AWS re:Invent

WPS313

Container security and avoiding the 2 a.m. call

Len Henry
Senior Solutions Architect
WWPS/Education
Amazon Web Services

Ramesh Jetty
AWS Solutions Architect
WWPS/Education
Amazon Web Services

AWS re:Invent

aws

Agenda

- The dreaded call
- What container threats have you seen?
- Container security model
- Key questions
- Takeaways

The dreaded call

By the numbers

- **60% of organizations have experienced container security incidents last year**
- **47% deployed containers are known to have vulnerabilities**
- Top concerns
  - 54% – Inadequate container security knowledge among the teams
  - 52% – Limited visibility into security of containers/images
  - 43% – Inability to assess risk in container images prior to deployment
- Average cost of a **data breach** is **$3.92 million**
- Approximately **60%** of the small businesses that experience a data breach are **out of business** within six months

Source:
https://www.tripwire.com/state-of-security/devops/organizations-container-security-incident/
https://www.ibm.com/security/data-breach

The context of an attack

- You lose control of your infrastructure

  Someone else (possibly multiple people) are in control of your resources

  You are paying to allow someone else to use your resources

- Your business function may no longer execute

  The actor does not share your goal of maintaining your business

  They may shut down your business functions to do what they intend

- Your resources may be used for nefarious purposes

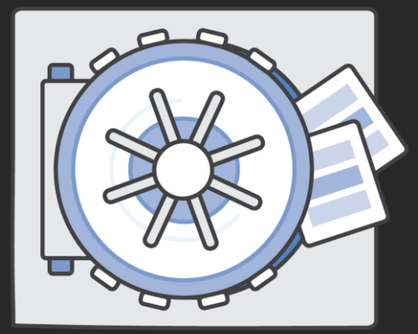
  More than likely your resources will be repurposed

  That purpose could be illegal and expose you

- The frequency of these events has increased for our customers

What container threats have you seen?

Container-specific threats (we have seen)



Container image corruption

Someone adds something to your image

- Example: Public images with embedded cryptocurrency-mining malware
- June 2018: Docker Hub – Used to mine cryptocurrency

Vulnerabilities in your platform

Use of dependencies can be exploited

- Example: Unsecured Kubernetes dashboard with cloud account credentials
- Feb. 2018: Tesla – Exploited to mine cryptocurrency
- June 2018: Weight Watchers – Not exploited

Containers are not hardened in the same way as your other servers

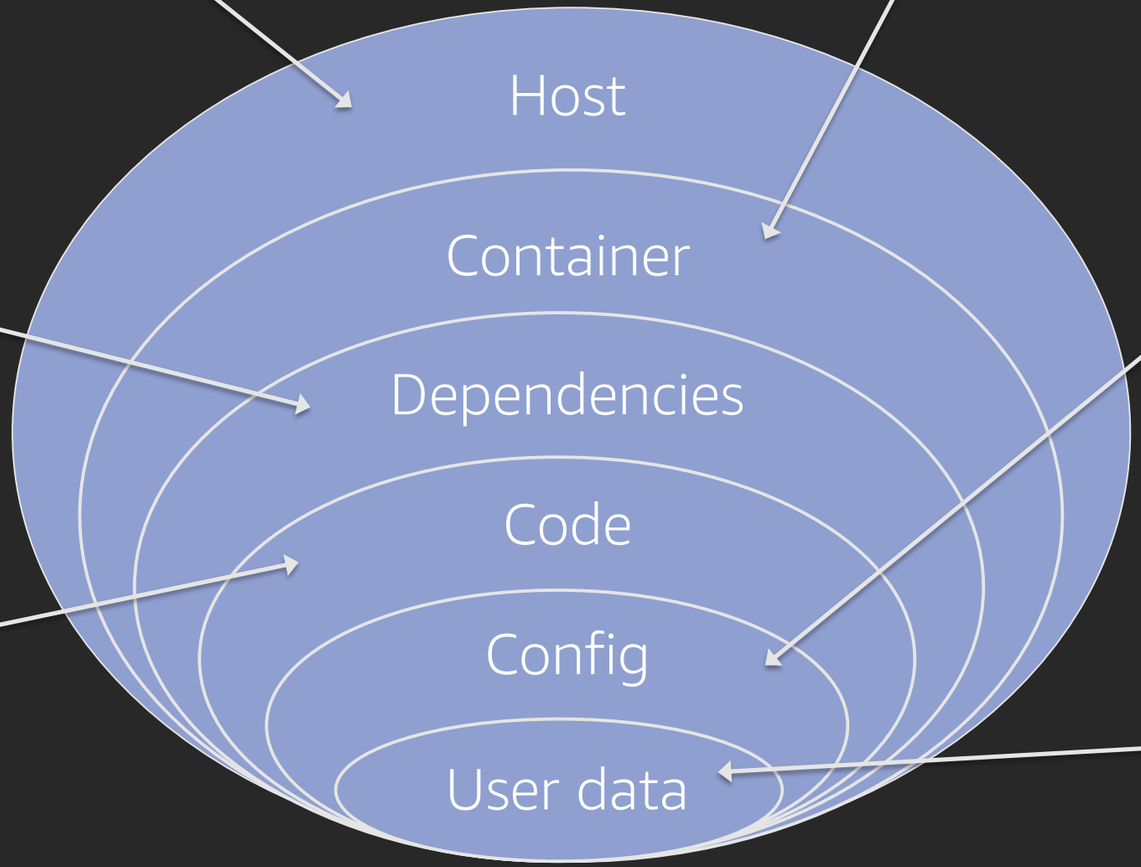
Tools and processes that work for server instances are not leveraged with your containers

