EP95: Linux File System Explained

BYTEBYTEGO
JAN 20, 2024

This week's system design refresher:

- Linux File System Explained (Youtube video)
- REST API Vs. GraphQL
- Key Use Cases for Load Balancers
- Types of memory. Which ones do you know?
- Top 6 Firewall Use Cases
- SPONSOR US

ScyllaDB Summit | Feb 14-15 | Free + Virtual (Sponsored)

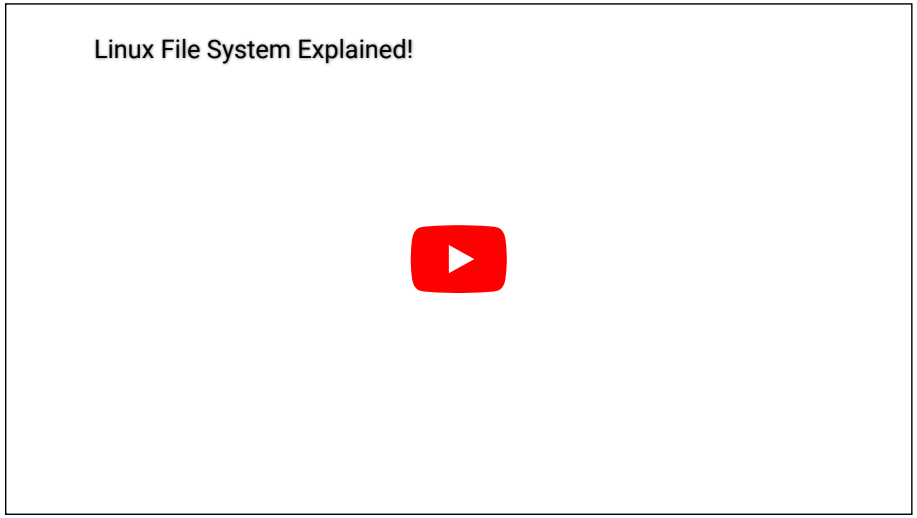
Join thousands of your peers for an unprecedented opportunity to learn from experts at Discord, Disney, Expedia, Zee, ShareChat, Paramount & more – for free, from anywhere.

30+ talks on NoSQL, Kafka, Kubernetes, and AI/ML system architectures and best practices, plus interactive hands-on training.

Discord Keynote: Everything's great until the database is down. Bo Ingram, staff engineer at Discord and author of *ScyllaDB in Action*, shares lessons from their database outage so that you can learn from their mistakes.

