Demystifying Kubernetes API Priority and Fairness (APF)

Naresh Amrutham

Software Engineer at Crusoe

SCaLE 23x

WHAT IS APF?

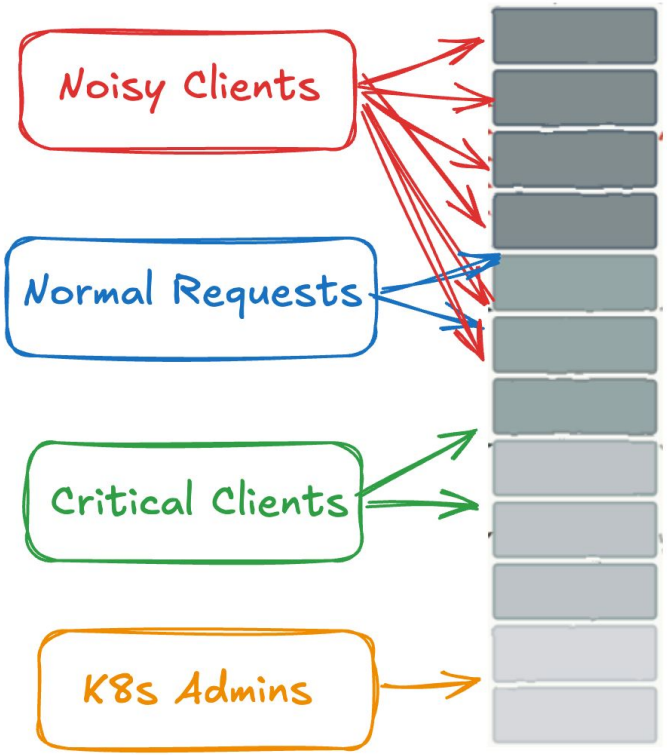
API Priority and Fairness (APF) in Kubernetes, classifies and isolates API server requests in a more fine-grained way, introducing prioritization, queuing, and isolation.

So critical requests always get through and no single client can starve the rest.

Kubernetes v1.29 [stable]

The Problem — Before APF

--max-requests-inflight
--max-mutating-requests-inflight



No awareness of who

Noisy Clients consume all capacity

Critical requests have zero protection and instantly dropped

No protection for the kube-apiserver

APF Goals

Fair Sharing

Max Allowed Requests

Capacity distributed equitably among all the clients

Prioritization

Critical Requests ↑ High Priority

Non - Critical Requests ↓ Low Priority

Queuing

Max Allowed Requests - FULL

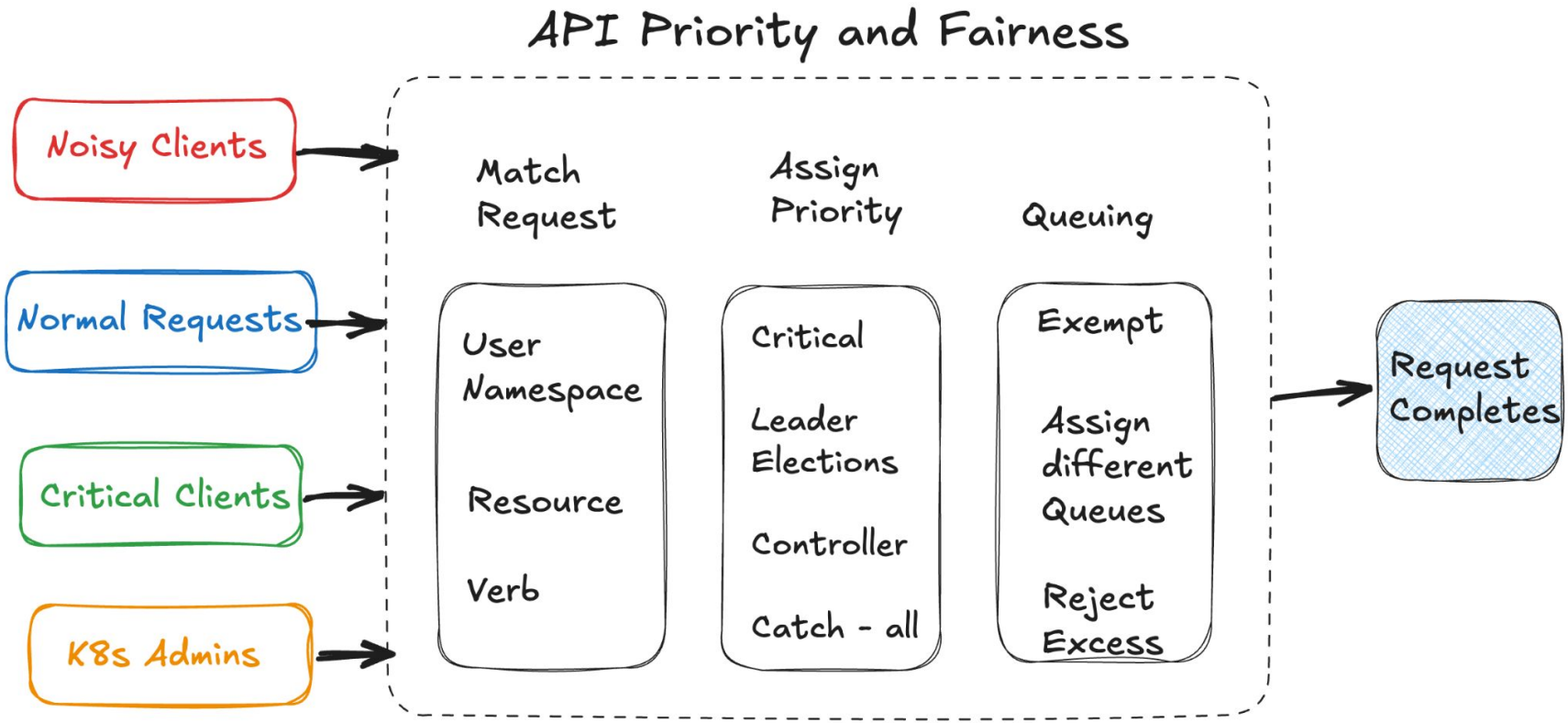
Excess requests are queued instead of getting dropped

