What is Hyrum’s Law?

Hyrum’s Law (also called the “Law of Leaky Abstractions”) is a principle in software engineering that says:

“With a sufficient number of users of an API, it does not matter what you promise in the contract:

“all observable behaviors of your system will be depended on by somebody.”

This means:

* Even if you document only certain behaviors of your API, users will start relying on unintended, undocumented, or accidental behavior.

Changing these “hidden” behaviors later can break users’ code.

Google engineers shared this principle based on their own experience at scale.

For example, Google’s **“Cloud Storage API”** had a method that returned a list of files in a bucket.

The Situation

The API contract only promised: “Returns a list of files.” But in practice, the implementation happened to return the files sorted by filename (even though sorting was not documented).

What Happened

- Many developers wrote applications that depended on the **accidental ordering.**
- When Google later optimized the backend and stopped sorting (to improve performance), thousands of applications broke.

Lesson

Even though the API did not guarantee order, users had treated the observable side-effect (sorting) as a feature. Google learned:

* Every observable detail becomes a dependency. Once released, it’s nearly impossible to remove such behavior without breaking customers.

Mitigation Strategies: Organizations use several strategies to reduce the risks of Hyrum’s Law:

Strict API Contracts – Be explicit about guarantees.

Black-box Testing – Check what clients might depend on unintentionally.

Deprecation Policies – Provide long transition periods before removing accidental behaviors.

Feature Flags / Versioning – Allow old behavior to coexist for legacy users.

Hyrum’s Law reminds us that

“**Users will exploit every behavior they observe” — whether or not it was intended. Once shipped, your accidental design decisions become permanent API commitments**.