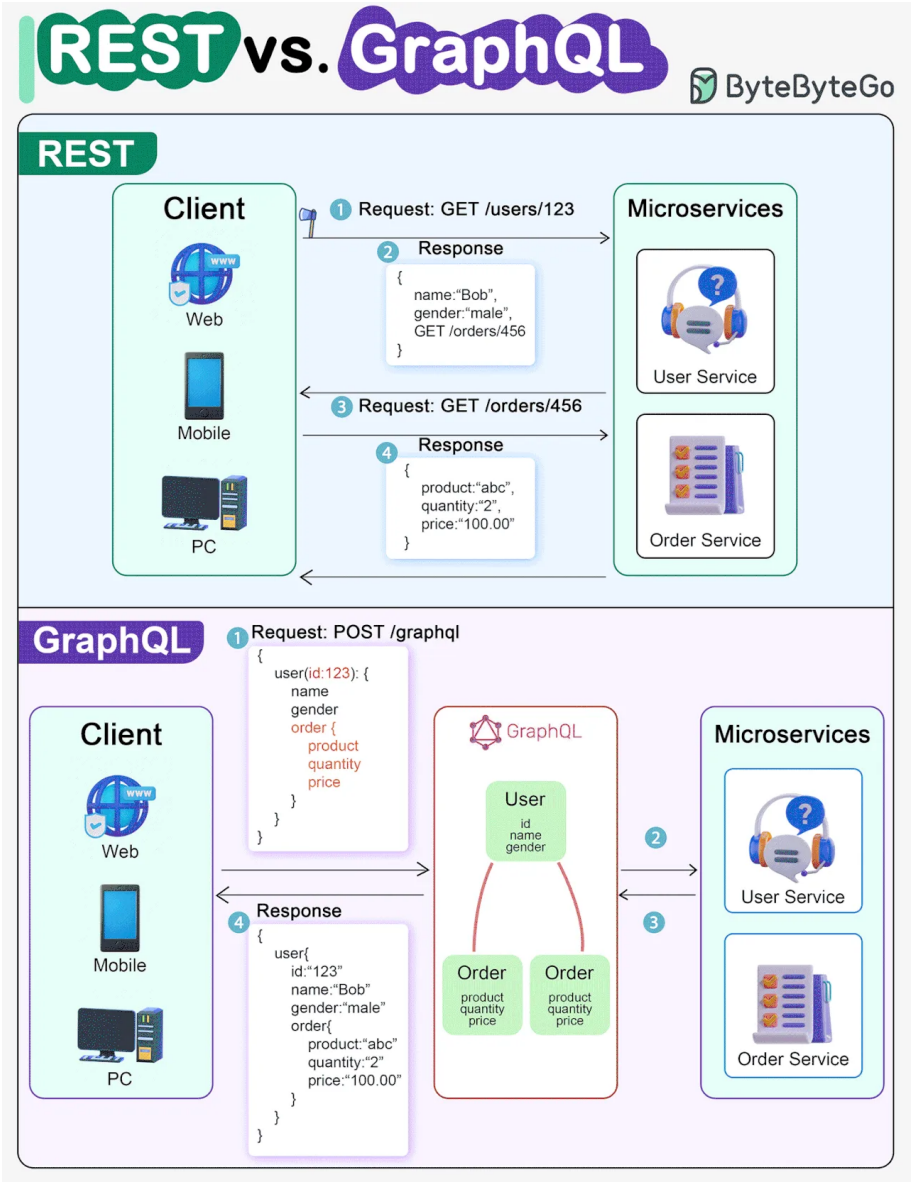
Bonus: Register this week to win one of **300 free SWAG PACKS**.

Linux File System Explained



REST API Vs. GraphQL

When it comes to API design, REST and GraphQL each have their own strengths and weaknesses.



REST

- Uses standard HTTP methods like GET, POST, PUT, DELETE for CRUD operations.
- Works well when you need simple, uniform interfaces between separate services/applications.
- Caching strategies are straightforward to implement.
- The downside is it may require multiple roundtrips to assemble related data from separate endpoints.

