

```
        image: master:5000/nginx
        name: www-wikipedia
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: wikipedia-org
      initContainers: []
      volumes:
      - name: wikipedia-org
        nfs:
          path: /configs/web/wikipedia.org
          server: storage
---
# Source: /home/greyadmin/cbcr/build/range/deployments/template.py
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: www-yandex
spec:
  replicas: 1
  selector:
    matchLabels:
      app: www-yandex
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: rtr2-svc
      labels:
        app: www-yandex
    spec:
      containers:
      - env:
        - name: IP_ADDR
          value: 77.88.55.55
        - name: LEN
          value: '8'
        - name: GATEWAY
          value: 77.0.0.1
        - name: INT
          value: net1
        image: master:5000/nginx
        name: www-yandex
        securityContext:
          privileged: true
        volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: yandex-ru
      initContainers: []
      volumes:
      - name: yandex-ru
        nfs:
          path: /configs/web/yandex.ru
          server: storage
```

Appendix TT

cbcr-kibana-export.json


```
[
  {
    "_id": "50de6720-f62e-11e6-8dfa-1b494d466532",
    "_type": "dashboard",
    "_source": {
      "title": "Dashboard",
      "hits": 0,
      "description": "",
      "panelsJSON":
"[{\"panelIndex\":\"3\",\"gridData\":{\"x\":0,\"y\":8,\"w\":9,\"h\":4,\"i\":\"3\"},\"id\":\"Desti
nation-IP-slash-
Time\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"4\",\"gridData\":{\"x\
":0,\"y\":2,\"w\":6,\"h\":3,\"i\":\"4\"},\"id\":\"Events-slash-
Time\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"7\",\"gridData\":{\"x\
":0,\"y\":0,\"w\":2,\"h\":2,\"i\":\"7\"},\"id\":\"Number\",\"type\":\"visualization\",\"version\"
:\"6.1.0\"},{\"panelIndex\":\"11\",\"gridData\":{\"x\":0,\"y\":12,\"w\":9,\"h\":4,\"i\":\"11\"},\
"id\":\"26d98690-f62d-11e6-8dfa-
1b494d466532\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"12\",\"gridDat
a\":{\"x\":9,\"y\":5,\"w\":3,\"h\":4,\"i\":\"12\"},\"id\":\"17dd5040-f62d-11e6-8dfa-
1b494d466532\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"13\",\"gridDat
a\":{\"x\":9,\"y\":9,\"w\":3,\"h\":4,\"i\":\"13\"},\"id\":\"Destination-
IP\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"19\",\"gridData\":{\"x\"
:0,\"y\":5,\"w\":6,\"h\":3,\"i\":\"19\"},\"id\":\"AWHWTDyouiCz3jvXMHmd\",\"type\":\"visualization
\",\"version\":\"6.1.0\"},{\"panelIndex\":\"20\",\"gridData\":{\"x\":6,\"y\":5,\"w\":3,\"h\":3,\"
i\":\"20\"},\"id\":\"AWHWhw9vuiCz3jvXS1Sb\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"
panelIndex\":\"21\",\"gridData\":{\"x\":2,\"y\":0,\"w\":4,\"h\":2,\"i\":\"21\"},\"id\":\"AWHWRzB1
uiCz3jvXLjB9\",\"type\":\"visualization\",\"version\":\"6.1.0\"},{\"panelIndex\":\"22\",\"gridDat
a\":{\"x\":6,\"y\":0,\"w\":6,\"h\":5,\"i\":\"22\"},\"version\":\"6.1.0\",\"type\":\"visualization
\",\"id\":\"Heat-Map\"}]",
      "optionsJSON": "{\"darkTheme\":true,\"useMargins\":false}",
      "uiStateJSON": "{\"P-1\":{\"vis\":{\"legendOpen\":false}},\"P-
11\":{\"vis\":{\"legendOpen\":true}},\"P-12\":{\"vis\":{\"legendOpen\":false}},\"P-
13\":{\"vis\":{\"legendOpen\":false}},\"P-17\":{\"vis\":{\"legendOpen\":false}},\"P-
20\":{\"vis\":{\"params\":{\"sort\":{\"columnIndex\":null,\"direction\":null}}}},\"P-
21\":{\"vis\":{\"legendOpen\":true}},\"P-3\":{\"vis\":{\"legendOpen\":true}},\"P-
4\":{\"vis\":{\"legendOpen\":false}},\"P-
5\":{\"mapCenter\":[8.754794702435618,0],\"mapZoom\":2},\"P-
6\":{\"vis\":{\"legendOpen\":false}},\"P-7\":{\"vis\":{\"defaultColors\":{\"0 -
100\":\"rgb(0,104,55)\"}}},\"P-8\":{\"vis\":{\"legendOpen\":false}}}",
      "version": 1,
      "timeRestore": false,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON":
"{\"filter\":[],\"query\":{\"language\":\"lucene\",\"query\":\"\"},\"highlightAll\":true,\"versio
n\":true}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "d16064a0-d2ad-11e7-9911-fdaddbe8dcd5",
    "_type": "search",
    "_source": {
      "title": "Search",
      "description": "",
      "hits": 0,
      "columns": [
        "http.hostname",
        "src_ip",
        "dest_port",
        "src_port",
        "alert.action",
```

```
        "event_type",
        "dns.rrname",
        "flow.reason",
        "alert.signature"
      ],
      "sort": [
        "@timestamp",
        "desc"
      ],
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"highlightAll\":true,\"version\":true,\"query\":{\"query\":\"\",\"language\":\"lu
cene\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "e0e27eb0-d2ab-11e7-9911-fdaddbe8dcd5",
    "_type": "search",
    "_source": {
      "title": "Events",
      "description": "",
      "hits": 0,
      "columns": [
        "geoip.country_code2",
        "geoip.region_code",
        "src_ip",
        "src_port",
        "dest_ip",
        "dest_port",
        "http.hostname"
      ],
      "sort": [
        "@timestamp",
        "desc"
      ],
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"highlightAll\":true,\"version\":true,\"query\":{\"language\":\"lucene\",\"query\
":\"\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWefVTuiCz3jvXRbLT",
    "_type": "visualization",
    "_source": {
      "title": "TOP 10 HTTP REQUESTS",
      "visState": "{\"title\":\"TOP 10 HTTP
REQUESTS\",\"type\":\"histogram\",\"params\":{\"grid\":{\"categoryLines\":false,\"style\":{\"colo
r\":\"#eee\"},\"valueAxis\":\"ValueAxis-1\"},\"categoryAxes\":[{\"id\":\"CategoryAxis-
1\",\"type\":\"category\",\"position\":\"bottom\",\"show\":false,\"style\":{},\"scale\":{\"type\"
:\"linear\"},\"labels\":{\"show\":true,\"truncate\":0,\"filter\":false},\"title\":{\"text\":\"\"}
}],\"valueAxes\":[{\"id\":\"ValueAxis-1\",\"name\":\"LeftAxis-
1\",\"type\":\"value\",\"position\":\"left\",\"show\":true,\"style\":{},\"scale\":{\"type\":\"lin
ear\",\"mode\":\"normal\"},\"labels\":{\"show\":true,\"rotate\":0,\"filter\":false,\"truncate\":1
00},\"title\":{\"text\":\"Count\"}}],\"seriesParams\":[{\"show\":\"true\",\"type\":\"histogram\",
\"mode\":\"normal\",\"data\":{\"label\":\"Count\",\"id\":\"1\"},\"valueAxis\":\"ValueAxis-
1\",\"drawLinesBetweenPoints\":true,\"showCircles\":true}],\"addTooltip\":true,\"addLegend\":true
,\"legendPosition\":\"right\",\"times\":[],\"addTimeMarker\":false,\"type\":\"histogram\",\"order
BucketsBySum\":false},\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"count\",\"schema\":\"me
tric\",\"params\":{}},{\"id\":\"2\",\"enabled\":true,\"type\":\"terms\",\"schema\":\"group\",\"pa
rams\":{\"field\":\"referrer.keyword\",\"exclude\":\"-
```

```
\",\"size\":10,\"order\":\"desc\",\"orderBy\":\"1\",\"customLabel\":\"HTTP
Host\"}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"negate\":false,\"disabled\":false,\"alias\":null,\"typ
e\":\"phrase\",\"key\":\"type.keyword\",\"value\":\"bro-
