

Container Investigations

Containers

Full container images include a full operating system, offering more flexibility and familiarity for forensic investigations compared to distroless containers. However, they can also present a larger attack surface.

- **Richer File System:** Standard forensic artifacts and tools such as the `passwd` file and shells are present for traditional forensic analysis.
- **Diverse Tools:** A wider range of forensic tools can be leveraged due to the familiar Linux environment.
- **Focus on Application Logs:** Reliance on application logs and runtime inspection for evidence collection.

Distroless Containers

Distroless containers are a lightweight container image format that removes unnecessary operating system components. This minimizes attack surface but also presents challenges for traditional forensic methods.

Limited File System

Distroless containers may lack standard forensic artifacts such as `/etc/passwd` files or interactive shells. This absence complicates the ability to perform interactive investigations and retrieve traditional system information, necessitating alternative approaches for forensic analysis.

Lack of Diagnostic Tools

Distroless containers often exclude common diagnostic and debugging tools, such as `strace`, `lsof`, or `tcpdump`. This absence complicates the process of real-time analysis and troubleshooting within the container, necessitating external monitoring solutions.

Ephemeral Nature

Distroless containers are typically designed to be ephemeral, meaning they can be quickly destroyed and recreated. This transient nature makes it difficult to capture and preserve forensic evidence over time, as traditional methods rely on stable, persistent environments for comprehensive analysis.

Investigation Techniques

Image Inspection

Analyze the container image for vulnerabilities, suspicious binaries, or embedded secrets using tools like `docker inspect` or [Clair](#).

Runtime Analysis

Utilize container runtime security tools like [Azure Container Insights](#) or [Falco](#) to monitor container activity for anomalies.

Extracting Application Logs

Leverage log collectors like [Fluentd](#) or [Logstash](#) to capture application logs for analysis.

Debuggers

A separate container with debugging tools can sometimes be attached to the target container to inspect its runtime state.

Quick Tips for Container DFIR

Understand the Environment

- Identify the orchestrator (Kubernetes, Docker Swarm, etc.).
- Determine the container runtime (Docker, containerd, CRI-O, etc.).
- Identify where logs and configurations are stored.

Initial Steps

- Take snapshots of the running containers and their metadata. There are various options depending on orchestration in use.
- Collect configuration files and logs from the host and container.

Data Collection

- Gather container images and metadata.
- Extract logs from containers and orchestrators.
- Capture network traffic if possible.

Useful commands for investigating containers

Docker-Specific Commands

List containers: `docker ps -a`
Inspect a container: `docker inspect <container_id>`
Export container filesystem: `docker export <container_id> -o <container_id>.tar`
View container logs: `docker logs <container_id>`
Check network settings: `docker network inspect <network_name>`

Kubernetes-Specific Commands

List pods: `kubectl get pods --all-namespaces`
Describe a pod: `kubectl describe pod <pod_name> -n <namespace>`
Get pod logs: `kubectl logs <pod_name> -n <namespace>`
Export pod details: `kubectl get pod <pod_name> -n <namespace> -o yaml > pod_details.yaml`
Capture network traffic: `kubectl exec -it <pod_name> -n <namespace> -- tcpdump -i eth0 -w /tmp/capture.pcap`

Containerd Commands

List containers: `ctr -n k8s.io container ls`
Inspect a container: `ctr -n k8s.io container info <container_id>`
Export container filesystem: `ctr -n k8s.io snapshot mount <snapshot_id> /mnt`
View container logs: `journalctl CONTAINER_NAME=<container_name>`

