# 4.4 Архітектура та компоненти Kubernetes

**Kubernetes кластер** — об’єднання декількох елементів, у цьому разі серверів, чи фізичні або віртуальні обчислювальні ресурси, що являє собою самостійну одиницю. За термінологію Kubernetes, **ноди (node)** — це сервери. У минулому їх ще називали міньйони. Але сьогодні це точно ноди. Тобто, вузли кластера.

Ноди за своєю роллю можуть бути **майстром** або **воркером**. Тобто, керуючі ноди — контролери, на яких запущені сервіси, що здійснюють контроль і управління всієї системи. І робочі сервери — за своєю суттю обчислювальні ресурси, на яких виконується основна робота. Тобто запускаються контейнери.

Запущені контейнери використовують саме ресурси воркерів. Процесор, пам'ять, дисковий простір, спеціальні пристрої GPU, TPU, гетерогенні середовища виконання, що об'єднують пристрої з різними операційними системами або протоколами передачі даних. Так само і різні платформи та архітектури — це все ми відносимо до воркерів. Але хочу зауважити, Kubernetes може бути запущений на одній ноді і навіть одним процесом і поєднувати ролі мастера і воркер ноди в межах однієї машини.

**Pod** — мінімальна одиниця управління в Kubernetes наділена унікальними властивостями. Якщо коротко — це набір із контейнерів і файлової системи, що працюють в одному мережевому середовищі.

Поди в кластері характеризуються наступним:

- Усі контейнери в Pod завжди розміщуються на одній ноді;

- Кожен контейнер у Pod працює у своїй власній контрольній групі, але вони розділяють один неймспейс;
- Процеси, запущені в одному Pod, використовують одну й ту саму IP-адресу і пул портів;
- Файловий простір доступний і розділяється між усіма процесами в поді.

Для того щоб зрозуміти, як саме розмістити ваш контейнер в одному поді чи в різних, можна скористатися контрольним питанням: «Чи будуть ці контейнери працювати правильно, якщо вони розмістяться на різних машинах?» Якщо відповідь на це запитання «Ні», то один Pod є правильним вибором для контейнерів. Якщо ж відповідь «Так, вони будуть працювати» — то кілька Pod, імовірно, правильне рішення.

**Kubernetes деплоймент** — це ресурс ще має назву **pod blueprint** — проєкт поду. Об’єкт, у якому ми описуємо специфікацію пода. Деплоймент за цією специфікацією створить репліка сет, об'єкт, що керує кінцевим набором подів.

Усе що ми запускаємо в Kubernetes кластері можна розділити на такі категорії:

- long term running сервіси;
- задачі, або джоби (jobs);
- stateless процеси без збереження стану — наприклад, вебсервер;
- statefull із збереженням стану — наприклад, бази даних.

Усе перераховане вище Deployment, ReplicaSet, DaemonSet, StatefullSet, Jobs тощо — це все верхньорівневі API-об'єкти Kubernetes, що контролюють групу або один Pod.

Сполучною ланкою, елементом, що дає змогу всій цій Kubernetes машинерії ожити і стати доступною для користувачів, виступає Kubernetes Service.

**Kubernetes Service** — це абстракція над мережевою підсистемою Linux. Утворює комунікацію усередині кластера — це потік трафіку, що називається схід-захід, і комунікацію із зовнішнім світом — північ-південь. У вигляді схеми Kubernetes Service можна розглянути як ієрархію від зовнішнього лоад балансера, зарезервованих під кожний сервіс портів на Kubernetes нодах, портмаппінг та маршрутизація на IP-адреси кожного конкретного контейнера.

Kubernetes надає об'єкти і ресурси під кожне конкретне завдання, наприклад:

- TLS сертифікати або паролі — Secrets.
- Конфігураційні файли — ConfigMap.
- Диски — PersistantVolume
- CRD (Custom Resource Definition) — кастомно визначені ресурси, нові об'єкти, що обслуговуються Kubernetes згідно специфікації API.

