Leaky bucket

Leaky Bucket হলো রেট লিমিটিং করার এমন একটি পদ্ধতি যা নিশ্চিত করে যে সার্ভার থেকে তথ্য বা রিকোয়েস্ট সব সময় একটি নির্দিষ্ট এবং স্থির গতিতে (Constant Rate) বের হবে।

এটি অনেকটা ফুটো হওয়া একটি পানির বালতির মতো। আপনি বালতিতে যে গতিতেই পানি ঢালুন না কেন, নিচের ফুটো দিয়ে পানি কিন্তু সব সময় একই গতিতে ফোঁটায় ফোঁটায় পড়বে।

১. Leaky Bucket **কীভাবে কাজ করে**?

এই অ্যালগরিদমটি বোঝার জন্য নিচের ধাপগুলো কল্পনা করুন:

1. **The Bucket (Queue):** এখানে বালতিটি হলো একটি **FIFO (First-In-First-Out) Queue**। যার একটি নির্দিষ্ট ধারণক্ষমতা আছে।
2. **Incoming Requests (Water In):** ইউজার যখন রিকোয়েস্ট পাঠায়, তখন সেগুলো বালতিতে জমা হয়। যদি বালতি ভরে যায় (Overflow), তবে নতুন রিকোয়েস্টগুলো ফেলে দেওয়া হয় (Discard)।
3. **Outgoing Requests (Water Out):** বালতির নিচ দিয়ে একটি নির্দিষ্ট হারে (Fixed Rate) রিকোয়েস্টগুলো প্রসেস হওয়ার জন্য বের হয়।

২. Token Bucket-**এর সাথে এর পার্থক্য কী**?

অনেকে এই দুটির মধ্যে গুলিয়ে ফেলেন, তবে এদের মূল পার্থক্য হলো আউটপুট বা কাজের গতিতে:

বৈশিষ্ট্য	Token Bucket	Leaky Bucket
Burst Traffic	হঠাৎ অনেক রিকোয়েস্ট আসলে তা অ্যালাউ করে (যদি টোকেন থাকে)।	হঠাৎ অনেক রিকোয়েস্ট আসলেও আউটপুট সব সময় স্থির থাকে।

বৈশিষ্ট্য	Token Bucket	Leaky Bucket
Output Rate	আউটপুট রেট উঠানামা করতে পারে।	আউটপুট রেট সব সময় সমান থাকে (Smooth)।
ব্যবহার	সাধারণ API লিমিটিংয়ের জন্য জনপ্রিয়।	নেটওয়ার্ক ট্রাফিক শেপিং (Traffic Shaping)-এর জন্য ব্যবহৃত হয়।

৩. সুবিধা ও অসুবিধা

সুবিধা:

- **স্থির গতি** (Smooth Flow): এটি সার্ভারের ওপর হঠাৎ বড় কোনো চাপের সৃষ্টি হতে দেয় না। আউটপুট সব সময় কন্ট্রোলে থাকে।
- **সহজ ইমপ্লিমেন্টেশন**: এটি একটি সিম্পল কিউ (Queue) ব্যবহার করে তৈরি করা যায়।

অসুবিধা:

- **ইনফ্লেক্সিবিলিটি**: যদি সার্ভার ফ্রিও থাকে, তবুও এটি তার নির্দিষ্ট গতির চেয়ে দ্রুত কাজ করবে না। ফলে রিসোর্সের অপচয় হতে পারে।
- **ডেটা লস**: বালতি ভরে গেলে নতুন রিকোয়েস্টগুলো কোনো সুযোগ না দিয়েই সরাসরি রিজেক্ট করে দেয়।

৪. বাস্তব জীবনের উদাহরণ

মনে করুন একটি ব্যাংকের কাউন্টার। সেখানে একজন মাত্র অফিসার আছেন।

- বাইরে থেকে ১০ জন মানুষ একসাথে লাইনে আসুক বা ১ জন করে আসুক, অফিসার কিন্তু প্রতি ৫ মিনিটে ১ জনের বেশি মানুষের কাজ করবেন না।
- যদি ব্যাংকের ভেতরে দাঁড়ানোর জায়গা শেষ হয়ে যায়, তবে নতুন আসা মানুষদের গেট থেকেই বিদায় করে দেওয়া হবে। এটাই হলো Leaky Bucket।

৫. কোথায় ব্যবহার করা হয়?

- **Traffic Shaping:** ইন্টারনেট সার্ভিস প্রোভাইডাররা (ISP) এটি ব্যবহার করে যাতে কেউ হঠাৎ ব্যান্ডউইথ স্পাইক ঘটিয়ে নেটওয়ার্ক স্লো না করতে পারে।
- **E-commerce:** বড় সেলের সময় (যেমন: ১১.১১ বা ব্ল্যাক ফ্রাইডে) যখন কোটি কোটি রিকোয়েস্ট আসে, তখন ডাটাবেসকে রক্ষা করতে এটি ব্যবহার করা হয়।

সারাংশ:

Leaky Bucket হলো একটি ট্রাফিক কন্ট্রোল সিস্টেম যা অনিয়মিত রিকোয়েস্টগুলোকে একটি সুন্দর, গোছানো এবং স্থির গতিতে রূপান্তর করে।

🪣 Leaky Bucket কী?

Leaky Bucket হলো একটি **rate limiting + traffic shaping algorithm**।

Bucket-এ request ঢুকতে পারে যেকোনো গতিতে
কিন্তু **বের হয় constant rate এ**
ঠিক যেন ফুটো বালতি 🚰

🧠 Core Idea

- Bucket = Queue
- Incoming request = Water
- Leak rate = Fixed processing speed

Incoming traffic → [Bucket] → Constant outflow

⚙️ How it works (Step by Step)

Configuration:

Bucket capacity = 10 requests

Leak rate = 1 request / second

Scenario:

1. Sudden 10 request → bucket ভরে যায়
2. 11th request → drop ❌
3. প্রতি second এ 1টা করে request process
4. Traffic smooth & stable ✅

❌ Burst handle করতে পারে না কেন?

👉 Output speed **fixed**
👉 Extra burst queue overflow করে drop

এটাই Token Bucket থেকে main difference 🔥

📈 Visualization

Burst input: ██████████

Output rate: █ █ █ █

🧱 Leaky Bucket Architecture

Client

↓

Load Balancer

↓

Leaky Bucket Queue

↓ (fixed rate)

Worker / Server

⚖️ Leaky Bucket vs Token Bucket

Topic	**Leaky Bucket**	**Token Bucket**
Burst allowed	❌ No	✅ Yes
Output rate	Constant	Variable
Traffic shaping	Excellent	Medium
User-friendly	Medium	High
API rate limit	❌	✅

🧠 When to use Leaky Bucket?

✓ Network traffic shaping
✓ Video streaming
✓ ISP bandwidth control
✓ Systems that need **smooth & predictable load**

🧪 Pseudocode (Concept)

```
if queue not full:
  enqueue request
else:
  drop request
```

🌍 Real-world usage

- Routers
- Firewalls
- ISP traffic controllers
- Old-school rate limiters

🧠 Interview Golden Lines ⭐

“Leaky Bucket smooths traffic by enforcing a constant processing rate, sacrificing burst tolerance.”

“It is better suited for traffic shaping than API rate limiting.”

📝 Memory Trick

Token Bucket = Save tokens 🪙

Leaky Bucket = Drain water 🚰

🔥 Quick Decision Guide

API → Token Bucket

Network → Leaky Bucket

Burst OK → Token

Smooth flow → Leaky