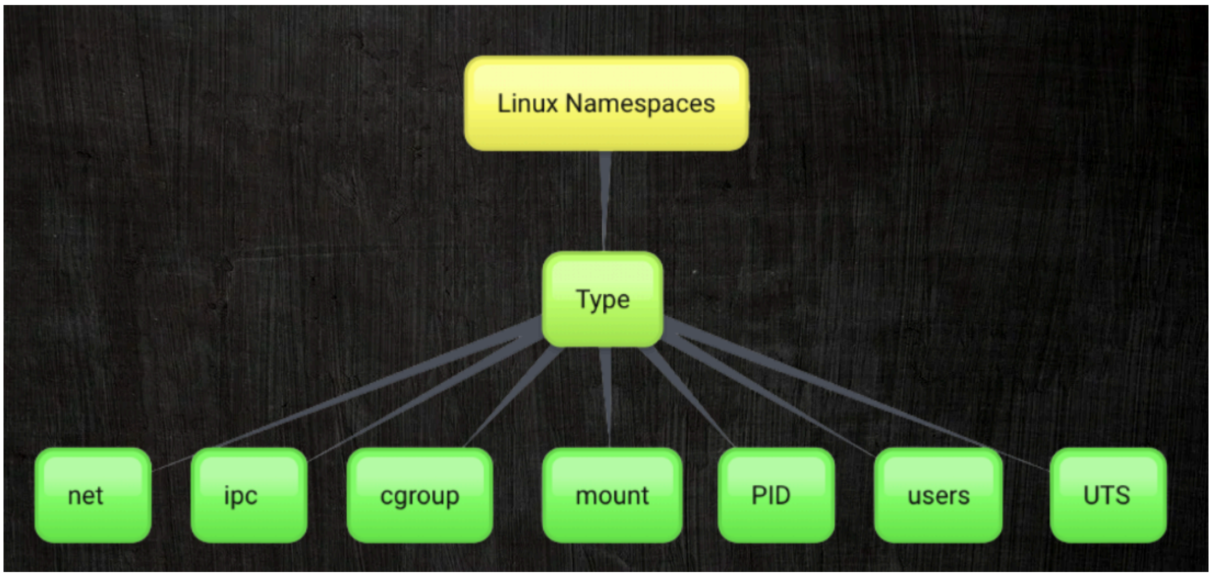
Linux Namespaces

Linux Namespaces – A Deep Dive into Process Isolation

What are Linux Namespaces?

Linux **Namespaces** are a core feature of the Linux kernel that enables process isolation. They allow different processes to have their own separate view of system resources, such as the filesystem, process IDs, network interfaces, and more.

This isolation is the foundation of **containerization technologies like Docker and Kubernetes**, which rely on namespaces to run multiple applications independently on the same host.



Namespaces are a feature of the Linux kernel that partitions kernel resources such that one set of processes sees one set of resources and another set of processes sees a different set of resources. The feature works by having the same namespace for a group of resources and processes, but those namespaces refer to distinct resources.

How Do Namespaces Work?

When a process is assigned to a namespace, it perceives only the resources assigned to that namespace. Other processes outside the namespace remain invisible to it.

For example:

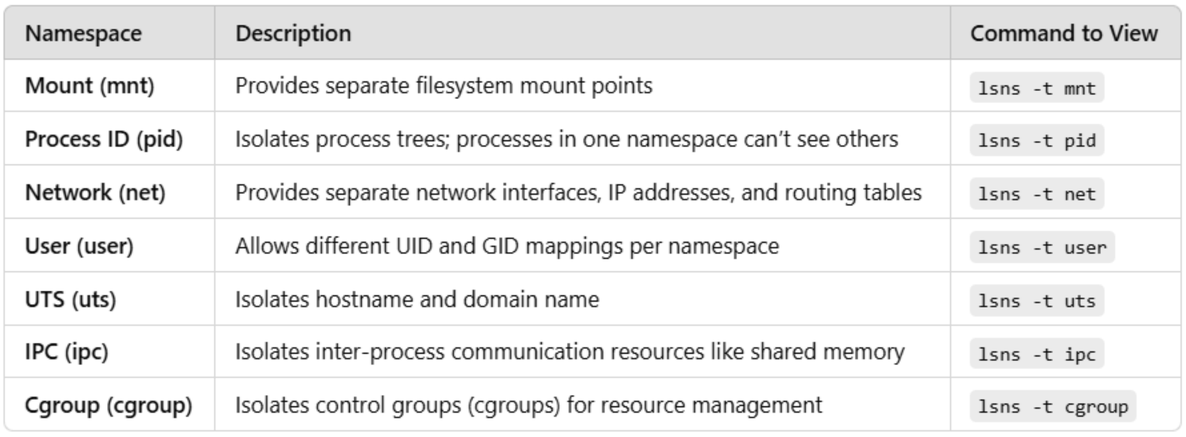
- A process in a **PID namespace** will only see processes inside that namespace, not the global system processes.

- A process in a **Network namespace** will have its own virtual network stack separate from the host system.

Each namespace is like a "mini isolated world" inside the Linux system.

Types of Linux Namespaces

There are **seven types** of namespaces in Linux, each isolating a specific resource:



Namespace	Description	Command to View
Mount (mnt)	Provides separate filesystem mount points	`lsns -t mnt`
Process ID (pid)	Isolates process trees; processes in one namespace can't see others	`lsns -t pid`
Network (net)	Provides separate network interfaces, IP addresses, and routing tables	`lsns -t net`
User (user)	Allows different UID and GID mappings per namespace	`lsns -t user`
UTS (uts)	Isolates hostname and domain name	`lsns -t uts`
IPC (ipc)	Isolates inter-process communication resources like shared memory	`lsns -t ipc`
Cgroup (cgroup)	Isolates control groups (cgroups) for resource management	`lsns -t cgroup`

---

4. Practical Example – Creating a New PID Namespace

Let's create a new **PID namespace** where a process will have an isolated process tree.

1️⃣ **Run the following command to start a new PID namespace:**

sudo unshare --pid --fork bash

2️⃣ **Inside this new shell, check the process tree:**

ps aux

🔹 **You will see only a few processes (mainly the shell itself), even though many processes are running on the host system.**

3️⃣ **Exit the namespace to return to the original environment:**

exit

Now, running `ps aux` will list all system processes again.

---

5. Why Are Namespaces Important?

- **Containerization:** Namespaces are essential for tools like **Docker** and **Kubernetes** to provide process and network isolation.
- **Security:** Helps in **sandboxing applications** to prevent unauthorized access to system resources.
- **Resource Management:** When combined with **cgroups**, namespaces help limit CPU, memory, and other system resources.
- **Multi-Tenancy:** Cloud platforms use namespaces to run multiple applications securely on shared infrastructure.

Conclusion

Linux Namespaces are a powerful feature that enables **lightweight process isolation**. Unlike traditional virtual machines, they do not require a separate OS, making them **faster and more efficient**.

Would you like an example related to **Docker and Kubernetes namespaces**? 🚀