Security of Containers

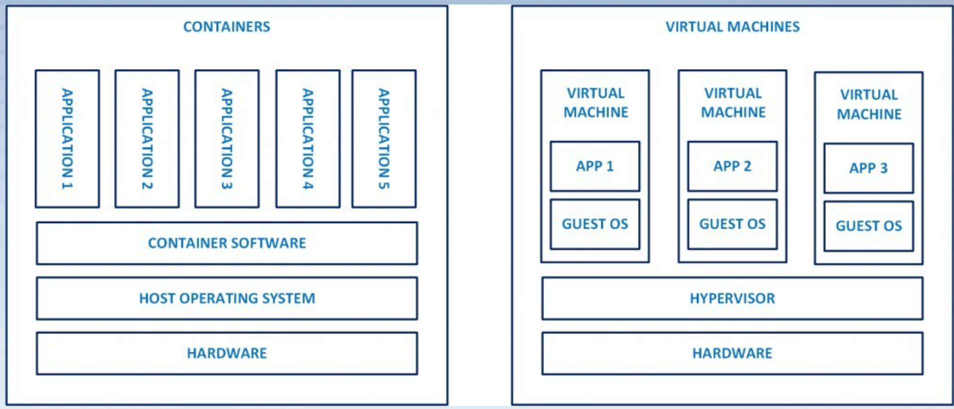
Shruti Kulkarni

About myself

- Enterprise Security Architect
- 12+ years of experience in Information Security
- Experience with security frameworks like ISO27001 and PCI-DSS
- Policies, Processes and Technology
- CISA, CRISC, CISSP, CCSK, ITILv3
- Architecture, Design, Implementation and Maintenance of security systems

What are containers?

- An application container is a mechanism that is used to isolate applications from each other within the context of a running operating system instance.
  - Conceptually, an application container is similar to a jail filesystem. When configured to do so, applications contained within a container cannot access components outside of the established boundary.
  - A container has a segmented network stack, process space and instance of a filesystem
  - A container shares the operating system with other containers that are running on the host
- Containers like any other asset need to be assured for security



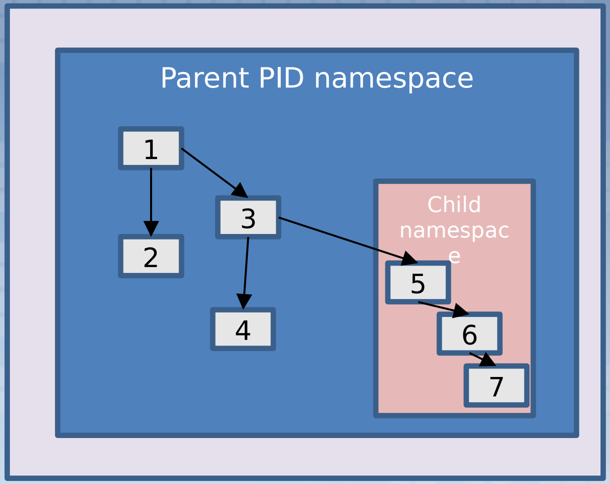
OWASP
Open Web Application Security Project

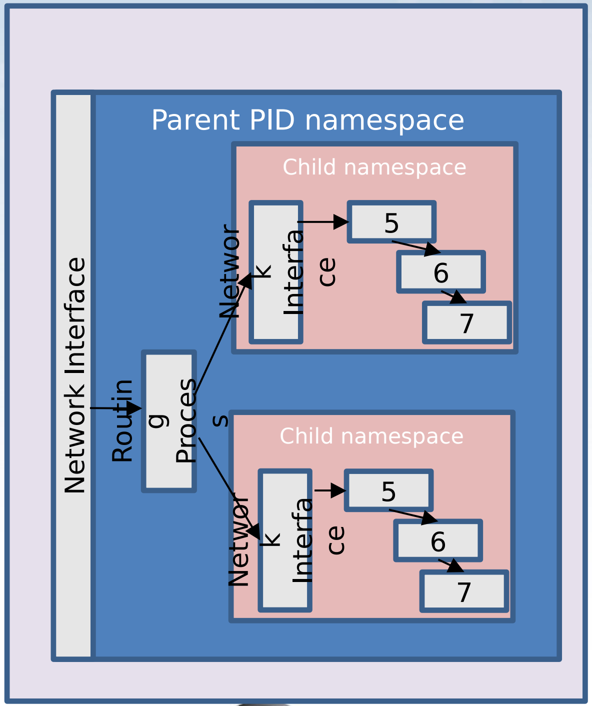
Uses of containers

- Malware analysis
- Easier deployment of repetitive jobs
- Micro-services
- Deployments to hybrid cloud, multi-cloud