Container security model

Container security model – Defense in depth

- Full-blown distro (Ubuntu, AL) vs. container-optimized distribution
- Multi-tenancy requirements
- **Gotchas:** Linux packages/CVEs, leaks

- Runtime/standards (OCI)
- Immutability of images
- All containers share a kernel (mitigation: Firecracker)
- **Gotchas:** Unnecessary privileged users, no scans, trust



- Code analysis
- Source available?
- **Gotchas:** Huge surface, many languages

- Sensitive config (passwords, API keys)
- **Gotchas:** Commits to source, non-separated access (dev has cleartext password)

- Sanitizing user input
- Static code analysis
- **Gotchas:** Log-leaking

- Business core data
- Personal identifiable information (PII)
- **Gotchas:** Leaks, GDPR (in Europe)

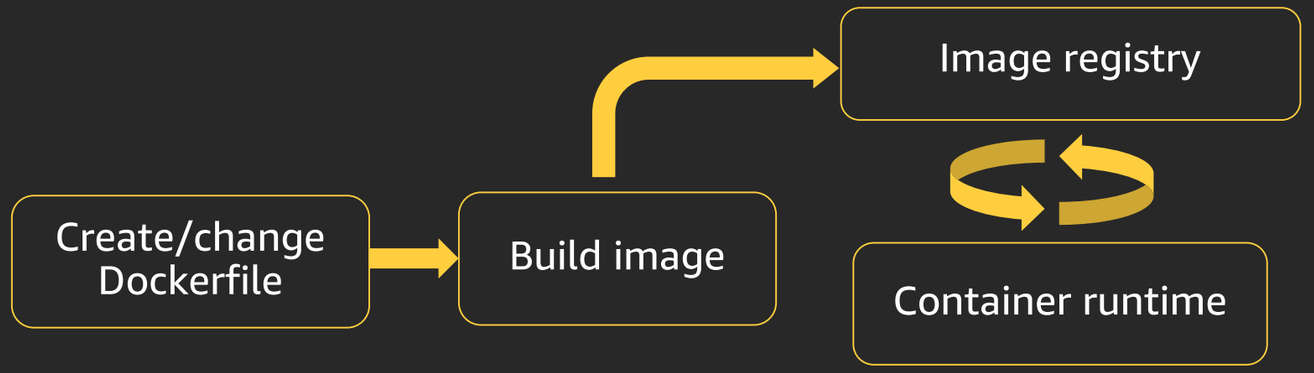
What are the key questions?

