Theory: Docker Capabilities

In Linux, the root user has special privileges that allow them to perform various actions on the system. These privileges are divided into smaller units called capabilities. Each capability represents a specific power or ability.

For instance, one capability called CAP_CHOWN allows the root user to change the ownership of files, while another capability called CAP_DAC_OVERRIDE lets them bypass certain permission checks when reading, writing, or executing files. These capabilities break down the root user's powers into separate and more manageable units.

By using capabilities, the Linux kernel can grant specific permissions to different users or processes, rather than giving them unrestricted access as the root user. This helps improve security by limiting the privileges granted to each user or process, ensuring that they only have the necessary capabilities to perform their intended tasks.

By default, Docker drops all capabilities except [those needed] using a whitelist approach.The following list contains all capabilities that are enabled by default when you run a docker container with their descriptions from the capabilities man page:

- CHOWN: Make arbitrary changes to file UIDs and GIDs
- DAC_OVERRIDE: Discretionary access control (DAC) - Bypass file read, write, and execute permission checks
- FSETID: Don’t clear set-user-ID and set-group-ID mode bits when a file is modified; set the set-group-ID bit for a file whose GID does not match the file system or any of the supplementary GIDs of the calling process.
- FOWNER: Bypass permission checks on operations that normally require the file system UID of the process to match the UID of the file, excluding those operations covered by CAP_DAC_OVERRIDE and CAP_DAC_READ_SEARCH.
- MKNOD: Create special files using mknod(2)
- NET_RAW: Use RAW and PACKET sockets; bind to any address for transparent proxying.
- SETGID: Make arbitrary manipulations of process GIDs and supplementary GID list; forge GID when passing socket credentials via UNIX domain sockets; write a group ID mapping in a user namespace.
- SETUID: Make arbitrary manipulations of process UIDs; forge UID when passing socket credentials via UNIX domain sockets; write a user ID mapping in a user namespace.
- SETFCAP: Set file capabilities.
- SETPCAP: If file capabilities are not supported: grant or remove any capability in the caller’s permitted capability set to or from any other process.
- NET_BIND_SERVICE: Bind a socket to Internet domain privileged ports (port numbers

less than 1024).
- SYS_CHROOT: Use chroot(2) to change to a different root directory.
- KILL: Bypass permission checks for sending signals. This includes use of the ioctl(2) KDSIGACCEPT operation.
- AUDIT_WRITE: Write records to kernel auditing log.

Lab Reference: Docker Capabilities