Background of containers

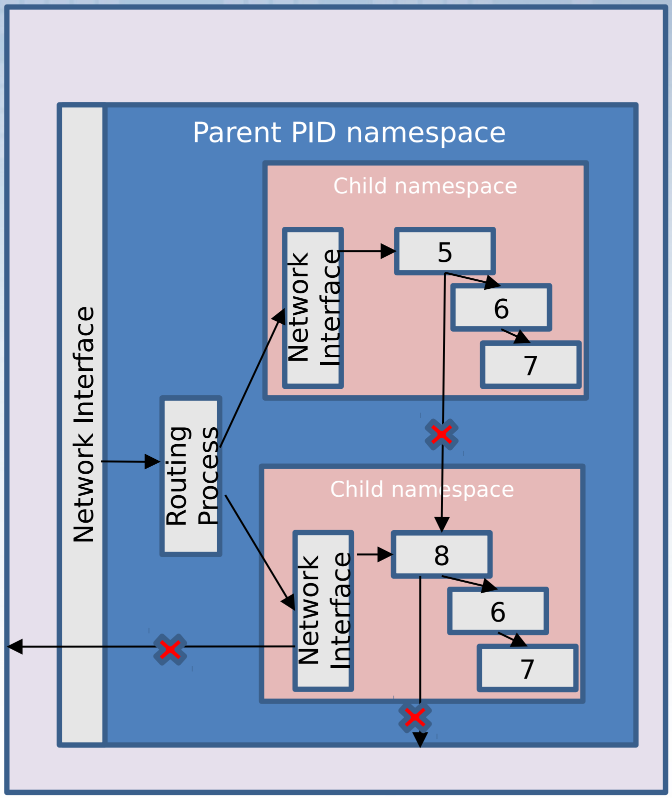
- Linux features
  - Namespaces –
    - CGroups – Isolate and manage resources
    - PID namespaces – Creates hierarchies of isolated sub-processes
    - IPC – Inter-process communication
    - Network – Separates network stack
    - Mount – mount and unmount file systems without affecting the host file system
    - User – Separates user ids and host ids between hosts and containers
    - UTS – provide isolation of two system identifiers: the hostname and the NIS domain name
  - chroot – allows a process to change its root file system





Security in Containers

- What are the risks
  - Unauthorised inter-process communication (5 -> 8)
  - A process running as root (8 running as root)
  - Unauthorised network connections
  - Denial of Service via excessive consumption of resources



- How are they addressed
  - Namespaces
  - SELinux
  - AppArmor
  - Seccomp BPF

Container images

- Container image
  - At rest
- Container
  - When running

Who does the Heavy Lifting?

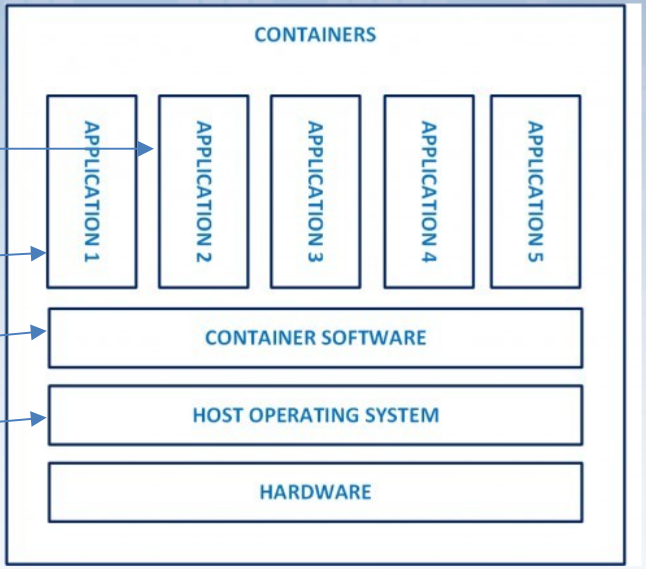
- Container runtimes / container software
  - Unpacks required files and metadata of an image before handing off to kernel
  - Makes API call to kernel
  - Initiates isolation and mounts the files
  - Responsible for:
    - Handling user input
    - Handling input over an API often from a Container Orchestrator
    - Preparing a container mount point

