apigee

Traffic Management and Rate Limiting

Google Cloud

Traffic management policies

Traffic management policies let you configure cache, control traffic quotas and spikes, set concurrent rate limits, and so on.

It gives you control over raw throughput, and can also control traffic on a per-app basis. Traffic management policy types enable you to enforce quotas, and they also help you to mitigate denial of service attacks.

Each of the traffic management policy types addresses a distinct aspect of traffic management. In some cases, you might use the three policy types in a single API proxy.

In addition, Apigee Edge includes a cache, that can also be accessed via policies, to manage traffic and handling of requests.

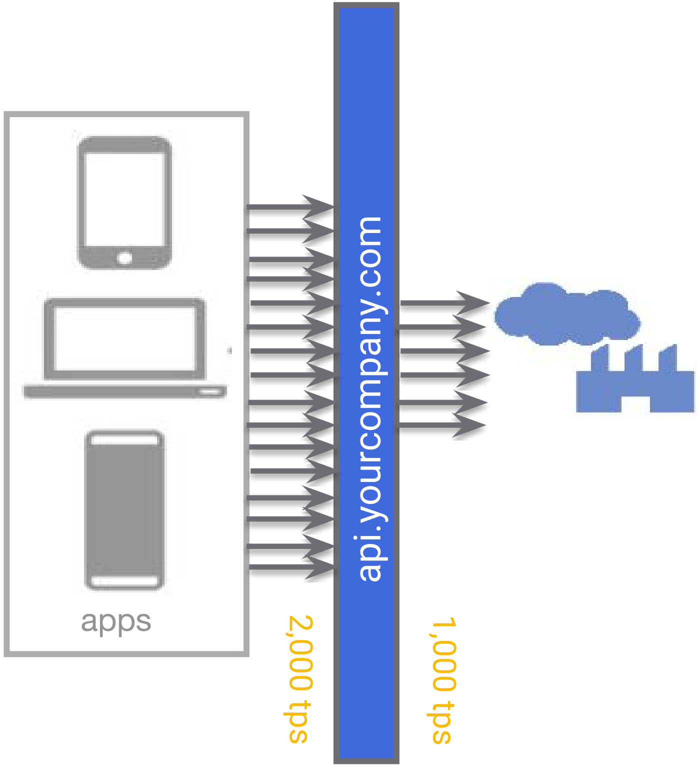
Traffic Management Policy - Spike Arrest

Spike arrest

- Spike Arrest smooth's inflow request patterns over shorter intervals to ensure that demand does not outstrip capacity.

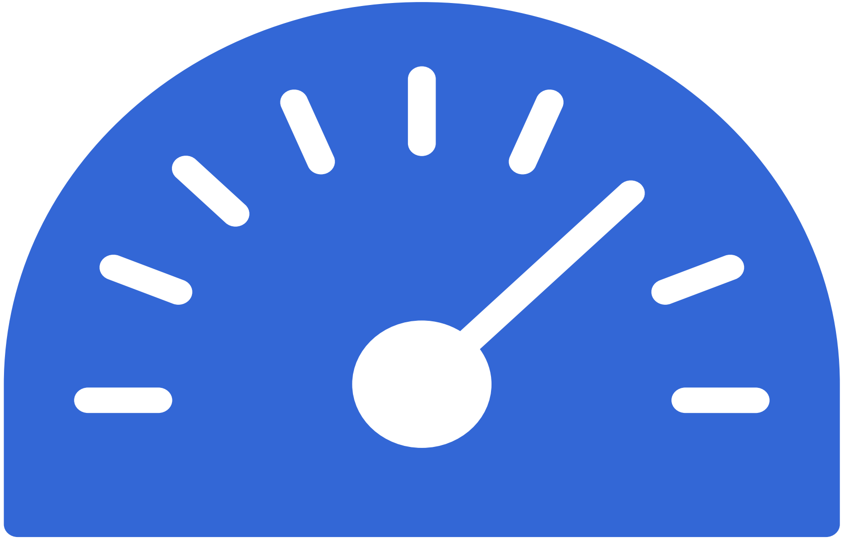
- Spike Arrest is a technical requirement to protect the backend as opposed to a Quota or Rate Limit which is a business requirement to manage developer relationships.