How do you secure the container lifecycle?

How do you secure the container lifecycle?

Registries

- Create private registries
- Keep images close to runtime (cache image copies in your registry)
- Use curated registry (official images on Docker Hub)



Images

- Scan images and perform static analysis
- Sign images and verify signature

clair

aqua

Build and automate container image continuous-delivery pipeline

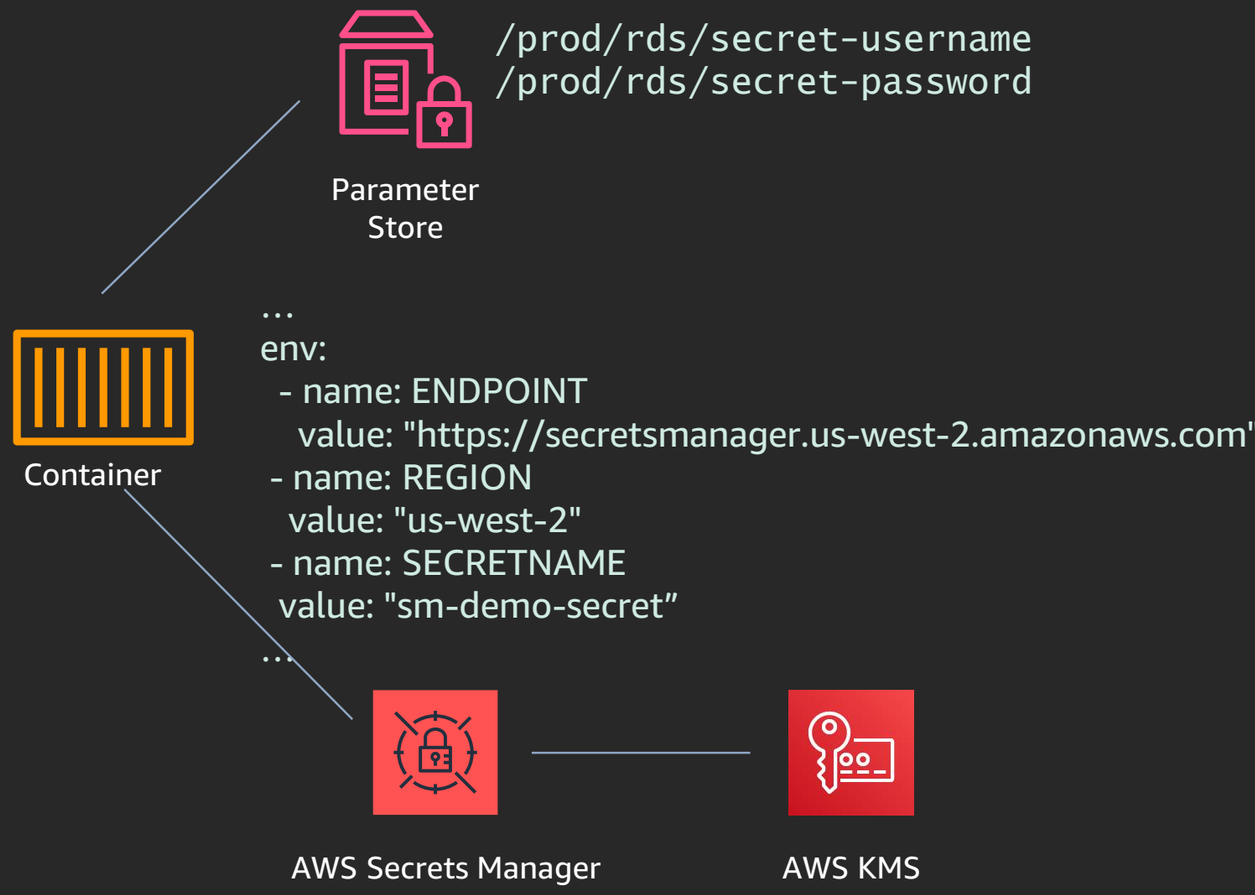
Demo

How do you manage secrets?

How do you manage secrets?