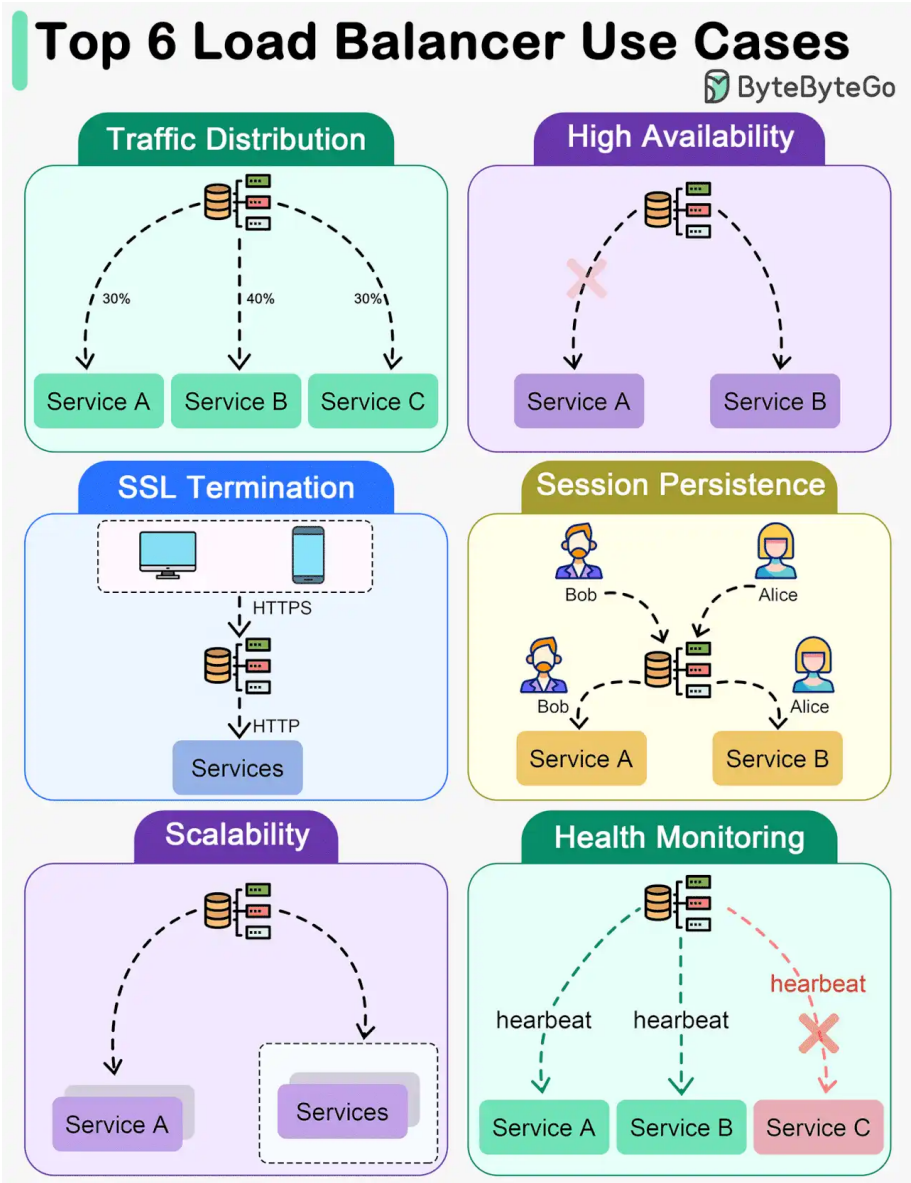
GraphQL

- Provides a single endpoint for clients to query for precisely the data they need.
- Clients specify the exact fields required in nested queries, and the server returns optimized payloads containing just those fields.
- Supports Mutations for modifying data and Subscriptions for real-time notifications.
- Great for aggregating data from multiple sources and works well with rapidly evolving frontend requirements.
- However, it shifts complexity to the client side and can allow abusive queries if not properly safeguarded
- Caching strategies can be more complicated than REST.

The best choice between REST and GraphQL depends on the specific requirements of the application and development team. GraphQL is a good fit for complex or frequently changing frontend needs, while REST suits applications where simple and consistent contracts are preferred.

Key Use Cases for Load Balancers

The diagram below shows top 6 use cases where we use a load balancer.