- See Also: http://docs.apigee.com/api-services/reference/spike-arrest-policy#howspikearrestworks



Spike arrest and Denial of Service

- Denial of Service attacks is an attack on your platform instigated by sending thousands of requests against your API as a means to either bring down your platform or expose other vulnerabilities
- The Spike Arrest policy protects against traffic spikes.
- It throttles the number of requests processed by an API proxy and sent to a backend, protecting against performance lags and downtime.
- Spike Arrest monitors the rate at which traffic comes in, it does not count each request by putting them in timed buckets.

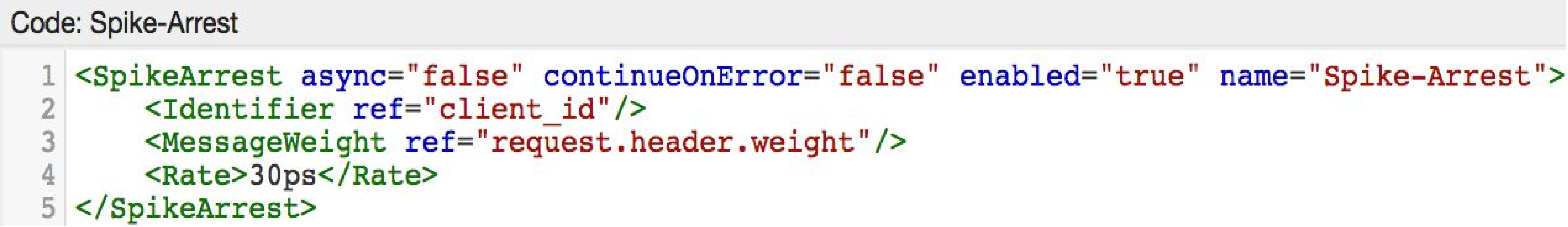


apigee

Spike arrest configuration

- Identifier
  - Mechanism to scope your incoming traffic
  - Uses a separate rate for each unique Id
- Message weight
  - Used to give extra weight to conditioned requests

- Rate
  - Specifies the rate you want to allow traffic to come into your API Proxy.
  - Rate is for each Message Processors
  - Rate is formatted with a number and a unit
  - Unit is defined as ps, pm (per second, per minute)



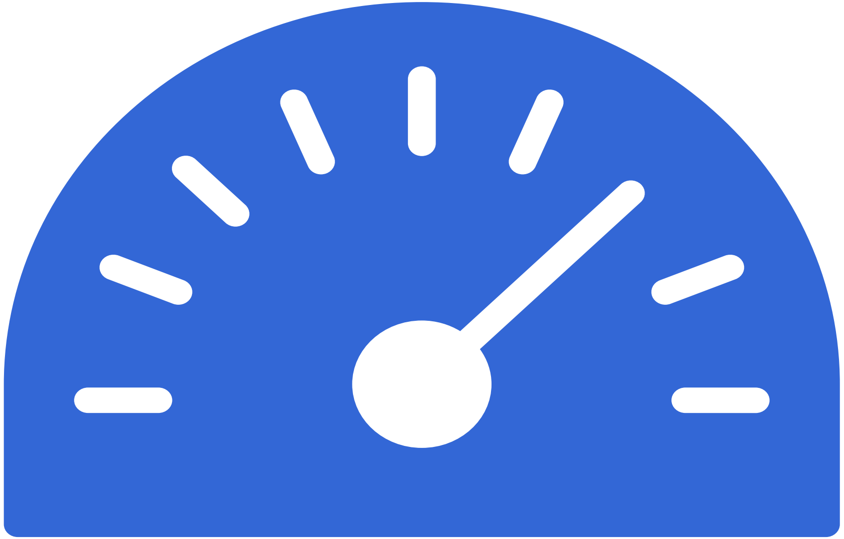
Code: Spike-Arrest

```
<SpikeArrest async="false" continueOnError="false" enabled="true" name="Spike-Arrest">
    <Identifier ref="client_id"/>
    <MessageWeight ref="request.header.weight"/>
    <Rate>30ps</Rate>
</SpikeArrest>
```