- Do not store secrets in the Dockerfile
- Leverage AWS Secrets Manager or SSM Parameter Store to store secrets

	SSM Parameter Store	AWS Secrets Manager
Encryption	AWS KMS	AWS KMS
Authentication/ authorization	AWS Identity and Access Management (IAM)	IAM
Secret rotation	Static	Dynamic



aws secretsmanager create-secret --name <SECRETNAME> --description "rds/secret" --secret-string [{"testkey1":"testvalue1"}] --region <REGION>

Demo

How do you secure a container host?

How do you secure a container host?

Hardening

- SE Linux
- Bastille Linux
- App Armor

Protection

- AWS Systems Manager (Patch Manager, State Manager)
- AWS WAF

Detection

- Amazon Inspector
- Amazon GuardDuty

AWS WAF

AWS Systems Manager

Amazon GuardDuty

Amazon Inspector

Demo

How do you gain visibility into a container-based workload?

Container logging and monitoring

Logging

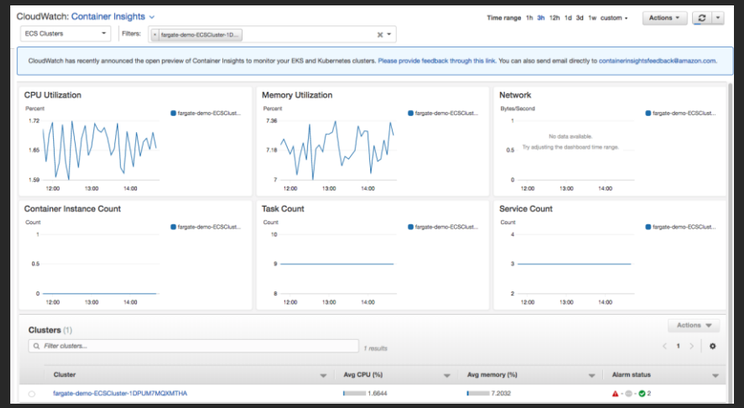
- Amazon CloudWatch Logs/awslogs
- Log routing (Fluentd, Fluent Bit, Logstash, FireLens)

Monitoring

- Amazon CloudWatch Events
- Container insights
- Third party – Twistlock, Aqua, NeuVector, etc.

Forensics

- Evolving space
- Docker ecosystem tools and AWS IR
- Third party/open-source – GRR, Sysdig, ThreatResponse



Demo

How do you securely network your container-based application?

How do you securely network your container-based application?

Least privilege

- IAM roles at container or task level

Network isolation

- Task networking
- VPC CNI plugin for Kubernetes
- Private subnets for your worker nodes

Service mesh

- Monitor and control microservice communications with AWS App Mesh

Third-party solutions

- Sysdig
- Qualys
- Tigera
- NeuVector

Typical features include:

- Configuration/asset management
- Reporting and dashboard
- Running container analysis
- Vulnerability assessment
- Network analysis and compliance

Takeaways

Takeaways

- Experiencing security exposure can be fatal to an organization
- AWS container services have more AWS responsibility than other services
- There is a structured approach to developing a secure container application

Related sessions

CON334-R1 – Running high-security workloads on Amazon EKS

Friday, Dec. 6, 9:15 a.m.–10:15 a.m., Venetian, Level 3, Lido 3005

CON315-R1 – Deep dive: Observability of Kubernetes applications

Friday, Dec. 6, 10:45 a.m.–11:45 a.m., Mirage, St. Thomas B

CON317-R2 – Securing your Amazon EKS cluster

Thursday, Dec. 5, 2:30 p.m.–3:30 p.m., Mirage, Events Center C1, Table 1

CON414-R3 – Security best practices for AWS Fargate

Friday, Dec. 6, 10 a.m.–11 a.m., Mirage, Grand Ballroom B, Table 8

CON317-R3 – Securing your Amazon EKS cluster

Friday, Dec. 6, 10 a.m.–11 a.m., Mirage, Events Center C1, Table 9

Thank you!

AWS re:Invent

aws

Please complete the session survey in the mobile app.