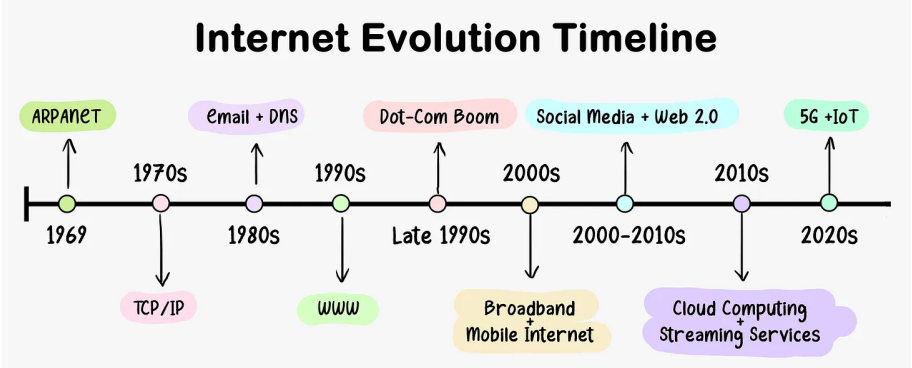


- Traffic Distribution
  Load balancers evenly distribute incoming traffic among multiple servers, preventing any single server from becoming overwhelmed. This helps maintain optimal performance, scalability, and reliability of applications or websites.

- High Availability
  Load balancers enhance system availability by rerouting traffic away from failed or unhealthy servers to healthy ones. This ensures uninterrupted service even if certain servers experience issues.
- SSL Termination
  Load balancers can offload SSL/TLS encryption and decryption tasks from backend servers, reducing their workload and improving overall performance.
- Session Persistence
  For applications that require maintaining a user's session on a specific server, load balancers can ensure that subsequent requests from a user are sent to the same server.
- Scalability
  Load balancers facilitate horizontal scaling by effectively managing increased traffic. Additional servers can be easily added to the pool, and the load balancer will distribute traffic across all servers.
- Health Monitoring
  Load balancers continuously monitor the health and performance of servers, removing failed or unhealthy servers from the pool to maintain optimal performance.

Latest articles

If you're not a paid subscriber, here's what you missed this month.

