# Infrastructure Comparison Tables

Martina Castellucci

## Comparison Tables

### Docker vs Docker Compose vs Udocker

| **Feature** | **Docker** | **Docker Compose** | **Udocker** |
|---|---|---|---|
| Root Access | Required for daemon | Depends on setup | Not required; runs in user space |
| Use Case | Running individual containers | Orchestrating multi-container apps | Running containers in HPC without root |
| Security | Host-level access required | Shares Docker context | User-level, no root access |
| Deployment | CLI or GUI | Uses YAML config files | CLI compatible with Docker images |

### Cloud Models

| **Model** | **Description and Responsibility** |
|---|---|
| IaaS (Infrastructure as a Service) | Provides virtualized computing resources over the internet. The user manages the operating system, storage, deployed applications, and runtime. Examples include AWS EC2, Google Compute Engine, Microsoft Azure VM. |
| PaaS (Platform as a Service) | Provides a platform allowing customers to develop, run, and manage applications without dealing with infrastructure. The provider handles OS, middleware, and runtime. Examples: Heroku, Google App Engine. |
| SaaS (Software as a Service) | Offers ready-to-use software applications over the web. The provider manages everything from infrastructure to software. Users only interact with the interface. Examples: Gmail, Dropbox, Microsoft 365. |

## Cooling Techniques

| Method | Description |
|---|---|
| Air Cooling | Uses fans and heat sinks to remove heat from components. It is the most economical method, ideal for general-purpose systems, but becomes inefficient in high-density setups. |
| Liquid Cooling | Employs a coolant fluid circulated via tubes and blocks to draw heat away from processors and components. Offers superior thermal performance and quieter operation. |
| Immersion Cooling | Submerges entire components or servers in a dielectric fluid. It ensures uniform cooling, reduces noise, and is used in high-performance, energy-efficient data centers. |

## File Systems

| File System | Usage and Characteristics |
|---|---|
| ext4 | Widely used local file system on Linux. Supports journaling, fast performance, good for general-purpose servers. |
| xfs | High-performance journaling FS for parallel I/O. Efficient for large files and used in enterprise Linux systems. |
| NFS | Network File System. Enables sharing files over network. Suitable for shared storage in clusters. |
| GPFS (IBM Spectrum Scale) | Parallel file system for HPC. Provides scalability and high-throughput for large workloads. |
| Lustre | Designed for large-scale parallel file systems in supercomputers and clusters. Open source and scalable. |

## Kubernetes vs Docker

| Aspect | Docker | Kubernetes |
|---|---|---|
| Function | Tool to build, run, and manage containers on a single host | Orchestration system to deploy, scale, and manage containers across multiple hosts |
| Scalability | Limited to a single host (or Docker Swarm) | Designed for scaling across many machines and fault tolerance |
| Networking | Provides simple bridge and host networking | Built-in service discovery and load balancing via Services |
| Use Case | Local development, small deployments | Production-grade orchestration of microservices at scale |

## Cloud vs Virtualization

| Aspect | Description |
|---|---|
| Virtualization | Virtualization allows multiple operating systems to run on the same physical hardware by abstracting the hardware using a hypervisor (e.g., KVM, VMware ESXi). It forms the basis for cloud computing but can be used independently in on-prem setups. |
| Cloud Computing | A service delivery model that provides access to computing resources (VMs, storage, databases, etc.) over the internet. Built on top of virtualization, it adds elasticity, scalability, APIs, billing, and managed services. |

## Recall vs Backfill vs Reservation

| **Term** | **Meaning** |
|---|---|
| Recall | The process of reclaiming allocated but unused or idle resources (e.g., from preemptible or low-priority jobs) so they can be reassigned. Useful in opportunistic environments. |
| Backfill | A scheduling strategy where smaller jobs are temporarily scheduled in idle gaps between larger reservations, improving overall resource utilization without delaying high-priority jobs. |
| Reservation | Explicit allocation of compute resources for a specific user, group, or job at a future time. Ensures resource availability for time-sensitive workloads. |

## OSI vs TCP/IP Models

| **Layer** | **OSI Model** | **TCP/IP Model** |
|---|---|---|
| Layer 7 | Application | Application |
| Layer 6 | Presentation | – (merged into Application) |
| Layer 5 | Session | – (merged into Application) |
| Layer 4 | Transport | Transport |
| Layer 3 | Network | Internet |
| Layer 2 | Data Link | Network Access |
| Layer 1 | Physical | Network Access |

## TOR (The Onion Router) Architecture

| **Layer** | **Function** |
|---|---|
| Entry Node | First relay in the TOR network. It knows the user's IP address and establishes an encrypted path to the next node. |
| Relay Node | Intermediate node that passes traffic in encrypted form. It does not know the source or final destination. |
| Exit Node | Final node in the circuit. It decrypts the last layer and forwards the request to the destination server. It sees the destination but not the source. |