Connecting them all

- Container orchestration
  - Pulls images from registry
  - Schedules workloads within a cluster

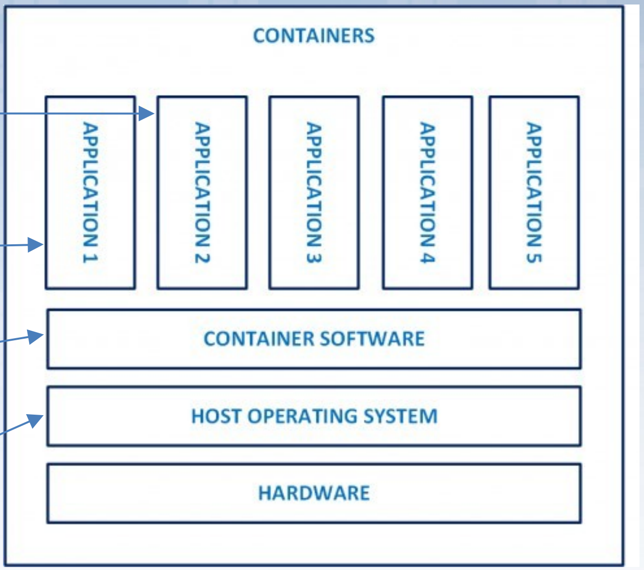
Why is security assurance for containers essential?

- Attack surface includes:
  - Isolation of containers
  - Configuration of containers
  - Container software
  - Host operating system
- And the applications themselves

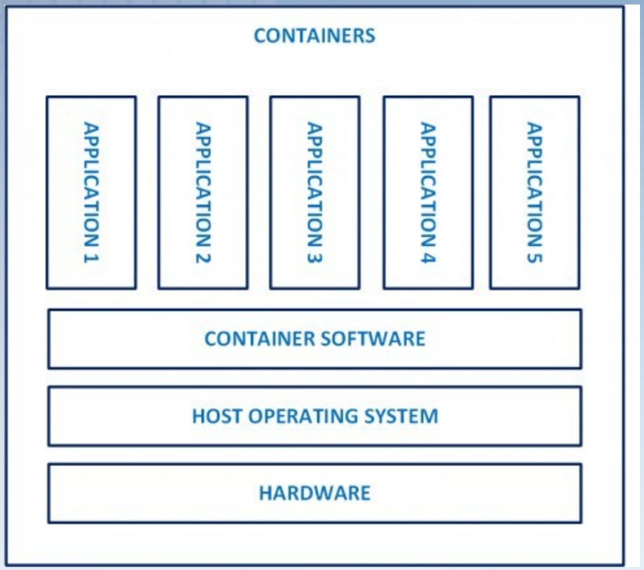


Why is security assurance for containers essential?

- Threats include:
  - Lateral privilege escalation
  - Exploitation of lack of secure configs
  - Exploitation of vulnerabilities in container software
  - Exploitation of vulnerabilities on host operating system
- Exploitation of vulnerabilities and configurations of the applications