http_log\"},\"query\":{\"match\":{\"type.keyword\":{\"query\":\"bro-
http_log\",\"type\":\"phrase\"}}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWhw9vuiCz3jvXS1Sb",
    "_type": "visualization",
    "_source": {
      "title": "TOP 10 REMOTE PORTS",
      "visState": "{\"title\":\"TOP 10 REMOTE
PORTS\",\"type\":\"table\",\"params\":{\"perPage\":10,\"showPartialRows\":false,\"showMeticsAtAll
Levels\":false,\"sort\":{\"columnIndex\":null,\"direction\":null},\"showTotal\":false,\"totalFunc
\":\"sum\",\"type\":\"table\"},\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"count\",\"sche
ma\":\"metric\",\"params\":{}},{\"id\":\"2\",\"enabled\":true,\"type\":\"terms\",\"schema\":\"buc
ket\",\"params\":{\"field\":\"id_resp_port.keyword\",\"exclude\":\"0\",\"size\":10,\"order\":\"de
sc\",\"orderBy\":\"1\",\"customLabel\":\"Remote Port\"}}],\"listeners\":{}}",
      "uiStateJSON":
"{\"vis\":{\"params\":{\"sort\":{\"columnIndex\":null,\"direction\":null}}}}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"negate\":false,\"disabled\":false,\"alias\":null,\"typ
e\":\"phrase\",\"key\":\"type.keyword\",\"value\":\"bro-
conn_log\"},\"query\":{\"match\":{\"type.keyword\":{\"query\":\"bro-
conn_log\",\"type\":\"phrase\"}}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWRzB1uiCz3jvXLjB9",
    "_type": "visualization",
    "_source": {
      "title": "TOP PROTOCOLS",
      "visState": "{\"title\":\"TOP
PROTOCOLS\",\"type\":\"pie\",\"params\":{\"addTooltip\":true,\"addLegend\":true,\"legendPosition\
":\"right\",\"isDonut\":false,\"type\":\"pie\"},\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\"
:\"count\",\"schema\":\"metric\",\"params\":{}},{\"id\":\"2\",\"enabled\":true,\"type\":\"terms\"
,\"schema\":\"segment\",\"params\":{\"field\":\"proto.keyword\",\"size\":5,\"order\":\"desc\",\"o
rderBy\":\"1\",\"customLabel\":\"Protocol\"}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"negate\":false,\"disabled\":false,\"alias\":null,\"typ
e\":\"phrase\",\"key\":\"type.keyword\",\"value\":\"bro-
conn_log\"},\"query\":{\"match\":{\"type.keyword\":{\"query\":\"bro-
conn_log\",\"type\":\"phrase\"}}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
```

```
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWq6wBuiCz3jvXXqYo",
    "_type": "visualization",
    "_source": {
      "title": "BRO LOGS",
      "visState": "{\"title\":\"BRO
LOGS\",\"type\":\"table\",\"params\":{\"perPage\":3,\"showMeticsAtAllLevels\":false,\"showPartial
Rows\":false,\"showTotal\":true,\"sort\":{\"columnIndex\":null,\"direction\":null},\"totalFunc\":
\"sum\",\"type\":\"table\"},\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"count\",\"schema\
":\"metric\",\"params\":{\"customLabel\":\"# OF
RECORDS\"}},{\"id\":\"2\",\"enabled\":true,\"type\":\"terms\",\"schema\":\"bucket\",\"params\":{\
"field\":\"type.keyword\",\"include\":\"\",\"size\":5,\"order\":\"desc\",\"orderBy\":\"1\",\"cust
omLabel\":\"LOG FILES\"}}],\"listeners\":{}}",
      "uiStateJSON":
"{\"vis\":{\"params\":{\"sort\":{\"columnIndex\":null,\"direction\":null}}}}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"type\":\"phrases\",\"key\":\"type\",\"value\":\"bro-
conn_log, bro-http_log, bro-intel_log\",\"params\":[\"bro-conn_log\",\"bro-http_log\",\"bro-
intel_log\"],\"negate\":false,\"disabled\":false,\"alias\":null},\"query\":{\"bool\":{\"should\":
[{\"match_phrase\":{\"type\":\"bro-conn_log\"}},{\"match_phrase\":{\"type\":\"bro-
http_log\"}},{\"match_phrase\":{\"type\":\"bro-
intel_log\"}}],\"minimum_should_match\":1}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWRA0BuiCz3jvXLG2E",
    "_type": "visualization",
    "_source": {
      "title": "CONNECTIONS COUNT PER MINUTE",
      "visState": "{\"title\":\"CONNECTIONS COUNT PER
MINUTE\",\"type\":\"line\",\"params\":{\"addLegend\":true,\"addTimeMarker\":false,\"addTooltip\":
true,\"categoryAxes\":[{\"id\":\"CategoryAxis-
1\",\"labels\":{\"show\":true,\"truncate\":0},\"position\":\"bottom\",\"scale\":{\"type\":\"linea
r\"},\"show\":true,\"style\":{},\"title\":{\"text\":\"@timestamp per
minute\"},\"type\":\"category\"}],\"grid\":{\"categoryLines\":true,\"style\":{\"color\":\"#eee\"}
,\"valueAxis\":\"ValueAxis-
1\"},\"legendPosition\":\"top\",\"seriesParams\":[{\"show\":true,\"mode\":\"normal\",\"type\":\"l
ine\",\"drawLinesBetweenPoints\":true,\"showCircles\":true,\"data\":{\"id\":\"3\",\"label\":\"Cou
nt\"},\"valueAxis\":\"ValueAxis-
1\"}],\"times\":[],\"type\":\"line\",\"valueAxes\":[{\"id\":\"ValueAxis-
1\",\"labels\":{\"filter\":false,\"rotate\":0,\"show\":true,\"truncate\":100},\"name\":\"LeftAxis
-
1\",\"position\":\"left\",\"scale\":{\"mode\":\"normal\",\"type\":\"linear\"},\"show\":true,\"sty
le\":{},\"title\":{\"text\":\"Count\"},\"type\":\"value\"}]},\"aggs\":[{\"id\":\"2\",\"enabled\":
true,\"type\":\"date_histogram\",\"schema\":\"segment\",\"params\":{\"field\":\"@timestamp\",\"in
terval\":\"m\",\"customInterval\":\"2h\",\"min_doc_count\":1,\"extended_bounds\":{}}},{\"id\":\"3
\",\"enabled\":true,\"type\":\"count\",\"schema\":\"metric\",\"params\":{}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"negate\":false,\"disabled\":false,\"alias\":null,\"typ
e\":\"phrase\",\"key\":\"type.keyword\",\"value\":\"bro-
conn_log\"},\"query\":{\"match\":{\"type.keyword\":{\"query\":\"bro-
conn_log\",\"type\":\"phrase\"}}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
```

```
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "AWHWTDyouiCz3jvXMHmd",
    "_type": "visualization",
    "_source": {
      "title": "TOP 10 TALKERS",
      "visState": "{\"title\":\"TOP 10
TALKERS\",\"type\":\"histogram\",\"params\":{\"grid\":{\"categoryLines\":false,\"style\":{\"color
\":\"#eee\"},\"valueAxis\":\"ValueAxis-1\"},\"categoryAxes\":[{\"id\":\"CategoryAxis-
1\",\"type\":\"category\",\"position\":\"bottom\",\"show\":false,\"style\":{},\"scale\":{\"type\"
:\"linear\"},\"labels\":{\"show\":true,\"truncate\":0},\"title\":{\"text\":\"\"}}],\"valueAxes\":
[{\"id\":\"ValueAxis-1\",\"name\":\"LeftAxis-
1\",\"type\":\"value\",\"position\":\"left\",\"show\":true,\"style\":{},\"scale\":{\"type\":\"lin
ear\",\"mode\":\"normal\"},\"labels\":{\"show\":true,\"rotate\":0,\"filter\":false,\"truncate\":1
00},\"title\":{\"text\":\"Count\"}}],\"seriesParams\":[{\"show\":\"true\",\"type\":\"histogram\",
\"mode\":\"normal\",\"data\":{\"label\":\"Count\",\"id\":\"1\"},\"valueAxis\":\"ValueAxis-
1\",\"drawLinesBetweenPoints\":true,\"showCircles\":true}],\"addTooltip\":true,\"addLegend\":true
,\"legendPosition\":\"right\",\"times\":[],\"addTimeMarker\":false,\"type\":\"histogram\",\"order
BucketsBySum\":false},\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"count\",\"schema\":\"me
tric\",\"params\":{}},{\"id\":\"2\",\"enabled\":true,\"type\":\"terms\",\"schema\":\"group\",\"pa
rams\":{\"field\":\"id_orig_host.keyword\",\"size\":10,\"order\":\"desc\",\"orderBy\":\"1\",\"cus