## HTCondor vs SLURM

| **Feature** | **HTCondor** | **SLURM (HPC Scheduler)** |
|---|---|---|
| Scheduling Model | Opportunistic scheduling with late-binding | Resource-reservation with partitioned scheduling |
| Target Use | High Throughput Computing (HTC), grid environments | High Performance Computing (HPC), tightly coupled tasks |
| Fault Tolerance | Supports checkpointing and job migration | Limited fault tolerance; job restarts on failure |
| Resource Match | Uses ClassAds to match jobs with resources | Uses queues and partitions, less dynamic |
| Scalability | Scales across heterogeneous systems in a grid | Scales efficiently in tightly integrated clusters |

## INFN CNAF vs Other Sites

| **Location** | **Description** |
|---|---|
| INFN CNAF | National computing center (Tier-1) in Bologna. Offers large-scale storage, computing, and data services for LHC and Italian research. Connected to GRID, provides access to HPC and HTC resources. |
| Other Sites | Regional centers (Tier-2 or Tier-3). Focused on localized support, smaller infrastructure, often used for preprocessing or analysis tied to larger collaborations. |

## Amdahl's Law: Speedup vs Efficiency

| **Metric** | **Formula and Explanation** |
|---|---|
| Speedup (S) | $S = \frac{1}{(1-P)+\frac{P}{N}}$ — quantifies how much faster a program runs when parallelized, where $P$ is the parallelizable portion and $N$ the number of processors. |
| Efficiency (E) | $E = \frac{S}{N}$ — evaluates how effectively the parallel resources are being used; close to 1 means optimal utilization. |

## Power vs Storage

| Aspect | Metric / Description |
|---|---|
| Power | Measured in Watts. TDP (Thermal Design Power) indicates the max heat a component generates; influences cooling requirements. |
| Storage | Measured in GB/TB. Performance determined by latency, throughput, and IOPS (Input/Output Operations Per Second). |

## Node vs Job

| Entity | Description |
|---|---|
| Node | A compute resource — physical or virtual — in a cluster that executes assigned jobs. Often includes CPUs, RAM, and access to shared storage. |
| Job | A user-defined task or process submitted to a scheduler for execution on one or more nodes. Can be serial, parallel, or array-based. |

## Server vs Data Center vs Distributed

| System Type | Scope and Characteristics |
|---|---|
| Server | A single machine offering computing or storage services. Can be physical or virtual and may serve multiple users or applications. |
| Data Center | A facility that houses many servers with supporting infrastructure (cooling, power, security). Centralized and managed IT operations. |
| Distributed Computing | A computing model in which components located on networked computers communicate and coordinate to achieve a common goal. Supports scalability and fault tolerance. |

## CPU vs GPU

| **Feature** | **CPU (Central Processing Unit)** | **GPU (Graphics Processing Unit)** |
|---|---|---|
| Purpose | General-purpose computing | Parallel processing, mainly for graphics and large-scale computations |
| Cores | Fewer (2–64), optimized for sequential tasks | Hundreds to thousands of cores, optimized for parallel tasks |
| Latency | Low latency per task | Higher latency per task |
| Throughput | Lower overall throughput | Very high throughput |
| Use Case | OS management, general logic, low-latency tasks | Deep learning, image rendering, matrix operations |

## Single-core vs Multi-core CPU

| **Aspect** | **Single-core CPU** | **Multi-core CPU** |
|---|---|---|
| Definition | One execution core per CPU | Multiple cores in a single CPU chip |
| Parallelism | Executes one thread at a time | Executes multiple threads simultaneously |
| Performance | Slower for multitasking | Better multitasking and parallel computing |
| Power Consumption | Typically lower | More power, but better performance/watt |
| Use Cases | Embedded devices, old systems | Modern desktops, servers, smartphones |

## The 5 Vs of Big Data

| **V** | **Description** |
|---|---|
| Volume | Refers to the huge amounts of data generated every second (e.g., terabytes, petabytes) |
| Velocity | Speed at which data is generated, processed, and analyzed |
| Variety | Different types of data: structured, semi-structured, unstructured |
| Veracity | Quality and trustworthiness of data; presence of noise and uncertainty |
| Value | The usefulness of the data once processed and analyzed |

## DAS vs NAS vs SAN

| Type | Description |
|---|---|
| DAS (Direct Attached Storage) | Storage attached directly to a server or workstation without a network |
| NAS (Network Attached Storage) | File-level storage shared over a network; uses protocols like NFS, SMB |
| SAN (Storage Area Network) | Block-level storage accessed over a high-speed network, typically via Fibre Channel or iSCSI |