The Start – Operating System layer



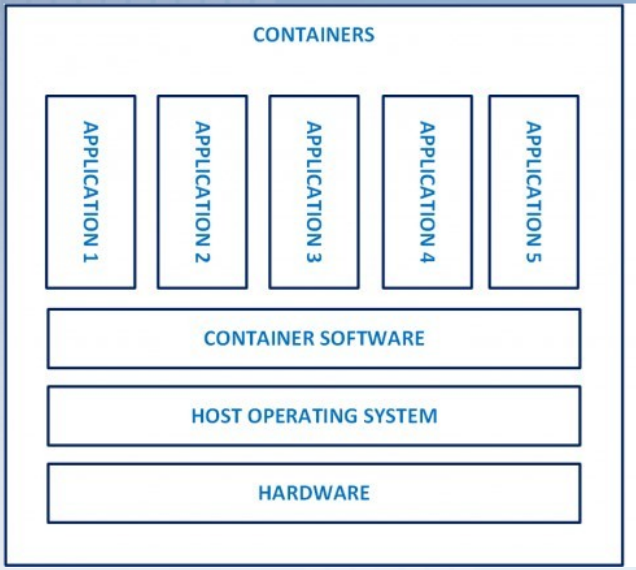
- Controls
  - Operating Systems
    - Secure configurations are critical. Do not mix containerized and non-containerized workloads on the same host instance
    - Apply security controls outlined for operating systems (hardening, patching, FIM, access control)
  - Network
    - Implement & enforce network segmentation
    - Following are the network configurations that apply in this scenario
      - Bridge
      - Host
      - None

- Vulnerabilities:
  - Operating systems have vulnerabilities if they are not hardened by removing unwanted services and protocols.
  - Operating systems are applications and have coding errors. Coding errors become vulnerabilities if they are not patched on time.
  - Incorrectly applied access control opens an attack surface that can be used by malicious actors
  - So on

Container specific OS (Examples include CoreOS Container Linux, Project Atomic Google Container-Optimized OS, Bottlerocket from AWS)

OWASP
Open Web Application Security Project

The Next Layer – Container Software / Runtime



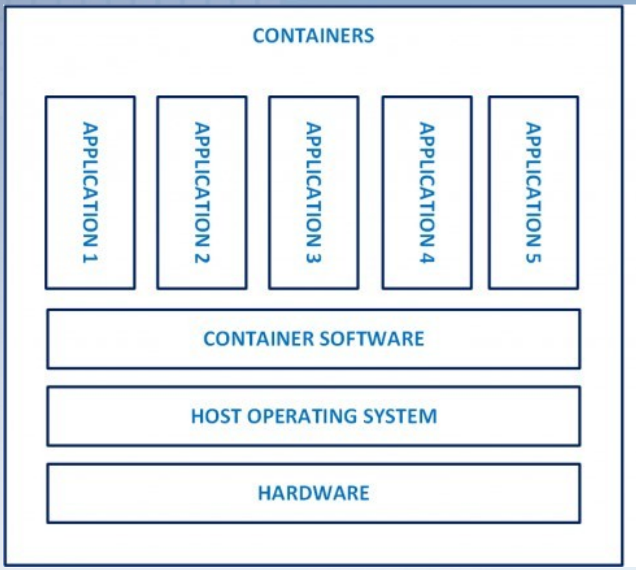
- Controls
  - Runtime
    - Keep container software up to date
    - Use role-based access control (RBAC) to restrict access to key components
  - Runtime security
    - Document known container processes
    - Monitor writes to operating system
    - Document and monitor network traffic
    - Explicitly define the permissions required by the container and its components

- Vulnerabilities:
  - Malicious actors trying to break out of container isolation
  - Missing patches on container software / runtime
  - /bin or /etc writes on operating systems
  - Ingress / egress traffic from containers
  - Rogue containers

Container Software / runtimes (runC, Docker, AWS Fargate, Google Kubernetes Engine, Amazon ECS, LXC)