tomLabel\":\"Host\"}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"match_all\":{}},\"language\":\"lucene\"},\"filter\":[{\"me
ta\":{\"index\":\"AWHWHTYfuiCz3jvXGox0\",\"negate\":false,\"disabled\":false,\"alias\":null,\"typ
e\":\"phrase\",\"key\":\"type.keyword\",\"value\":\"bro-
conn_log\"},\"query\":{\"match\":{\"type.keyword\":{\"query\":\"bro-
conn_log\",\"type\":\"phrase\"}}},\"$state\":{\"store\":\"appState\"}}]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Number",
    "_type": "visualization",
    "_source": {
      "title": "Number",
      "visState":
"{\"title\":\"Number\",\"type\":\"metric\",\"params\":{\"handleNoResults\":true,\"fontSize\":60},
\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"cardinality\",\"schema\":\"metric\",\"params\
":{\"field\":\"_id\"}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"query_string\":{\"query\":\"*\",\"analyze_wildcard\":true}
},\"language\":\"lucene\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Protocal",
    "_type": "visualization",
    "_source": {
      "title": "Protocal",
      "visState":
"{\"title\":\"Protocal\",\"type\":\"pie\",\"params\":{\"shareYAxis\":true,\"addTooltip\":true,\"a
ddLegend\":true,\"legendPosition\":\"right\",\"isDonut\":true},\"aggs\":[{\"id\":\"1\",\"enabled\
":true,\"type\":\"count\",\"schema\":\"metric\",\"params\":{}},{\"id\":\"2\",\"enabled\":true,\"t
```

```
ype\":\"terms\",\"schema\":\"segment\",\"params\":{\"field\":\"proto.keyword\",\"size\":5,\"order
\":\"desc\",\"orderBy\":\"1\"}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"query_string\":{\"query\":\"*\",\"analyze_wildcard\":true}
},\"language\":\"lucene\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Events-slash-Time",
    "_type": "visualization",
    "_source": {
      "title": "Events/Time",
      "visState":
"{\"title\":\"Events/Time\",\"type\":\"histogram\",\"params\":{\"shareYAxis\":true,\"addTooltip\"
:true,\"addLegend\":true,\"legendPosition\":\"right\",\"scale\":\"linear\",\"mode\":\"stacked\",\
"times\":[],\"addTimeMarker\":false,\"defaultYExtents\":false,\"setYExtents\":false,\"yAxis\":{}}
,\"aggs\":[{\"id\":\"1\",\"enabled\":true,\"type\":\"count\",\"schema\":\"metric\",\"params\":{}}
,{\"id\":\"2\",\"enabled\":true,\"type\":\"date_histogram\",\"schema\":\"segment\",\"params\":{\"
field\":\"@timestamp\",\"interval\":\"auto\",\"customInterval\":\"2h\",\"min_doc_count\":1,\"exte
nded_bounds\":{}}}],\"listeners\":{}}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"query_string\":{\"query\":\"*\",\"analyze_wildcard\":true}
},\"language\":\"lucene\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Interfaces",
    "_type": "visualization",
    "_source": {
      "title": "Interfaces",
      "visState":
"{\"title\":\"Interfaces\",\"type\":\"pie\",\"params\":{\"shareYAxis\":true,\"addTooltip\":true,\
"addLegend\":true,\"legendPosition\":\"right\",\"isDonut\":true,\"type\":\"pie\"},\"aggs\":[{\"id
\":\"1\",\"enabled\":true,\"type\":\"count\",\"schema\":\"metric\",\"params\":{}},{\"id\":\"2\",\
"enabled\":true,\"type\":\"terms\",\"schema\":\"segment\",\"params\":{\"field\":\"in_iface.keywor
d\",\"size\":5,\"order\":\"desc\",\"orderBy\":\"1\"}}]}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\"index\":\"a6b67290-9475-11e9-a460-
d1a5dde94362\",\"query\":{\"query\":{\"query_string\":{\"query\":\"*\",\"analyze_wildcard\":true,
\"default_field\":\"*\"}},\"language\":\"lucene\"},\"filter\":[]}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Heat-Map",
    "_type": "visualization",
    "_source": {
      "title": "Heat Map",
```

```
      "visState": "{\n  \"title\": \"Heat Map\",\n  \"type\": \"tile_map\",\n  \"params\": {\n
\"mapType\": \"Heatmap\",\n    \"isDesaturated\": true,\n    \"addTooltip\": true,\n
\"heatMaxZoom\": 16,\n    \"heatMinOpacity\": 0.1,\n    \"heatRadius\": 25,\n    \"heatBlur\":
15,\n    \"heatNormalizeData\": true,\n    \"mapZoom\": 2,\n    \"mapCenter\": [\n      15,\n
5\n    ],\n    \"wms\": {\n      \"enabled\": false,\n      \"url\":
\"https://basemap.nationalmap.gov/arcgis/services/USGSTopo/MapServer/WMSServer\",\n
\"options\": {\n        \"version\": \"1.3.0\",\n        \"layers\": \"0\",\n        \"format\":
\"image/png\",\n        \"transparent\": true,\n        \"attribution\": \"Maps provided by
USGS\",\n        \"styles\": \"\"\n      }\n    },\n    \"heatClusterSize\": 2,\n
\"legendPosition\": \"bottomright\"\n  },\n  \"aggs\": [\n    {\n      \"id\": \"1\",\n
\"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n      \"params\":
{}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"geohash_grid\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"geoip.location\",\n        \"autoPrecision\": true,\n        \"isFilteredByCollar\": true,\n
\"useGeocentroid\": true,\n        \"precision\": 2\n      }\n    }\n  ]\n}",
      "uiStateJSON": "{\n  \"mapCenter\": [\n    15.114552871944115,\n    5.09765625\n  ]\n}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true,\n        \"default_field\": \"*\"\n      }\n    },\n    \"language\":
\"lucene\"\n  },\n  \"filter\": []\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "26d98690-f62d-11e6-8dfa-1b494d466532",
    "_type": "visualization",
    "_source": {
      "title": "Source IP/Time",
      "visState": "{\n  \"title\": \"Source IP/Time\",\n  \"type\": \"histogram\",\n  \"params\":
{\n    \"shareYAxis\": true,\n    \"addTooltip\": true,\n    \"addLegend\": true,\n
\"legendPosition\": \"right\",\n    \"scale\": \"linear\",\n    \"mode\": \"stacked\",\n
\"times\": [],\n    \"addTimeMarker\": false,\n    \"defaultYExtents\": false,\n
\"setYExtents\": false,\n    \"yAxis\": {}\n  },\n  \"aggs\": [\n    {\n      \"id\": \"1\",\n
\"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n      \"params\":
{}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"date_histogram\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"@timestamp\",\n        \"interval\": \"auto\",\n        \"customInterval\": \"2h\",\n
\"min_doc_count\": 1,\n        \"extended_bounds\": {}\n      }\n    },\n    {\n      \"id\":
\"3\",\n      \"enabled\": true,\n      \"type\": \"terms\",\n      \"schema\": \"group\",\n
\"params\": {\n        \"field\": \"id_orig_host.keyword\",\n        \"size\": 5,\n
\"order\": \"desc\",\n        \"orderBy\": \"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Destination-IP-slash-Time",
    "_type": "visualization",
    "_source": {
      "title": "Destination IP/Time",
      "visState": "{\n  \"title\": \"Destination IP/Time\",\n  \"type\": \"histogram\",\n
\"params\": {\n    \"shareYAxis\": true,\n    \"addTooltip\": true,\n    \"addLegend\": true,\n
\"legendPosition\": \"right\",\n    \"scale\": \"linear\",\n    \"mode\": \"stacked\",\n
\"times\": [],\n    \"addTimeMarker\": false,\n    \"defaultYExtents\": false,\n
\"setYExtents\": false,\n    \"yAxis\": {}\n  },\n  \"aggs\": [\n    {\n      \"id\": \"1\",\n
```