Additional Tools

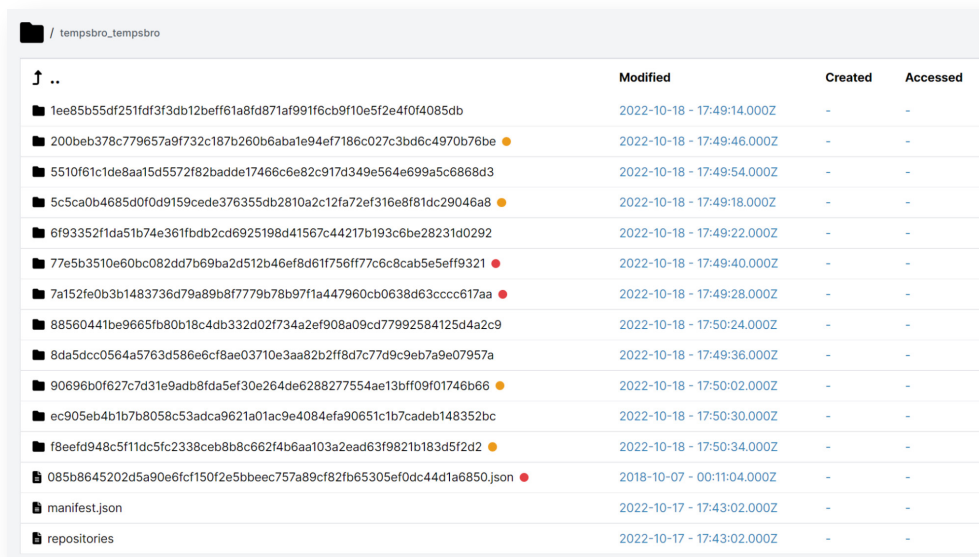
- Docker Explorer: A free tool from Google for Container forensics (<https://github.com/google/docker-explorer>)
- Docker Forensics Toolkit: Numerous scripts for forensicating Docker (<https://github.com/docker-forensics-toolkit/toolkit>)
- Distroless Forensics Tools: Tools like Distroless Forensics (<https://github.com/GoogleContainerTools/distroless/issues>) are under development to specifically address the challenges of distroless container forensics.

Remember

- Container investigations often require a combination of techniques and tools.
- Understanding container technology and the specific container image format is crucial.
- Consider incorporating these techniques into your overall Azure incident response plan.

How Cado Can Help

The Cado platform can natively import from a number of systems, including EKS/ECS/GKE/AKS and more generic Kubernetes installations. For more, see our blog on [Analyzing Docker Images in the Cado Platform](#).



..	Modified	Created	Accessed
1ee85b55df251fdf3f3db12beff61a8fd871af991f6cb9f10e5f2e4f0f4085db	2022-10-18 - 17:49:14.000Z	-	-
200beb378c779657a9f732c187b260b6aba1e94ef7186c027c3bd6c4970b76be	2022-10-18 - 17:49:46.000Z	-	-
5510f61c1de8aa15d5572f82badde17466c6e82c917d349e564e699a5c6868d3	2022-10-18 - 17:49:54.000Z	-	-
5c5ca0b4685d0f0d9159ced376355db2810a2c12fa72ef316e8f81dc29046a8	2022-10-18 - 17:49:18.000Z	-	-
6f93352f1da51b74e361fbd2cd6925198d41567c44217b193c6be28231d0292	2022-10-18 - 17:49:22.000Z	-	-
77e5b3510e60bc082dd7b69ba2d512b46ef8d61f756ff77c6c8cab5e5eff9321	2022-10-18 - 17:49:40.000Z	-	-
7a152fe0b3b1483736d79a89b8f7779b78b97f1a447960cb0638d63cccc617aa	2022-10-18 - 17:49:28.000Z	-	-
88560441be9665fb80b18c4db332d02f734a2ef908a09cd77992584125d4a2c9	2022-10-18 - 17:50:24.000Z	-	-
8da5dcc0564aa5763d586e6cf8ae03710e3aa82b2ff8d7c77d9c9eb7a9e07957a	2022-10-18 - 17:49:36.000Z	-	-
90696b0f627c7d31e9adb8fda5ef30e264de6288277554ae13bf09f01746b66	2022-10-18 - 17:50:02.000Z	-	-
ec905eb4b1b7b8058c53adca9621a01ac9e4084efa90651c1b7cadeb148352bc	2022-10-18 - 17:50:30.000Z	-	-
f8eefd948c5f11dc5fc2338ceb8b8c662f4b6aa103a2ead63f9821b183d5f2d2	2022-10-18 - 17:50:34.000Z	-	-
085b8645202d5a90e6f9c150f2e5bbeec757a89cf82fb65305ef0dc44d1a6850.json	2018-10-07 - 00:11:04.000Z	-	-
manifest.json	2022-10-17 - 17:43:02.000Z	-	-
repositories	2022-10-17 - 17:43:02.000Z	-	-

Further Resources

- Docker Security documentation (<https://docs.docker.com/engine/security/>)
- Cloud Native Security Foundation (CNSF) Container Security Working Group (<https://www.cncf.io/blog/2023/06/08/5g-deployment-as-simple-as-gitops-thanks-to-fluxcd/>)
- SANS Institute Cloud Security resources (<https://www.sans.org/cloud-security/>)