Traffic Management Policy - Quota

Quota

- Use quota to allow a defined number of requests to an entity
- Normally used for business logic requirements unlike Spike Arrest or Concurrent Rate Limit policies which are for security requirements
- Quotas use buckets of requests per time unit, unlike spike arrest, which limits based on the rate of traffic
- Once a quota limit is reached all requests generated by the limited entity will be rejected



Quota configurations

- Identifier
  - Mechanism to scope your incoming traffic
  - Uses a separate bucket for each unique Id
- Message weight
  - Used to give extra weight to conditioned requests

- Allow
  - Specifies the number of requests allowed for a particular entity
- Time Unit
  - Month, Day, Week, Minute, Year
- Interval
  - Frequency of the time unit

Quota type

Quota type indicates when the quota counter starts counting usage. Quota type specified in “type” attribute of the Quota root element

- **calendar** (default if not specified)
  - Quota counting has an explicit start time
  - <StartTime> is specified and quota counting starts at the StartTime
  - <StartTime> is reset based on the specified <Interval> and <TimeUnit>
- **rollingwindow**
  - Quota uses a rolling window that resets based on the <Interval> and <TimeUnit>
  - The start time is the time the first message is received matching the <Identifier>
  - rollingwindow allows a quota that recalculates the window on each message
  - See Also: http://docs.apigee.com/api-services/reference/quota-policy#quotapolicytypes
- **flexi**
  - Start time is dynamic for each <Identifier> based on first message being received
  - Calls can be used until interval has elapsed
  - Quota does not automatically reset at the end of the interval
  - flexi allows access for a specified period of time

Distributed quotas

- <Distributed> specifies whether the count is shared among all message processors (Distributed = true) or maintained separately for each message processor (Distributed = false)
- <Synchronous> specifies whether the distributed quota counter is updated synchronously
- <AsynchronousConfiguration> specifies how an asynchronous distributed quota is updated
- If <PreciseAtSecondsLevel> is set to true, the quota will be enforced with an accuracy of a second, even if the <TimeUnit> is set at a unit longer than a second

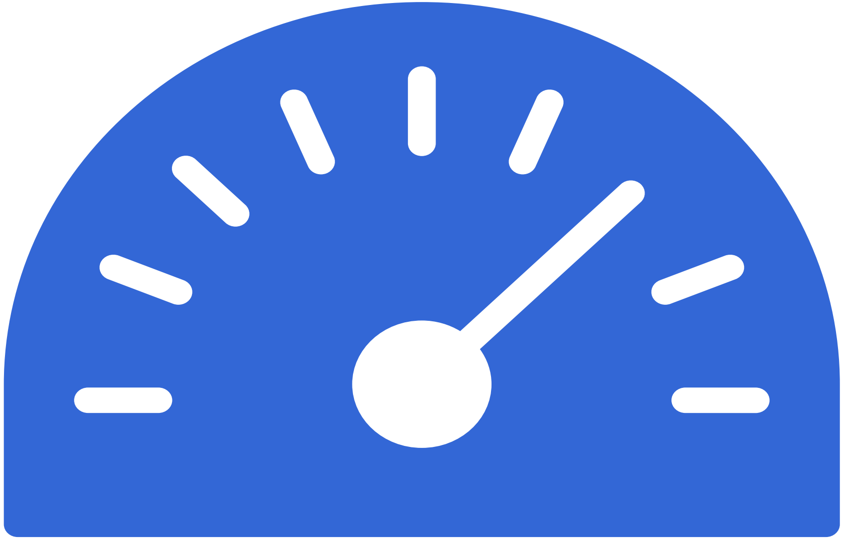
apigee

Traffic Management Policy - Concurrent Rate Limit

Google Cloud

Concurrent rate limit

- Throttles inbound connections from your API proxies running on Apigee Edge to your backend services.
- This policy has specific placement requirements. It must be attached on both the Request and Response flows in the **Target Endpoint**
  - In addition, it is recommended that you place it in a DefaultFaultRule (this ensures that, in the event of an error, the rate limit counters are maintained correctly)
- Use this policy to limit this traffic to a manageable number of connections



apigee

Need help deciding which rate limiting policy to use?

http://docs.apigee.com/api-services/content/comparing-quota-spike-arrest-and-concurrent-rate-limit-policies

Thank You