Bryan Scarbrough, bryan.scarbrough@gmail.com

```
\"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n      \"params\":
{}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"date_histogram\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"@timestamp\",\n        \"interval\": \"auto\",\n        \"customInterval\": \"2h\",\n
\"min_doc_count\": 1,\n        \"extended_bounds\": {}\n      }\n    },\n    {\n      \"id\":
\"3\",\n      \"enabled\": true,\n      \"type\": \"terms\",\n      \"schema\": \"group\",\n
\"params\": {\n        \"field\": \"id_resp_host.keyword\",\n        \"size\": 5,\n
\"order\": \"desc\",\n        \"orderBy\": \"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "534412b0-f6c1-11e6-8c30-e9ddbd718208",
    "_type": "visualization",
    "_source": {
      "title": "Country/Time",
      "visState": "{\n  \"title\": \"Country/Time\",\n  \"type\": \"histogram\",\n  \"params\":
{\n    \"addTooltip\": true,\n    \"addLegend\": true,\n    \"legendPosition\": \"right\",\n
\"scale\": \"linear\",\n    \"mode\": \"stacked\",\n    \"times\": [],\n    \"addTimeMarker\":
false,\n    \"defaultYExtents\": false,\n    \"setYExtents\": false\n  },\n  \"aggs\": [\n    {\n
\"id\": \"1\",\n      \"enabled\": true,\n      \"type\": \"count\",\n      \"schema\":
\"metric\",\n      \"params\": {}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\":
true,\n      \"type\": \"date_histogram\",\n      \"schema\": \"segment\",\n      \"params\": {\n
\"field\": \"@timestamp\",\n        \"interval\": \"auto\",\n        \"customInterval\":
\"2h\",\n        \"min_doc_count\": 1,\n        \"extended_bounds\": {}\n      }\n    },\n    {\n
\"id\": \"3\",\n      \"enabled\": true,\n      \"type\": \"terms\",\n      \"schema\":
\"group\",\n      \"params\": {\n        \"field\": \"geoip.location\",\n        \"size\": 5,\n
\"order\": \"desc\",\n        \"orderBy\": \"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Top-Countries",
    "_type": "visualization",
    "_source": {
      "title": "Top Countries",
      "visState": "{\n  \"title\": \"Top Countries\",\n  \"type\": \"table\",\n  \"params\": {\n
\"perPage\": 10,\n    \"showPartialRows\": false,\n    \"showMeticsAtAllLevels\": false,\n
\"sort\": {\n      \"columnIndex\": null,\n      \"direction\": null\n    },\n    \"showTotal\":
false,\n    \"totalFunc\": \"sum\"\n  },\n  \"aggs\": [\n    {\n      \"id\": \"1\",\n
\"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n      \"params\":
{}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\": \"terms\",\n
\"schema\": \"bucket\",\n      \"params\": {\n        \"field\": \"geoip.location\",\n
\"size\": 10,\n        \"order\": \"desc\",\n        \"orderBy\": \"1\"\n      }\n    }\n  ],\n
\"listeners\": {}\n}",
      "uiStateJSON": "{\n  \"vis\": {\n    \"params\": {\n      \"sort\": {\n
\"columnIndex\": null,\n        \"direction\": null\n      }\n    }\n  }\n}",
      "description": "",
      "version": 1,
```

```
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "17dd5040-f62d-11e6-8dfa-1b494d466532",
    "_type": "visualization",
    "_source": {
      "title": "Destination IP",
      "visState": "{\n  \"title\": \"Destination IP\",\n  \"type\": \"pie\",\n  \"params\": {\n
\"shareYAxis\": true,\n    \"addTooltip\": true,\n    \"addLegend\": true,\n
\"legendPosition\": \"right\",\n    \"isDonut\": true\n  },\n  \"aggs\": [\n    {\n      \"id\":
\"1\",\n      \"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n
\"params\": {}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"terms\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"id_resp_host.keyword\",\n        \"size\": 25,\n        \"order\": \"desc\",\n
\"orderBy\": \"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Destination-IP",
    "_type": "visualization",
    "_source": {
      "title": "Source IP",
      "visState": "{\n  \"title\": \"Source IP\",\n  \"type\": \"pie\",\n  \"params\": {\n
\"shareYAxis\": true,\n    \"addTooltip\": true,\n    \"addLegend\": true,\n
\"legendPosition\": \"right\",\n    \"isDonut\": true\n  },\n  \"aggs\": [\n    {\n      \"id\":
\"1\",\n      \"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n
\"params\": {}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"terms\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"id_orig_host.keyword\",\n        \"size\": 25,\n        \"order\": \"desc\",\n
\"orderBy\": \"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  },
  {
    "_id": "Countries",
    "_type": "visualization",
    "_source": {
      "title": "Countries",
```

```
      "visState": "{\n  \"title\": \"Countries\",\n  \"type\": \"pie\",\n  \"params\": {\n
\"shareYAxis\": true,\n    \"addTooltip\": true,\n    \"addLegend\": true,\n
\"legendPosition\": \"right\",\n    \"isDonut\": true\n  },\n  \"aggs\": [\n    {\n      \"id\":
\"1\",\n      \"enabled\": true,\n      \"type\": \"count\",\n      \"schema\": \"metric\",\n
\"params\": {}\n    },\n    {\n      \"id\": \"2\",\n      \"enabled\": true,\n      \"type\":
\"terms\",\n      \"schema\": \"segment\",\n      \"params\": {\n        \"field\":
\"geoip.location\",\n        \"size\": 25,\n        \"order\": \"desc\",\n        \"orderBy\":
\"1\"\n      }\n    }\n  ],\n  \"listeners\": {}\n}",
      "uiStateJSON": "{}",
      "description": "",
      "version": 1,
      "kibanaSavedObjectMeta": {
        "searchSourceJSON": "{\n  \"index\": \"a6b67290-9475-11e9-a460-d1a5dde94362\",\n
\"query\": {\n    \"query\": {\n      \"query_string\": {\n        \"query\": \"*\",\n
\"analyze_wildcard\": true\n      }\n    },\n    \"language\": \"lucene\"\n  },\n  \"filter\":
[]\n}"
      }
    },
    "_meta": {
      "savedObjectVersion": 2
    }
  }
]
```

