🐳 Understanding cgroups and namespaces in Docker/ Containers

Modern container technologies like **Docker** are made possible by two key Linux kernel features:

1. **Namespaces** – provide *isolation*
2. **Control Groups (cgroups)** – provide *resource control*

Together, they form the foundation of container security, performance, and independence — without the need for a full-blown virtual machine.

⚙️ 2. What Are cgroups (Control Groups)?

🧠 Definition:

cgroups (short for control groups) limit, prioritize, and account for resource usage (CPU, memory, disk I/O, etc.) among process groups.

cgroups ensure that one container can't starve others or the host of system resources.

📁 Key Features:

Feature	**Example**
CPU limits	Restrict container to 2 CPU cores
Memory limits	Enforce 512MB memory cap
BlkIO control	Limit disk I/O read/write speeds

Feature	Example
PIDs limit	Max number of processes per container
Accounting	Monitor per-container usage statistics

🧪 Docker Example:

```
docker run -m 256m --cpus="1.0" ubuntu
```

- `-m 256m`: Limits memory usage to 256MB
- `--cpus=1.0`: Restricts container to one logical CPU

🛡️ Benefits for Containers

Feature	Enabled By	Benefit
Process isolation	PID namespace	Each container sees only its own processes
Filesystem separation	MNT namespace	Unique root filesystem per container
Network independence	NET namespace	Independent IP stack, ports
Resource fairness	cgroups	Prevent resource hogging
Secure multi-tenancy	user namespace	UID/GID remapping for safety

🧪 Manual Namespace (Example)

```
unshare --pid --mount --uts --ipc --net --user --fork bash
```

This launches a shell with isolated namespaces — similar to a minimal container environment.

🔚 Conclusion

- **Namespaces** isolate containers from each other and the host.
- **Cgroups** control and limit how much a container can use.
- Together, they provide the foundational mechanisms that make containerization secure, efficient, and scalable.