**Docker Container Security Implementation**

1. **Docker Security Best Practices:**

* Use as lightweight an image as possible to reduce the attack surface, such as distroless or alpine.
* Use only images from trusted sources and use only LKG images, avoiding using latest images until the newer or latest version is thoroughly tested internally and/or community affirmed.
* Use multi-staged builds to help reduce the final size of the docker image and more efficiently execute build steps in parallel.
* Avoid using root user account. Instead use a less privileged user account, with minimal needed access.
* Set container resource quotas to ensure that a container is unable to consume all available host node resources.
* Do not expose more ports than necessary.  
  **Example file:** <https://github.com/roostursauce/skillsshare/blob/main/alpine3-22-nginx.dockerfile>

1. **Kubernetes Security Configuration:**
   1. Some Available Security Context Features:
      1. Discretionary access.
      2. Allow or disallow privileged escalation on a container.
      3. Apply port bindings.
      4. Specify run\_as user accounts.
      5. Apply read Only access for root file systems (readOnlyRootFilesystem: true) where container images must run as root.
   2. Kubernetes YAML configuration with Security Context pod settings.
2. **IaaS Security Measures – IaaS Concepts and Security Implications**

For Infrastructure as a Service (IaaS) solutions and security, it is necessary to understand first which areas of responsibility belong to the consumer and which are the responsibility of the cloud service provider. In short, IaaS resources the CSP is responsible for everything below the OS, including the hypervisor, physical hardware, cooling, physical networking, storage solution options and management, as well as overall regional datacenter security/integrity. The stability of this base infrastructure is backed up by SLAs, which are ensured as best they can by facility, engineering staff, regular maintenance, patching and automated service healing which is purposefully obfuscated for the public cloud customer.

Of the available cloud service options (SaaS,PaaS, and IaaS), IaaS allows for the most flexibility offering full access to an OS, as well as choice of OS, memory, virtual CPU cores, infrastructure storage tiers, and even the chipsets in some cases. Because a customer’s AoR begins with the OS, it is necessary for a customer to consider some key categories about a VM, VM Scale Set, Containers and Kube controller nodes: Securing the resource, maintaining the OS and Applications, determining backup/Disaster recovery and data retention strategies, and access management to the given resources.

Using a VM as an example, on creation, a public IP is often provided by default, and if you choose to open the management of the VM via SSH/RDP, that will enable remote access to the VM to you and anyone with an internet connection until further action is taken with that NSG rule. All IaaS resources should follow a few strategies:

1. **Follow the model of Least-privilege for networks, resources, and user access to and from resources, in your design specifications.** These services in the end should not have access extending beyond their daily functions, and access to them should be limited to authorized IT parties and preferably for a limited time where possible. Network segmentation should allow for isolation from critical resources or data
2. **Public access to an IaaS resource should be limited to a non-management port and preferably behind a firewall** that allows DDOS protection, rate limiting, logging, and advanced detection features. At minimum an NSG and/or ASG should be used to control inbound/outbound services/IPs and ports.
3. **Resources which are mission critical should be made redundant, internally or externally and either locally or globally load balanced.** A minimum of 2 instances running the same service are generally required to achieve a 99.95% uptime SLA guaranteed by the cloud service. It is often wise to ensure that a service is extended to multiple cloud regions or availability zones within a given region to be additionally resilient to regional outages.
4. **Create a regular patch management schedule around best available low traffic times to apply OS and application patches which align with business policy.** This can be a manual process or automation via VM guest patching, configuration management, or new container image deployments.
5. **Remote access to an OS resource should be limited to authorized personnel only.** This can be best achieved over a Point to Site VPN connection to the internal network or in the case of VMs, Bastion remote management is also an option. Cloud shell, or Portal based-tools like Run-Command can provide web based CLI options for small tasks.
6. **Lastly, most IaaS resources incur costs by compute time and it is best to shut down/scale back a resource, when not in use.** For IaaS options such as VM Scalesets or Kubernetes clusters, scaling down during low traffic hours can save on overall costs. Similarly scaling up services for typical or unexpected high traffic situations help to ensure the resiliency of your service.