Appendix UU

local.bro

```
##! Local site policy. Customize as appropriate.
##!
##! This file will not be overwritten when upgrading or reinstalling!

# This script logs which scripts were loaded during each run.
@load misc/loaded-scripts

# Apply the default tuning scripts for common tuning settings.
@load tuning/defaults

# Estimate and log capture loss.
@load misc/capture-loss

# Enable logging of memory, packet and lag statistics.
@load misc/stats

# Load the scan detection script.
@load misc/scan

# Detect traceroute being run on the network. This could possibly cause
# performance trouble when there are a lot of traceroutes on your network.
# Enable cautiously.
@load misc/detect-traceroute

# Generate notices when vulnerable versions of software are discovered.
# The default is to only monitor software found in the address space defined
# as "local".  Refer to the software framework's documentation for more
# information.
@load frameworks/software/vulnerable

# Detect software changing (e.g. attacker installing hacked SSHD).
@load frameworks/software/version-changes

# This adds signatures to detect cleartext forward and reverse windows shells.
@load-sigs frameworks/signatures/detect-windows-shells

# Load all of the scripts that detect software in various protocols.
@load protocols/ftp/software
@load protocols/smtp/software
@load protocols/ssh/software
@load protocols/http/software
# The detect-webapps script could possibly cause performance trouble when
# running on live traffic.  Enable it cautiously.
#@load protocols/http/detect-webapps

# This script detects DNS results pointing toward your Site::local_nets
# where the name is not part of your local DNS zone and is being hosted
# externally.  Requires that the Site::local_zones variable is defined.
@load protocols/dns/detect-external-names

# Script to detect various activity in FTP sessions.
@load protocols/ftp/detect

# Scripts that do asset tracking.
@load protocols/conn/known-hosts
@load protocols/conn/known-services
@load protocols/ssl/known-certs

# This script enables SSL/TLS certificate validation.
@load protocols/ssl/validate-certs

# This script prevents the logging of SSL CA certificates in x509.log
@load protocols/ssl/log-hostcerts-only
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
# Uncomment the following line to check each SSL certificate hash against the ICSI
# certificate notary service; see http://notary.icsi.berkeley.edu .
# @load protocols/ssl/notary

# If you have libGeoIP support built in, do some geographic detections and
# logging for SSH traffic.
@load protocols/ssh/geo-data
# Detect hosts doing SSH bruteforce attacks.
@load protocols/ssh/detect-bruteforcing
# Detect logins using "interesting" hostnames.
@load protocols/ssh/interesting-hostnames

# Detect SQL injection attacks.
@load protocols/http/detect-sqli

#### Network File Handling ####

# Enable MD5 and SHA1 hashing for all files.
@load frameworks/files/hash-all-files

# Detect SHA1 sums in Team Cymru's Malware Hash Registry.
@load frameworks/files/detect-MHR

# Uncomment the following line to enable detection of the heartbleed attack. Enabling
# this might impact performance a bit.
# @load policy/protocols/ssl/heartbleed

# Uncomment the following line to enable logging of connection VLANs. Enabling
# this adds two VLAN fields to the conn.log file.
# @load policy/protocols/conn/vlan-logging

# Uncomment the following line to enable logging of link-layer addresses. Enabling
# this adds the link-layer address for each connection endpoint to the conn.log file.
@load policy/protocols/conn/mac-logging

# Uncomment the following line to enable the SMB analyzer.  The analyzer
# is currently considered a preview and therefore not loaded by default.
@load policy/protocols/smb
```

Appendix VV

elasticsearch.yml

```
---
## Default Elasticsearch configuration from elasticsearch-docker.
## from https://github.com/elastic/elasticsearch-
docker/blob/master/build/elasticsearch/elasticsearch.yml
#
cluster.name: "docker-cluster"
network.host: 0.0.0.0

# minimum_master_nodes need to be explicitly set when bound on a public IP
# set to 1 to allow single node clusters
# Details: https://github.com/elastic/elasticsearch/pull/17288
discovery.zen.minimum_master_nodes: 1

## Use single node discovery in order to disable production mode and avoid bootstrap checks
## see https://www.elastic.co/guide/en/elasticsearch/reference/current/bootstrap-checks.html
#
discovery.type: single-node
```

Appendix WW

kibana.yml

```
---
## Default Kibana configuration from kibana-docker.
## from https://github.com/elastic/kibana-docker/blob/master/build/kibana/config/kibana.yml
#
server.name: kibana
server.host: "0"
elasticsearch.url: http://elasticsearch:9200
```

Appendix XX

logstash.yml

```
---
## Default Logstash configuration from logstash-docker.
## from https://github.com/elastic/logstash-docker/blob/master/build/logstash/config/logstash-
oss.yml
#
http.host: "0.0.0.0"
path.config: /usr/share/logstash/pipeline
```

Appendix YY

logstash.conf

```
########################
# logstash Configuration Files - Bro IDS Logs
#
# For use with logstash, elasticsearch, and kibana to analyze logs
#
# Usage: Reference this config file for your instance of logstash to parse Bro conn logs
#
# Limitations: Standard Bro log delimiter is tab.
#
# Dependencies: Utilizing the logstash 'translate' filter requires having the logstash contrib
plugins added, which are community supported and not part of the official release. Visit
logstash.net to find out how to install these
#
########################

input {
  file {
    type => "bro-conn_log"
    start_position => "end"
    sincedb_path => "/var/tmp/.bro_conn_sincedb"

    #Edit the following path to reflect the location of your log files. You can also change the
extension if you use something else
    path => "/usr/share/logstash/bro/conn.log"
  }
}

filter {

  #Let's get rid of those header lines; they begin with a hash
  if [message] =~ /^#/ {
    drop { }
  }

  #Now, using the csv filter, we can define the Bro log fields
  if [type] == "bro-conn_log" {
    csv {
      columns =>
["ts","uid","id.orig_h","id.orig_p","id.resp_h","id.resp_p","proto","service","duration","orig_by
tes","resp_bytes","conn_state","local_orig","missed_bytes","history","orig_pkts","orig_ip_bytes",
"resp_pkts","resp_ip_bytes","tunnel_parents"]

      #If you use a custom delimiter, change the following value in between the quotes to your
delimiter. Otherwise, insert a literal <tab> in between the two quotes on your logstash system,
use a text editor like nano that doesn't convert tabs to spaces.
      separator => "	"
    }

    #Let's convert our timestamp into the 'ts' field, so we can use Kibana features natively
    date {
      match => [ "ts", "UNIX" ]
    }

    # add geoip attributes
    geoip {
      source => "id.orig_h"
      target => "orig_geoip"
    }
    geoip {
      source => "id.resp_h"
      target => "resp_geoip"
    }
```

```
    #The following makes use of the translate filter (logstash contrib) to convert conn_state
into human text. Saves having to look up values for packet introspection
    translate {
      field => "conn_state"

      destination => "conn_state_full"

      dictionary => [
                    "S0", "Connection attempt seen, no reply",
                    "S1", "Connection established, not terminated",
                    "S2", "Connection established and close attempt by originator seen (but no
reply from responder)",
                    "S3", "Connection established and close attempt by responder seen (but no
reply from originator)",
                    "SF", "Normal SYN/FIN completion",
                    "REJ", "Connection attempt rejected",
                    "RSTO", "Connection established, originator aborted (sent a RST)",
                    "RSTR", "Established, responder aborted",
                    "RSTOS0", "Originator sent a SYN followed by a RST, we never saw a SYN-ACK
from the responder",
                    "RSTRH", "Responder sent a SYN ACK followed by a RST, we never saw a SYN from
the (purported) originator",
                    "SH", "Originator sent a SYN followed by a FIN, we never saw a SYN ACK from
the responder (hence the connection was 'half' open)",
                                        "SHR", "Responder sent a SYN ACK followed by a FIN, we never
saw a SYN from the originator",
                    "OTH", "No SYN seen, just midstream traffic (a 'partial connection' that was
not later closed)"
                    ]
    }

    mutate {
      convert => [ "id.orig_p", "integer" ]
      convert => [ "id.resp_p", "integer" ]
      convert => [ "orig_bytes", "integer" ]
      convert => [ "duration", "float" ]
      convert => [ "resp_bytes", "integer" ]
      convert => [ "missed_bytes", "integer" ]
      convert => [ "orig_pkts", "integer" ]
      convert => [ "orig_ip_bytes", "integer" ]
      convert => [ "resp_pkts", "integer" ]
      convert => [ "resp_ip_bytes", "integer" ]
      rename =>  [ "id.orig_h", "id_orig_host" ]
      rename =>  [ "id.orig_p", "id_orig_port" ]
      rename =>  [ "id.resp_h", "id_resp_host" ]
      rename =>  [ "id.resp_p", "id_resp_port" ]
    }
  }
}

output {
  # stdout { codec => rubydebug }
  elasticsearch { hosts => "elasticsearch:9200" }
}
```