## Edge vs Fog vs Cloud Computing

| Model | Description |
|---|---|
| Edge Computing | Processing done near the data source (e.g., sensors); reduces latency and bandwidth |
| Fog Computing | Intermediate layer between edge and cloud; distributes computing/storage closer to edge |
| Cloud Computing | Centralized processing in data centers; offers scalability, redundancy, and global access |

## AWS and IaaS Example

| Service | Description |
|---|---|
| AWS EC2 | Elastic Compute Cloud; provides resizable compute capacity in the cloud (IaaS) |
| AWS S3 | Simple Storage Service; object storage with web interface, often used with EC2 |
| AWS VPC | Virtual Private Cloud; isolated network for AWS resources |
| Relation to IaaS | AWS provides IaaS by allowing users to manage VMs, storage, and networking infrastructure |

## Batch Computing vs Cloud Computing

| Feature | Batch Computing | Cloud Computing |
|---|---|---|
| Job Execution | Scheduled in queues, managed by local scheduler | On-demand job execution via cloud APIs |
| Resource Allocation | Static, predefined | Elastic and scalable |
| Cost Model | Fixed, shared among users | Pay-per-use (metered billing) |
| Infrastructure | On-premise or academic clusters | Hosted and managed by provider (e.g., AWS, Azure) |

## Batch Computing vs Grid Computing

| Aspect | Batch Computing | Grid Computing |
|---|---|---|
| Scope | Typically within a single organization or cluster | Federated, across multiple administrative domains |
| Resource Sharing | Centralized, static | Decentralized, dynamic sharing |
| Middleware | Simple batch schedulers (e.g., SLURM, PBS) | Grid middleware (e.g., gLite, Globus, ARC) |
| Security | Based on local users | Often uses certificates and federated identities |

## Scheduler vs Cloud Provider

| Aspect | Scheduler (e.g., SLURM, HTCondor) | Cloud Provider (e.g., AWS, Azure) |
|---|---|---|
| Responsibility | Manages job queueing, prioritization, and execution on a cluster | Provides infrastructure and services for compute, storage, and networking |
| Control | Full control over resource management | Limited to APIs and services offered |
| Location | Local clusters or HPC/HTC systems | Remote, virtualized data centers |
| Scalability | Bound to physical resources | Virtually unlimited scaling |

## Dockerfile vs Docker Image

| Concept | Dockerfile | Docker Image |
|---|---|---|
| Definition | Script with instructions to build an image | Executable snapshot of a container environment |
| Use | Written by developer to define build steps | Created after Dockerfile is built; used to run containers |
| Example<br>`COPY . /app` | `FROM python:3.8`<br>`myimage:v1` (result of `docker build`) | |

## Digital Twins

| Term | Description |
|---|---|
| Digital Twin | A digital representation of a real-world object or system that is updated with real-time data. Used for monitoring, simulation, and optimization (e.g., industrial IoT, smart cities). |

## Cloud-Aware vs Cloud-Native Efficiency

| Aspect | Cloud-Aware | Cloud-Native |
|---|---|---|
| Definition | Applications adapted to run in cloud | Applications designed for cloud from the ground up |
| Architecture | Monolithic or partially containerized | Microservices, containers, orchestration (Kubernetes) |
| Scaling | Manual or limited automation | Automatic, elastic scaling |
| Efficiency | Limited by legacy architecture | Fully optimized for distributed environments |

## IP vs MAC Address + Common IPs

| Term | Description |
|---|---|
| IP Address | Logical address used for routing on network layer (e.g., 192.168.0.1) |
| MAC Address | Physical address tied to network interface; layer 2 identifier (e.g., 00:1A:2B:3C:4D:5E) |
| 127.0.0.1 | Loopback address (localhost); used for internal testing |
| 0.0.0.0 | Non-routable address; binds to all available interfaces |

## Checksums

| Term | Description |
|---|---|
| Checksum | A value derived from data to detect errors or integrity loss. Common algorithms: MD5, SHA-1, SHA-256. |
| Use Cases | File verification, network transmission, package downloads |

## BLASTn vs BWA

| Tool | BLASTn | BWA |
|---|---|---|
| Purpose | Nucleotide sequence similarity search | Read alignment to reference genome |
| Algorithm | Seed-and-extend; local alignment | Burrows-Wheeler Transform; global or semi-global |
| Speed | Slower, especially with large datasets | Very fast, optimized for NGS data |
| Use Case | Annotating genes, database comparison | DNA-seq data preprocessing, variant calling |

## HTT (High Throughput Technologies)

| Term | Description |
|---|---|
| High Throughput Technologies | Techniques that allow simultaneous processing of a large number of samples or data points (e.g., microarrays, NGS, mass spectrometry). Key in omics and screening platforms. |

## WMS (Workflow Management Systems)

| System | Description |
|---|---|
| WMS | Tools that help define, manage, and execute data pipelines (e.g., Snakemake, Nextflow, Galaxy). Support reproducibility, scalability, and parallelism in bioinformatics and big data workflows. |

