## Container Breakouts

The term “Container Breakout” refers to the event where a malicious or legitimate user is able to escape the container isolation and access resources (e.g. filesystem, processes, network interfaces) on the host machine. This section covers the different misconfigurations and excessive privileges that can be used to break out of the containers. The labs are based on an assumed breach approach which means that the lab starts when the attacker has already gained a command shell on the container.

### What will you learn?

- Exploiting misconfigurations for Docker breakout
- Leveraging excessive privileges to access Docker host
- Identifying and weaponizing additional Linux capabilities assigned to the container
- Targeting shared namespaces to breach the container isolation

**References:**

1. RunC container breakout vulnerability (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-5736)
2. Abusing SYS_MODULE capability to perform Docker container breakout (https://blog.pentesteracademy.com/abusing-sys-module-capability-to-perform-docker-container-breakout-cf5c29956edd)
3. Blackhat talk on Container Escapes (https://i.blackhat.com/USA-19/Thursday/us-19-Edwards-Compendium-Of-Container-Escapes-up.pdf)

**Labs:**

- Privileged Container
  In this lab, you will learn to break out of a privileged container. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Mount host disk inside the container
  - Break out of container using chroot

- Privileged Container II
  In this lab, you will learn to break out of a privileged container. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Mount host disk inside the container
  - Add a user to the host machine using chroot
  - Use newly added user to SSH into host machine

- Mounted Docker Socket
  In this lab, you will learn to break out of a container on which Docker socket is mounted. A non-exhaustive list of activities to be covered includes:
  - Locate mounted Docker socket
  - Use docker client to list Docker images present on Docker host
  - Run a container with host filesystem mounted to it
  - Use chroot to breakout of the container

- Process Injection
  In this lab, you will learn to break out of a container by abusing SYS_PTRACE capability. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Inject a Linux x64 Bind shell shellcode into a host machine process
  - Use docker client to list Docker images present on Docker host

In this lab, you will learn to break out of a container by abusing SYS_MODULE capability. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Compile a reverse shell payload as kernel module
  - Inject kernel module into host machine kernel
  - Get reverse shell session to execute commands on Docker host

- Abusing DAC_READ_SEARCH Capability
  In this lab, you will learn to break out of a container by abusing DAC_READ_SEARCH capability. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Identify the files mounted on container
  - Run exploit code to read shadow file from host machine
  - Crack root password using John The Ripper
  - Use root credentials to SSH into host machine

- Shared Network Namespace
  In this lab, you will learn to break out of a container by abusing shared namespace among containers. A non-exhaustive list of activities to be covered includes:
  - Exploit a web application to get command execution on web application container
  - Pivot over web application to access portainer web UI using reGeorg
  - Run a container with host filesystem mounted to it from portainer web UI
  - Use chroot to break out of the container

- Abusing DAC_OVERRIDE Capability
  In this lab, you will learn to break out of a container by abusing DAC_OVERRIDE capability. A non-exhaustive list of activities to be covered includes:
  - Check capabilities provided to the container
  - Read shadow file of host machine using exploit code
  - Add custom password for root in the copied file
  - Run exploit code to overwrite shadow file on the host machine
  - Use root credentials to SSH into host machine

Mounted Docker Socket — Start

Privileged Container — Start

Privileged Container II — Start

Process Injection — Start

Abusing SYS_MODULE Capability — Start