Appendix ZZ

daemons

```
# This file tells the frr package which daemons to start.
#
# Entries are in the format: <daemon>=(yes|no|priority)
#   0, "no"  = disabled
#   1, "yes" = highest priority
#   2 .. 10  = lower priorities
#
# For daemons which support multiple instances, a 2nd line listing
# the instances can be added. Eg for ospfd:
#   ospfd=yes
#   ospfd_instances="1,2"
#
# Priorities were suggested by Dancer <dancer@zeor.simegen.com>.
# They're used to start the FRR daemons in more than one step
# (for example start one or two at network initialization and the
# rest later). The number of FRR daemons being small, priorities
# must be between 1 and 9, inclusive (or the initscript has to be
# changed). /etc/init.d/frr then can be started as
#
#   /etc/init.d/frr <start|stop|restart|<priority>>
#
# where priority 0 is the same as 'stop', priority 10 or 'start'
# means 'start all'
#
# Sample configurations for these daemons can be found in
# /usr/share/doc/frr/examples/.
#
# ATTENTION:
#
# When activation a daemon at the first time, a config file, even if it is
# empty, has to be present *and* be owned by the user and group "frr", else
# the daemon will not be started by /etc/init.d/frr. The permissions should
# be u=rw,g=r,o=.
# When using "vtysh" such a config file is also needed. It should be owned by
# group "frrvty" and set to ug=rw,o= though. Check /etc/pam.d/frr, too.
#
watchfrr_enable=yes
watchfrr_options="-r '/usr/lib/frr/frr restart %s' -s '/usr/lib/frr/frr start %s' -k
'/usr/lib/frr/frr stop %s'"
#
zebra=yes
bgpd=yes
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isisd=no
ldpd=no
pimd=no
nhrpd=no
eigrpd=no
babeld=no
sharpd=no
pbrd=no
staticd=no
bfdd=no
fabricd=no
#
# Command line options for the daemons
#
zebra_options=("-A 127.0.0.1")
bgpd_options=("-A 127.0.0.1")
ospfd_options=("-A 127.0.0.1")
ospf6d_options=("-A ::1")
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
ripd_options=("-A 127.0.0.1")
ripngd_options=("-A ::1")
isisd_options=("-A 127.0.0.1")
ldpd_options=("-A 127.0.0.1")
pimd_options=("-A 127.0.0.1")
nhrpd_options=("-A 127.0.0.1")
eigrpd_options=("-A 127.0.0.1")
babeld_options=("-A 127.0.0.1")
sharpd_options=("-A 127.0.0.1")
pbrd_options=("-A 127.0.0.1")
staticd_options=("-A 127.0.0.1")
bfdd_options=("-A 127.0.0.1")
fabricd_options=("-A 127.0.0.1")

#
# If the vtysh_enable is yes, then the unified config is read
# and applied if it exists.  If no unified frr.conf exists
# then the per-daemon <daemon>.conf files are used)
# If vtysh_enable is no or non-existant, the frr.conf is ignored.
# it is highly suggested to have this set to yes
vtysh_enable=yes
```

Appendix AAA

daemons.conf

```
#
# If this option is set the /etc/init.d/frr script automatically loads
# the config via "vtysh -b" when the servers are started.
# Check /etc/pam.d/frr if you intend to use "vtysh"!
#
vtysh_enable=yes
zebra_options="  -s 90000000 --daemon -A 127.0.0.1"
bgpd_options="   --daemon -A 127.0.0.1"
ospfd_options="  --daemon -A 127.0.0.1"
ospf6d_options=" --daemon -A ::1"
ripd_options="   --daemon -A 127.0.0.1"
ripngd_options=" --daemon -A ::1"
isisd_options="  --daemon -A 127.0.0.1"
pimd_options="  --daemon -A 127.0.0.1"
ldpd_options="  --daemon -A 127.0.0.1"
nhrpd_options="  --daemon -A 127.0.0.1"
eigrpd_options="  --daemon -A 127.0.0.1"
babeld_options="  --daemon -A 127.0.0.1"
sharpd_options="  --daemon -A 127.0.0.1"
pbrd_options="  --daemon -A 127.0.0.1"
staticd_options="  --daemon -A 127.0.0.1"
bfdd_options="  --daemon -A 127.0.0.1"
fabricd_options="  --daemon -A 127.0.0.1"

# The list of daemons to watch is automatically generated by the init script.
watchfrr_enable=yes
watchfrr_options=(-d -r /usr/sbin/servicebBfrrbBrestartbB%s -s /usr/sbin/servicebBfrrbBstartbB%s
-k /usr/sbin/servicebBfrrbBstopbB%s -b bB)

# If valgrind_enable is 'yes' the frr daemons will be started via valgrind.
# The use case for doing so is tracking down memory leaks, etc in frr.
valgrind_enable=no
valgrind=/usr/bin/valgrind
```

Appendix BBB

Router 01 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:47:43
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-us
!
!
router bgp 6939
 neighbor 4.69.184.2 remote-as 52871
 neighbor 114.31.199.1 remote-as 4826
 neighbor 206.126.236.1 remote-as 3257
 !
 address-family ipv4 unicast
  network 1.0.0.0/8
  network 3.0.0.0/8
  network 4.0.0.0/8
  network 6.0.0.0/8
  network 7.0.0.0/8
  network 8.0.0.0/8
  network 9.0.0.0/8
  network 11.0.0.0/8
  network 12.0.0.0/8
  network 13.0.0.0/8
  network 15.0.0.0/8
  network 16.0.0.0/8
  network 17.0.0.0/8
  network 18.0.0.0/8
  network 19.0.0.0/8
  network 20.0.0.0/8
  network 21.0.0.0/8
  network 22.0.0.0/8
  network 23.0.0.0/8
  network 24.0.0.0/8
  network 26.0.0.0/8
  network 28.0.0.0/8
  network 29.0.0.0/8
  network 30.0.0.0/8
  network 32.0.0.0/8
  network 33.0.0.0/8
  network 34.0.0.0/8
  network 35.0.0.0/8
  network 38.0.0.0/8
  network 40.0.0.0/8
  network 44.0.0.0/8
  network 45.0.0.0/8
  network 47.0.0.0/8
  network 48.0.0.0/8
  network 50.0.0.0/8
  network 52.0.0.0/8
  network 53.0.0.0/8
  network 54.0.0.0/8
  network 55.0.0.0/8
  network 56.0.0.0/8
  network 57.0.0.0/8
  network 63.0.0.0/8
  network 64.0.0.0/8
  network 65.0.0.0/8
  network 66.0.0.0/8
  network 67.0.0.0/8
  network 68.0.0.0/8
  network 69.0.0.0/8
  network 70.0.0.0/8
  network 71.0.0.0/8
```

```
  network 72.0.0.0/8
  network 73.0.0.0/8
  network 74.0.0.0/8
  network 75.0.0.0/8
  network 76.0.0.0/8
  network 96.0.0.0/8
  network 97.0.0.0/8
  network 98.0.0.0/8
  network 99.0.0.0/8
  network 100.0.0.0/8
  network 104.0.0.0/8
  network 107.0.0.0/8
  network 108.0.0.0/8
  network 128.0.0.0/8
  network 129.0.0.0/8
  network 130.0.0.0/8
  network 131.0.0.0/8
  network 132.0.0.0/8
  network 134.0.0.0/8
  network 135.0.0.0/8
  network 136.0.0.0/8
  network 137.0.0.0/8
  network 138.0.0.0/8
  network 139.0.0.0/8
  network 140.0.0.0/8
  network 142.0.0.0/8
  network 143.0.0.0/8
  network 144.0.0.0/8
  network 146.0.0.0/8
  network 147.0.0.0/8
  network 148.0.0.0/8
  network 149.0.0.0/8
  network 152.0.0.0/8
  network 155.0.0.0/8
  network 156.0.0.0/8
  network 157.0.0.0/8
  network 158.0.0.0/8
  network 159.0.0.0/8
  network 160.0.0.0/8
  network 161.0.0.0/8
  network 162.0.0.0/8
  network 164.0.0.0/8
  network 165.0.0.0/8
  network 166.0.0.0/8
  network 167.0.0.0/8
  network 168.0.0.0/8
  network 169.0.0.0/8
  network 170.0.0.0/8
  network 172.0.0.0/8
  network 173.0.0.0/8
  network 174.0.0.0/8
  network 184.0.0.0/8
  network 192.0.0.0/8
  network 198.0.0.0/8
  network 199.0.0.0/8
  network 204.0.0.0/8
  network 205.0.0.0/8
  network 206.0.0.0/8
  network 207.0.0.0/8
  network 208.0.0.0/8
  network 209.0.0.0/8
  network 214.0.0.0/8
  network 215.0.0.0/8
  network 216.0.0.0/8
 exit-address-family
!
 rfp full-table-download off
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix CCC