## Vocabulary: Key Infrastructure Terms

| **Term** | **Definition** |
|---|---|
| Switch | Network device that connects devices within a LAN and forwards data based on MAC addresses. |
| Router | Device that connects multiple networks and routes packets based on IP addresses. Often links LANs to WANs. |
| Hub | Basic network device that broadcasts data to all connected devices; no intelligence like a switch. |
| LAN (Local Area Network) | Network covering a small geographic area (e.g., office, home), typically high-speed. |
| WAN (Wide Area Network) | Network covering a large area, such as the internet or inter-office networks. |
| Hardware | The physical components of a computer or infrastructure (CPU, memory, disk, etc.). |
| Software | Programs and operating systems that run on hardware and control tasks. |
| Kernel | Core of the operating system that manages system resources and communication between hardware and software. |
| Host | A device (physical or virtual) that provides services and has an operating system installed. |
| OS (Operating System) | System software that manages computer hardware, software resources, and provides services for applications. |
| HDD (Hard Disk Drive) | Traditional spinning magnetic storage device, high capacity, slower access. |
| SSD (Solid State Drive) | Fast, non-volatile flash memory storage device with no moving parts. |
| NFS (Network File System) | Protocol that allows file access over a network as if local; used in shared UNIX/Linux environments. |
| NUMA (Non-Uniform Memory Access) | Memory architecture where access time depends on the memory's location relative to the processor. |
| UMA (Uniform Memory Access) | Architecture where all processors share physical memory uniformly. |
| L1 Cache | The smallest and fastest memory level, located closest to the CPU core; typically stores instructions and data. |
| L2 Cache | Larger than L1, slower but still fast; stores data used less frequently. Often shared among cores. |
| L3 Cache | Shared among all cores in a processor; bigger but slower than L2, reduces memory latency at higher levels. |
| RAID (Redundant Array of Independent Disks) | A method of storing data on multiple hard disks for redundancy and performance, using various levels (RAID 0, 1, 5, etc.) |

## Vocabulary: Key Infrastructure Terms 2)

| | |
|---|---|
| Tiered Storage | Storage architecture that uses multiple storage media (e.g., SSD, HDD, tape) ranked by speed/cost, and data is moved accordingly. |
| Hyperthreading | Intel technology that allows one physical CPU core to execute two threads simultaneously to improve parallelism. |
| Throughput | The amount of data processed in a given time, often measured in bits per second or IOPS in storage. |
| Grid Computing | Distributed computing model where resources across multiple locations are federated and used collectively. |
| Cloud Computing | Delivery of computing services (compute, storage, networking) over the internet on demand. |
| Edge Computing | Processing data close to where it is generated (e.g., IoT sensors) to reduce latency. |
| Fog Computing | Extends cloud to be closer to edge devices, offering computation, storage, and networking. |
| Stream Processing | Real-time processing of continuous data streams (e.g., Kafka, Flink). |
| ALU (Arithmetic Logic Unit) | Part of the CPU that performs arithmetic and logical operations. |
| Control Unit | CPU component that directs operations of the processor; it fetches, decodes, and executes instructions. |
| I/O (Input/Output) | Interface between the computer and external devices; includes peripherals and network input/output. |
| Container | Lightweight, portable, and self-sufficient unit that includes everything needed to run a piece of software. |
| Docker | Popular container platform that automates deployment of applications inside containers. |

## Common Command Reference Table

| Command | Function / Description |
|---|---|
| docker pull <image> | Download an image from a Docker registry (e.g., Docker Hub) |
| docker push <image> | Upload an image to a Docker registry |
| docker run <image> | Create and start a container from an image |
| docker create <image> | Create a container without starting it |
| docker build -t <name> . | Build an image from a Dockerfile in the current directory |
| docker volume create <name> | Create a named volume (persistent storage for containers) |
| docker-compose up | Start all services defined in a docker-compose.yml file |
| docker-compose down | Stop and remove services, networks, volumes defined in the compose file |
| udocker run <container> | Execute a container without requiring root privileges (HPC environments) |
| condor_submit <file> | Submit a job to HTCondor using the specified job description file |
| condor_q | View the status of submitted jobs in the queue |
| condor_status | Show the status of machines in the Condor pool |
| cat /proc/cpuinfo | Display detailed CPU architecture and core information |
| free -m | Show memory usage in megabytes |
| df -h | Show available disk space on mounted file systems (human readable) |
| mount | List all currently mounted filesystems |
| lsblk | Show all block devices and partitions |
| md5sum <file> | Generate or verify MD5 checksum for a file |
| time <command> | Measure the time taken to execute a command (e.g., time blastn) |
| blastn -query file -db nt -out res | Run a nucleotide BLAST search |