Управління кластером здійснюється за допомогою:

- **Kubectl** — cli утиліта управління кластером;
- **Маніфести** — файли специфікації у yaml або json форматі;
- **Helm** — пакетний менеджер HELM, що об'єднує маніфести в пакети і реалізує валідацію та автоматизацію;
- **Kubernetes оператор** — сервісний процес усередині кластера Kubernetes. Взаємодіючи з API і циклом подій (eventloop), Kubernetes оператор створює, конфігурує, реагує на події та контролює стан об'єктів та ресурсів аплікейшена, за який відповідає.

Kubernetes — це чотири основні компоненти:

- API-сервер;
- Планувальник schedelure;
- Менеджер контролерів;
- Etcd база даних.

Усі вони покликані керувати, визначати і відповідати на події, що виникли у кластері. Прийнято виділяти окремі ноди під **Control Plane компоненти** і називати їх мастерами. Хоча базу даних можна, наприклад, розмістити на окремих зовнішніх серверах, а інші компоненти запустити на будь-якій машині в кластера. У разі сингл нод кластера — процеси Control Plane і контейнери запускаються на єдиній ноді.

На кожній ноді кластера працюють три обов'язкові компоненти, що утворюють **робоче Kubernetes оточення**:

- агент API-сервера — kubelet,
- мережевий проксі — kube-proxy, що імплементує концепцію Kubernetes нетворкінга, і
- контейнер runtime — програмне забезпечення, відповідальне за запуск контейнерів.

Також існують обов'язкові та не обов'язкові додаткові компоненти. Наприклад, **DNS** — система, що реалізує реєстрацію всіх сервісів у кластері та сервіс дискавері функцію. Додатково ще може бути моніторинг ресурсів і логування, а також лоадбалансер. Розглянемо більш детально чотири основні компоненти Control Plane.

Взаємодія з Kubernetes здійснюється через **REST API** інтерфейс, ключовою функціональністю API-сервера Kubernetes. Термінальні утиліти, кастомні сервіси, що використовують клієнтські бібліотеки, по суті використовують REST API для маніпулювання ресурсами кластера. Міжкомпонентна комунікація здійснюється також за

допомогою **HTTP API**, що базується на швидкому і надійному фреймворку gRPC.

В Kubernetes є можливість використанням сховища даних, відмінного від etcd. Наприклад, ви можете обрати **базу даних SQL** корпоративного рівня, наприклад MySQL або PostgreSQL. Якщо вам потрібно запустити простий кластер у середовищі CI/CD, ви можете використовувати вбудовану базу даних SQLite. При розгортанні в режимі HA практикується винесення бази даних за периметр кластера на окремі сервери.

Будь-який із компонентів мастера може бути розміщений на будь-якій ноді в кластері. Майстер у принципі створюється як звичайна воркернода. Але прийнято виносити Control Plane на окремі сервери в окремий неймспейс  для зручності адміністрування. Один CPU і один гігабайт оперативної пам'яті може забезпечити кластер у 50 нод.

На воркернодах основним компонентом є **Kubelet**. Агент, який запускається на кожному вузлі кластера. Він відповідає за **Pod lifecycle**. Повний цикл створення контейнера в поді згідно з специфікацією пода. Скачування іміджу, виділення мережевих адрес, cgroups, namespaces, монтування сторaджів. Доречі, завдяки спеціальним механізмам, kubelet в принципі стартує першим під час розгортання кластера Kubernetes.

**Kube-Proxy** — за визначенням це мережевий проксі, який працює на кожному вузлі, реалізуючи концепцію Kubernetes нетворкінг. Kube-proxy підтримує мережеві політики на вузлах. Ці мережеві правила забезпечують взаємодію з Pods як усередині, так і поза вашим кластером.

Один із найважливіших компонентів для контейнерів — **Container runtime**. Це програмне забезпечення, яке відповідає за запуск

контейнерів. Kubernetes підтримує такі рантайми як containerd, CRI-O і будь-які інші реалізації Kubernetes CRI (Container Runtime Interface).