Router 01 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-us
!
ip route 167.2.127.0/24 167.2.126.2
!
line vty
!
```

Appendix DDD

Router 01 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-us
!
```

Appendix EEE

Router 01 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-us
!
!
!
interface net1
 description US to Europe Link
 ip address 206.126.236.2/30
!
interface net2
 description US to Latin America Link
 ip address 4.69.184.1/30
!
interface net3
 description US to Oceania Link
 ip address 114.31.199.2/30
!
interface net4
 description US Service Link
 ip address 1.0.0.2/8
 ip address 100.0.0.1/8
 ip address 104.0.0.1/8
 ip address 107.0.0.1/8
 ip address 108.0.0.1/8
 ip address 11.0.0.1/8
 ip address 12.0.0.1/8
 ip address 128.0.0.1/8
 ip address 129.0.0.1/8
 ip address 13.0.0.1/8
 ip address 130.0.0.1/8
 ip address 131.0.0.1/8
 ip address 132.0.0.1/8
 ip address 134.0.0.1/8
 ip address 135.0.0.1/8
 ip address 136.0.0.1/8
 ip address 137.0.0.1/8
 ip address 138.0.0.1/8
 ip address 139.0.0.1/8
 ip address 140.0.0.1/8
 ip address 142.0.0.1/8
 ip address 143.0.0.1/8
 ip address 144.0.0.1/8
 ip address 146.0.0.1/8
 ip address 147.0.0.1/8
 ip address 148.0.0.1/8
 ip address 149.0.0.1/8
 ip address 15.0.0.1/8
 ip address 152.0.0.1/8
 ip address 155.0.0.1/8
 ip address 156.0.0.1/8
 ip address 157.0.0.1/8
 ip address 158.0.0.1/8
 ip address 159.0.0.1/8
 ip address 16.0.0.1/8
 ip address 160.0.0.1/8
 ip address 161.0.0.1/8
 ip address 162.0.0.1/8
 ip address 164.0.0.1/8
```

```
ip address 165.0.0.1/8
ip address 166.0.0.1/8
ip address 167.0.0.1/8
ip address 168.0.0.1/8
ip address 169.0.0.1/8
ip address 17.0.0.1/8
ip address 170.0.0.1/8
ip address 172.0.0.1/8
ip address 173.0.0.1/8
ip address 174.0.0.1/8
ip address 18.0.0.1/8
ip address 184.0.0.1/8
ip address 19.0.0.1/8
ip address 192.0.0.1/8
ip address 198.0.0.1/8
ip address 199.0.0.1/8
ip address 20.0.0.1/8
ip address 204.0.0.1/8
ip address 205.0.0.1/8
ip address 206.0.0.1/8
ip address 207.0.0.1/8
ip address 208.0.0.1/8
ip address 209.0.0.1/8
ip address 21.0.0.1/8
ip address 214.0.0.1/8
ip address 215.0.0.1/8
ip address 216.0.0.1/8
ip address 22.0.0.1/8
ip address 23.0.0.1/8
ip address 24.0.0.1/8
ip address 26.0.0.1/8
ip address 28.0.0.1/8
ip address 29.0.0.1/8
ip address 3.0.0.1/8
ip address 30.0.0.1/8
ip address 32.0.0.1/8
ip address 33.0.0.1/8
ip address 34.0.0.1/8
ip address 35.0.0.1/8
ip address 38.0.0.1/8
ip address 4.0.0.1/8
ip address 40.0.0.1/8
ip address 44.0.0.1/8
ip address 45.0.0.1/8
ip address 47.0.0.1/8
ip address 48.0.0.1/8
ip address 50.0.0.1/8
ip address 52.0.0.1/8
ip address 53.0.0.1/8
ip address 54.0.0.1/8
ip address 55.0.0.1/8
ip address 56.0.0.1/8
ip address 57.0.0.1/8
ip address 6.0.0.1/8
ip address 63.0.0.1/8
ip address 64.0.0.1/8
ip address 65.0.0.1/8
ip address 66.0.0.1/8
ip address 67.0.0.1/8
ip address 68.0.0.1/8
ip address 69.0.0.1/8
ip address 7.0.0.1/8
ip address 70.0.0.1/8
ip address 71.0.0.1/8
ip address 72.0.0.1/8
ip address 73.0.0.1/8
ip address 74.0.0.1/8
ip address 75.0.0.1/8
ip address 76.0.0.1/8
ip address 8.0.0.1/8
ip address 9.0.0.1/8
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

```
 ip address 96.0.0.1/8
 ip address 97.0.0.1/8
 ip address 98.0.0.1/8
 ip address 99.0.0.1/8
!
interface net5
 description External Link
 ip address 167.2.126.1/24
!
no ipv6 forwarding
!
!
!
line vty
!
```

Appendix FFF

Router 02 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/01/08 06:18:10
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-eu
!
!
router bgp 3257
 neighbor 193.0.0.2 remote-as 24441
 neighbor 195.66.224.2 remote-as 37474
 neighbor 206.126.236.2 remote-as 6939
 !
 address-family ipv4 unicast
  network 2.0.0.0/8
  network 31.0.0.0/8
  network 46.0.0.0/8
  network 51.0.0.0/8
  network 62.0.0.0/8
  network 77.0.0.0/8
  network 78.0.0.0/8
  network 79.0.0.0/8
  network 80.0.0.0/8
  network 81.0.0.0/8
  network 82.0.0.0/8
  network 83.0.0.0/8
  network 84.0.0.0/8
  network 85.0.0.0/8
  network 86.0.0.0/8
  network 88.0.0.0/8
  network 89.0.0.0/8
  network 90.0.0.0/8
  network 91.0.0.0/8
  network 92.0.0.0/8
  network 93.0.0.0/8
  network 109.0.0.0/8
  network 141.0.0.0/8
  network 145.0.0.0/8
  network 151.0.0.0/8
  network 176.0.0.0/8
  network 188.0.0.0/8
  network 193.0.0.0/8
  network 194.0.0.0/8
  network 195.0.0.0/8
  network 212.0.0.0/8
  network 213.0.0.0/8
  network 217.0.0.0/8
 exit-address-family
!
!
line vty
!
```

Appendix GGG

Router 02 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-eu
!
!
line vty
!
```