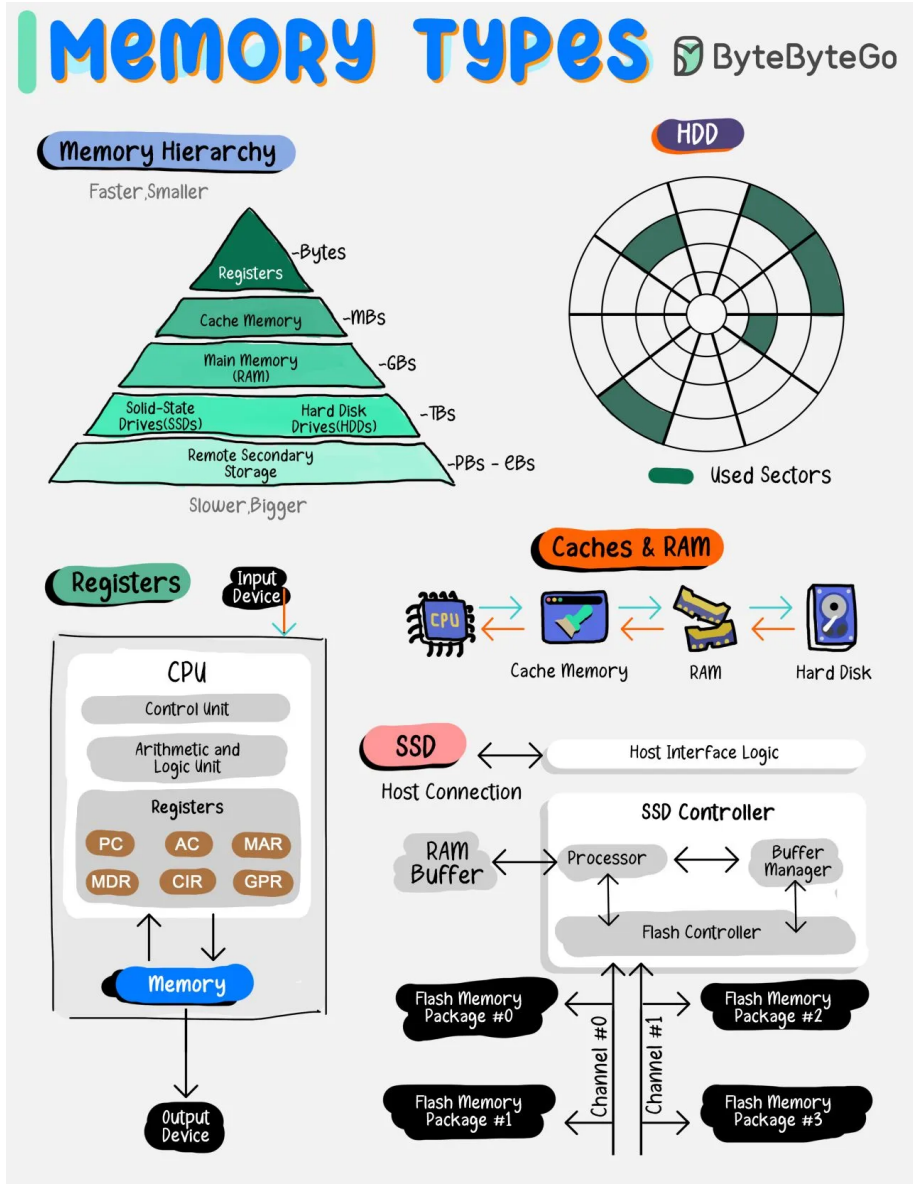


1. A Crash Course in Networking
2. Netflix: What Happens When You Press Play - Part 2
3. Netflix: What Happens When You Press Play?
4. 6 More Microservices Interview Questions
5. 7 Microservices Interview Questions

To receive all the full articles and support ByteByteGo, consider subscribing:

Types of memory. Which ones do you know?

Memory types vary by speed, size, and function, creating a multi-layered architecture that balances cost with the need for rapid data access.



By grasping the roles and capabilities of each memory type, developers and system architects can design systems that effectively leverage the strengths of each storage layer, leading to improved overall system performance and user experience.

Some of the common Memory types are:

1. Registers:
Tiny, ultra-fast storage within the CPU for immediate data access.

2. Caches:
Small, quick memory located close to the CPU to speed up data retrieval.

3. Main Memory (RAM):
Larger, primary storage for currently executing programs and data.

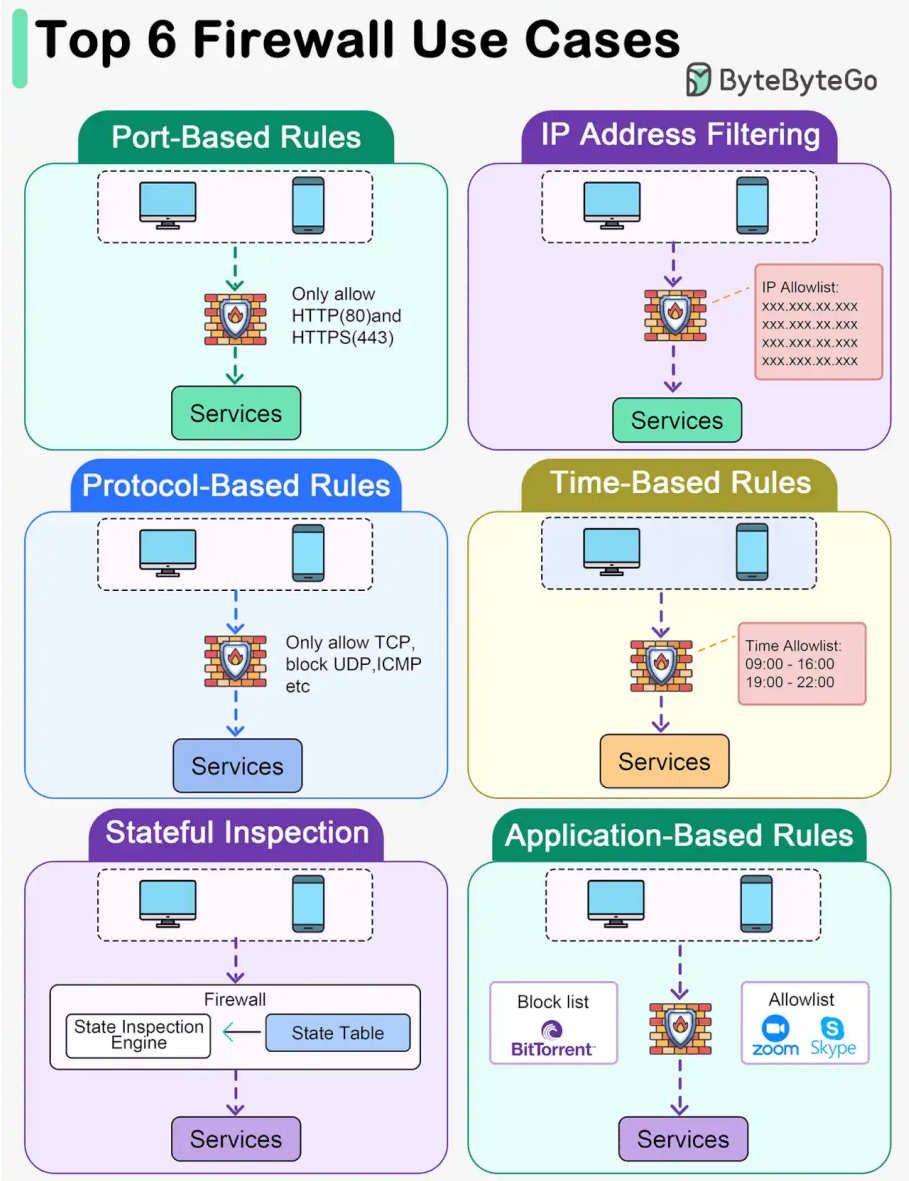
4. Solid-State Drives (SSDs):
Fast, reliable storage with no moving parts, used for persistent data.

5. Hard Disk Drives (HDDs):
Mechanical drives with large capacities for long-term storage.

6. Remote Secondary Storage:
Offsite storage for data backup and archiving, accessible over a network.

Over to you: Which memory type resonates most with your tech projects and why?
Share your thoughts!

Top 6 Firewall Use Cases



- Port-Based Rules
  Firewall rules can be set to allow or block traffic based on specific ports. For example, allowing only traffic on ports 80 (HTTP) and 443 (HTTPS) for web browsing.

- IP Address Filtering
  Rules can be configured to allow or deny traffic based on source or destination IP addresses. This can include whitelisting trusted IP addresses or blacklisting known malicious ones.
- Protocol-Based Rules
  Firewalls can be configured to allow or block traffic based on specific network protocols such as TCP, UDP, ICMP, etc. For instance, allowing only TCP traffic on port 22 (SSH).
- Time-Based Rules
  Firewalls can be configured to enforce rules based on specific times or schedules. This can be useful for setting different access rules during business hours versus after-hours.
- Stateful Inspection
  Stateful Inspection: Stateful firewalls monitor the state of active connections and allow traffic only if it matches an established connection, preventing unauthorized access from the outside.
- Application-Based Rules
  Some firewalls offer application-level control by allowing or blocking traffic based on specific applications or services. For instance, allowing or restricting access to certain applications like Skype, BitTorrent, etc.

509 Likes · 11 Restacks

5 Comments

Write a comment...

Name Jan 22

I love your diagrams? Do you use specific tool or?

LIKE (2) REPLY SHARE

Dank Tec The Dank Tec Newsletter Jan 24

Awesome information, well condensed and easy to understand.

I think the firewall use-cases are more like solution-candidates for a particular use-case. The use-case might be "blocking a certain traffic pattern" and the solution might be "application layer firewall"

Another use-case might be "protecting services from port-scanners to improve security posture" - solution would be "blocking all ports other than TCP 80/443"

LIKE (1) REPLY SHARE

3 more comments...