OWASP
Open Web Application Security Project

The Next Layer – Storage and retrieval

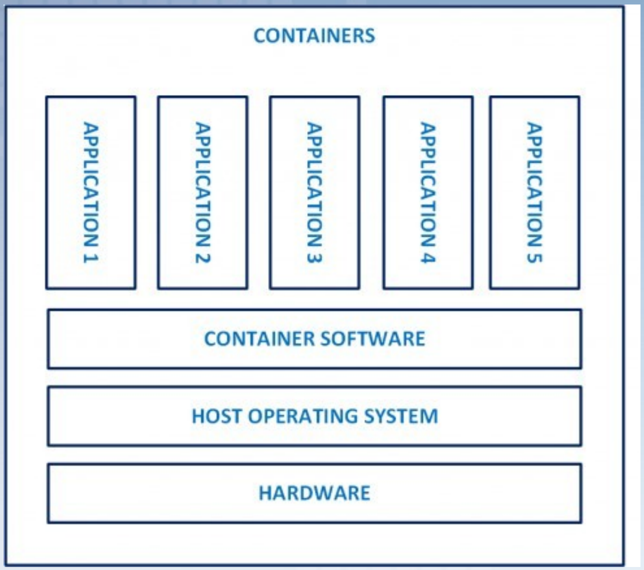


- Controls
  - Registry
    - Secure connection to registries (over TLS connection)
    - Secure images stored in registries
    - Role based access control applied to registries
    - Scan images
  - Orchestration
    - Access control (based on least privileges and separation of duties)
    - Grouping containers of relative sensitivity
    - Introducing nodes securely into cluster
    - Ensure that only containers with the same level of exposure (e.g. Internet facing) run on the same node.
    - Ensure that only containers with the same data classification level run on the same node.
    - Ensure that containers that are no longer needed are deleted.
    - Rotate the keys used by the orchestration process on a regular basis
    - Monitor Orchestration tool for vulnerabilities and patch regularly
    - Enable role based access control
    - Use registry namespace

- Vulnerabilities:
  - Missing access control
  - Lack of secure connection
  - Security of the orchestration tool
  - Images created with sensitive data

- Docker registry, Azure registry, OpenShift registry, redhat registry, Google registry, Kubernetes

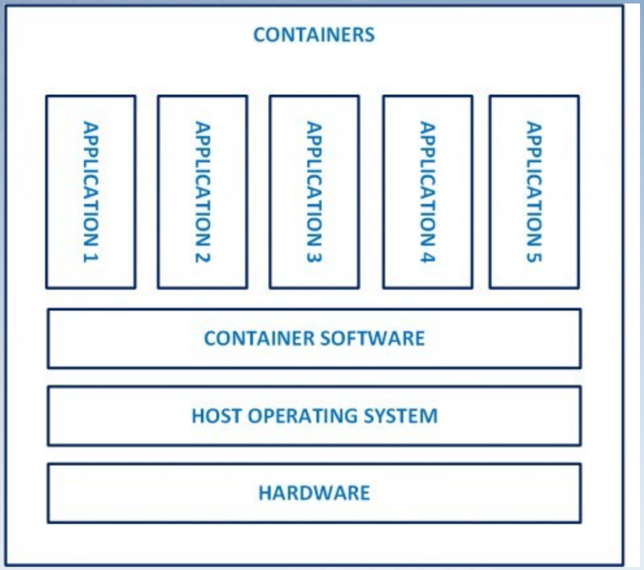
The Next Layer – Creation, testing and accreditation



- Control
  - Image creation and vulnerability scanning (*pipeline-based build approach, address component vulnerabilities and malicious code*)
  - Secure configuration best practices (*validation of configuration settings, access control – users running as non-privileged users, monitoring, no remote administration tools*)
  - Separating secrets from containers (*DB connection string part of a separate container with different access control, using secret management systems*)
  - Trusted images (*images with cryptographic signatures*)
- CI/CD pipeline

- Vulnerabilities:
  - Poisoned images
  - Embedded credentials
  - Lack of access control
  - Outdated components
  - And so on

Containers themselves



- Control
  - Use immutable images to instantiate containers
  - Enforce isolation/ segmentation by application / service / workload
  - Monitor containers for suspicious process or file system activity
  - Quarantine compromised containers
  - If not using immutable images, ensure an automated software update/replace process is implemented
  - Limit/ restrict access to resources (file system / kernel)
  - Regular automated security scans which cover the whole operating system and not just container related elements

- Vulnerabilities:
  - Unauthorised inter-process communication
  - Unauthorised breakout
  - Privilege escalation (account gaining uid0 privileges)
  - Stale images
  - Misconfigured applications running in containers
  - And so on

OWASP
Open Web Application Security Project

Summary

- Securing containers is not security of containers
- To ensure that you have a secure container ecosystem, secure the whole stack
  - Operating system
  - Container runtime
  - Image repositories
  - Orchestration tools
  - And the applications running in the containers

Questions

?