Appendix HHH

Router 02 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-eu
!
```

Appendix III

Router 02 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/01/08 06:18:10
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-eu
!
!
!
interface net1
 ip address 206.126.236.1/30
!
interface net2
 description Europe to Africa Link
 ip address 195.66.224.1/30
!
interface net3
 description Europe to Asia Link
 ip address 193.0.0.1/30
!
interface net4
 description Europe Service Link
 ip address 2.0.0.1/8
 ip address 31.0.0.1/8
 ip address 46.0.0.1/8
 ip address 51.0.0.1/8
 ip address 62.0.0.1/8
 ip address 77.0.0.1/8
 ip address 78.0.0.1/8
 ip address 79.0.0.1/8
 ip address 80.0.0.1/8
 ip address 81.0.0.1/8
 ip address 82.0.0.1/8
 ip address 83.0.0.1/8
 ip address 84.0.0.1/8
 ip address 85.0.0.1/8
 ip address 86.0.0.1/8
 ip address 88.0.0.1/8
 ip address 89.0.0.1/8
 ip address 90.0.0.1/8
 ip address 91.0.0.1/8
 ip address 92.0.0.1/8
 ip address 93.0.0.1/8
 ip address 109.0.0.1/8
 ip address 141.0.0.1/8
 ip address 145.0.0.1/8
 ip address 151.0.0.1/8
 ip address 176.0.0.1/8
 ip address 188.0.0.1/8
 ip address 193.0.0.1/8
 ip address 194.0.0.1/8
 ip address 195.0.0.1/8
 ip address 212.0.0.1/8
 ip address 213.0.0.1/8
 ip address 217.0.0.1/8
!
no ipv6 forwarding
!
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix JJJ

Router 03 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/01/09 02:26:41
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-as
!
!
router bgp 24441
 neighbor 193.0.0.1 remote-as 3257
 neighbor 202.93.8.2 remote-as 4826
 !
 address-family ipv4 unicast
  network 14.0.0.0/8
  network 27.0.0.0/8
  network 39.0.0.0/8
  network 42.0.0.0/8
  network 49.0.0.0/8
  network 59.0.0.0/8
  network 60.0.0.0/8
  network 61.0.0.0/8
  network 101.0.0.0/8
  network 106.0.0.0/8
  network 110.0.0.0/8
  network 111.0.0.0/8
  network 112.0.0.0/8
  network 113.0.0.0/8
  network 115.0.0.0/8
  network 116.0.0.0/8
  network 117.0.0.0/8
  network 118.0.0.0/8
  network 119.0.0.0/8
  network 120.0.0.0/8
  network 121.0.0.0/8
  network 122.0.0.0/8
  network 123.0.0.0/8
  network 124.0.0.0/8
  network 125.0.0.0/8
  network 126.0.0.0/8
  network 133.0.0.0/8
  network 153.0.0.0/8
  network 163.0.0.0/8
  network 171.0.0.0/8
  network 175.0.0.0/8
  network 180.0.0.0/8
  network 182.0.0.0/8
  network 183.0.0.0/8
  network 202.0.0.0/8
  network 211.0.0.0/8
  network 218.0.0.0/8
  network 219.0.0.0/8
  network 221.0.0.0/8
  network 222.0.0.0/8
  network 223.0.0.0/8
  network 5.0.0.0/8
  network 37.0.0.0/8
  network 87.0.0.0/8
  network 94.0.0.0/8
  network 95.0.0.0/8
  network 185.0.0.0/8
 exit-address-family
!
!
line vty
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix KKK

Router 03 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-as
!
!
line vty
!
```

Appendix LLL

Router 03 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-as
!
```

Appendix MMM

Router 03 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/01/09 02:26:41
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-as
!
!
!
interface net1
 description Asia to Europe Link
 ip address 193.0.0.2/30
!
interface net2
 description Asia to Oceania Link
 ip address 202.93.8.1/30
!
interface net3
 description Asia Service Link
 ip address 14.0.0.1/8
 ip address 27.0.0.1/8
 ip address 39.0.0.1/8
 ip address 42.0.0.1/8
 ip address 49.0.0.1/8
 ip address 59.0.0.1/8
 ip address 60.0.0.1/8
 ip address 61.0.0.1/8
 ip address 101.0.0.1/8
 ip address 106.0.0.1/8
 ip address 110.0.0.1/8
 ip address 111.0.0.1/8
 ip address 112.0.0.1/8
 ip address 113.0.0.1/8
 ip address 115.0.0.1/8
 ip address 116.0.0.1/8
 ip address 117.0.0.1/8
 ip address 118.0.0.1/8
 ip address 125.0.0.1/8
 ip address 126.0.0.1/8
 ip address 133.0.0.1/8
 ip address 153.0.0.1/8
 ip address 119.0.0.1/8
 ip address 120.0.0.1/8
 ip address 121.0.0.1/8
 ip address 122.0.0.1/8
 ip address 123.0.0.1/8
 ip address 124.0.0.1/8
 ip address 163.0.0.1/8
 ip address 171.0.0.1/8
 ip address 175.0.0.1/8
 ip address 180.0.0.1/8
 ip address 182.0.0.1/8
 ip address 183.0.0.1/8
 ip address 202.0.0.1/8
 ip address 211.0.0.1/8
 ip address 218.0.0.1/8
 ip address 219.0.0.1/8
 ip address 221.0.0.1/8
 ip address 222.0.0.1/8
 ip address 223.0.0.1/8
 ip address 5.0.0.1/8
 ip address 37.0.0.1/8
```

```
 ip address 87.0.0.1/8
 ip address 94.0.0.1/8
 ip address 95.0.0.1/8
 ip address 185.0.0.1/8

!
no ipv6 forwarding
!
!
!
line vty
!
```

Appendix NNN

Router 04 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/01/09 02:28:39
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-oc
!
!
router bgp 4826
 neighbor 114.31.199.2 remote-as 6939
 neighbor 202.93.8.1 remote-as 24441
 !
 address-family ipv4 unicast
  network 36.0.0.0/8
  network 43.0.0.0/8
  network 58.0.0.0/8
  network 103.0.0.0/8
  network 114.0.0.0/8
  network 150.0.0.0/8
  network 203.0.0.0/8
  network 210.0.0.0/8
 exit-address-family
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix OOO

Router 04 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-oc
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix PPP

Router 04 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-oc
!
```

Appendix QQQ

Router 04 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/01/09 02:28:39
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-oc
!
!
!
interface net1
 description Oceania to Asia Link
 ip address 202.93.8.2/30
!
interface net2
 description Oceania to US Link
 ip address 114.31.199.1/30
!
interface net3
 description Oceania Service Link
 ip address 36.0.0.1/8
 ip address 43.0.0.1/8
 ip address 58.0.0.1/8
 ip address 103.0.0.1/8
 ip address 114.0.0.1/8
 ip address 150.0.0.1/8
 ip address 203.0.0.1/8
 ip address 210.0.0.1/8
!
no ipv6 forwarding
!
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix RRR

Router 05 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/01/08 04:25:36
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-la
!
!
router bgp 52871
 neighbor 4.69.184.1 remote-as 6939
 !
 address-family ipv4 unicast
  network 177.0.0.0/8
  network 179.0.0.0/8
  network 181.0.0.0/8
  network 186.0.0.0/8
  network 187.0.0.0/8
  network 189.0.0.0/8
  network 190.0.0.0/8
  network 191.0.0.0/8
  network 200.0.0.0/8
  network 201.0.0.0/8
 exit-address-family
!
!
line vty
!
```

Appendix SSS

Router 05 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-la
!
!
line vty
!
```

Appendix TTT

Router 05 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-la
!
```

Appendix UUU

Router 05 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/01/08 04:25:36
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-la
!
!
!
interface net1
 description Latin America to US Link
 ip address 4.69.184.2/30
 no shut
!
interface net2
 description Latin America Service Link
 no shut
 ip address 177.0.0.1/8
 ip address 179.0.0.1/8
 ip address 181.0.0.1/8
 ip address 186.0.0.1/8
 ip address 187.0.0.1/8
 ip address 189.0.0.1/8
 ip address 190.0.0.1/8
 ip address 191.0.0.1/8
 ip address 200.0.0.1/8
 ip address 201.0.0.1/8
!
no ipv6 forwarding
!
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix VVV

Router 06 bgpd.conf

```
!
! Zebra configuration saved from vty
!   2019/01/07 04:49:44
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-af
!
router bgp 37474
 neighbor 195.66.224.1 remote-as 3257
!
 address-family ipv4 unicast
  network 41.0.0.0/8
  network 102.0.0.0/8
  network 105.0.0.0/8
  network 154.0.0.0/8
  network 196.0.0.0/8
  network 197.0.0.0/8
 exit-address-family
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com

Appendix WWW

Router 06 staticd.conf

```
!
! Zebra configuration saved from vty
!   2019/03/20 05:49:55
!
frr version 6.1-dev_git1042702486119
frr defaults traditional
!
hostname rtr-af
!
!
line vty
!
```

Appendix XXX

Router 06 vtysh.conf

```
!
! Sample configuration file for vtysh.
!
hostname rtr-af
!
!
```

Appendix YYY

Router 06 zebra.conf

```
!
! Zebra configuration saved from vty
!   2019/01/07 04:49:44
!
frr version 5.0.1
frr defaults traditional
!
hostname rtr-af
!
!
!
interface net1
 description Africa to Europe Link
 ip address 195.66.224.2/30
 no shut
!
interface net2
 description Africa Service Link
 no shut
 ip address 41.0.0.1/8
 ip address 102.0.0.1/8
 ip address 105.0.0.1/8
 ip address 154.0.0.1/8
 ip address 196.0.0.1/8
 ip address 197.0.0.1/8
!
no ipv6 forwarding
!
!
!
line vty
!
```

Bryan Scarbrough, bryan.scarbrough@gmail.com