Flow Isolation

Limit expensive and Noisy requests

Protect essential and critical requests with their own flow

APF - High Level



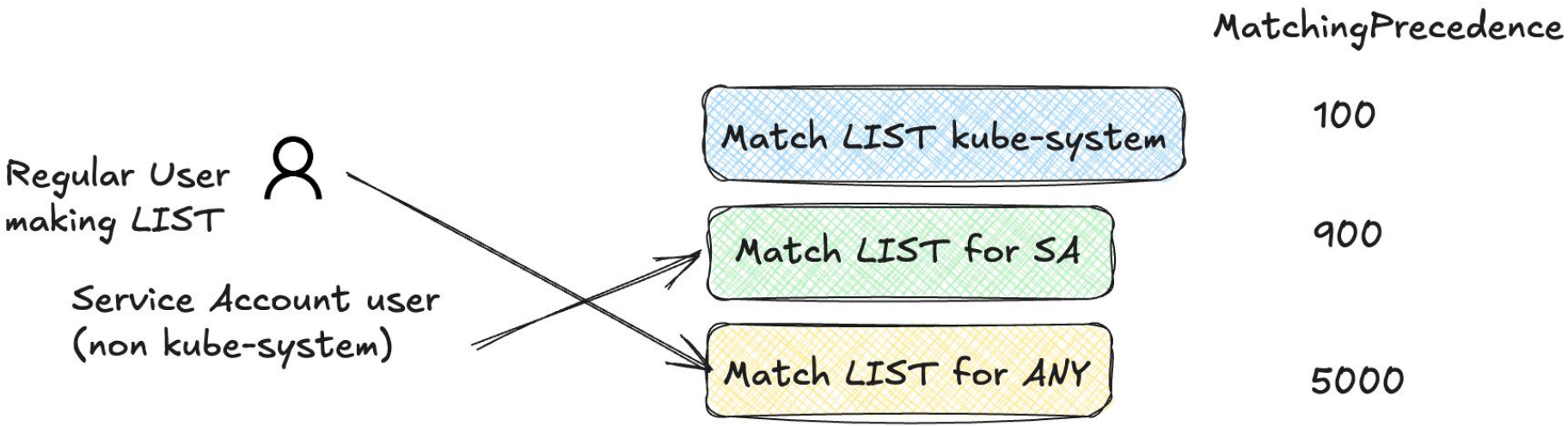
APF RESOURCES

FlowSchema

Matches incoming API requests against rules (user, verb, resource, namespace) and routes them to the appropriate **PriorityLevelConfiguration**.

```
→  ~ kubectl get flowschema
NAME                           PRIORITYLEVEL     MATCHINGPRECEDENCE   DISTINGUISHERMETHOD   AGE     MISSINGPL
exempt                         exempt            1                    <none>                9m49s   False
probes                         exempt            2                    <none>                9m49s   False
system-leader-election         leader-election   100                  ByUser                9m49s   False
endpoint-controller            workload-high     150                  ByUser                9m49s   False
workload-leader-election       leader-election   200                  ByUser                9m49s   False
system-node-high               node-high         400                  ByUser                9m49s   False
system-nodes                   system            500                  ByUser                9m49s   False
kube-controller-manager        workload-high     800                  ByNamespace           9m49s   False
kube-scheduler                 workload-high     800                  ByNamespace           9m49s   False
kube-system-service-accounts   workload-high     900                  ByNamespace           9m49s   False
service-accounts               workload-low      9000                 ByUser                9m49s   False
```

FLOWSCHEMA MatchingPrecedence



FLOWSCHEMA RESOURCE SPEC

ByUser, ByNamespace, None

```
~ k get flowschema service-accounts -o yaml | neat
apiVersion: flowcontrol.apiserver.k8s.io/v1
kind: FlowSchema
metadata:
  annotations:
    apf.kubernetes.io/autoupdate-spec: "true"
  name: service-accounts 1
spec:
  distinguisherMethod:
    type: ByUser 2
  matchingPrecedence: 9000 3
  priorityLevelConfiguration:
    name: workload-low 4
  rules:
  - nonResourceRules:
    - nonResourceURLs:
      - '*'
      verbs:
      - '*'
    resourceRules:
    - apiGroups:
      - '*'
      verbs:
      - '*'
  subjects:
  - group:
      name: system:serviceaccounts
    kind: Group
```

APF RESOURCES

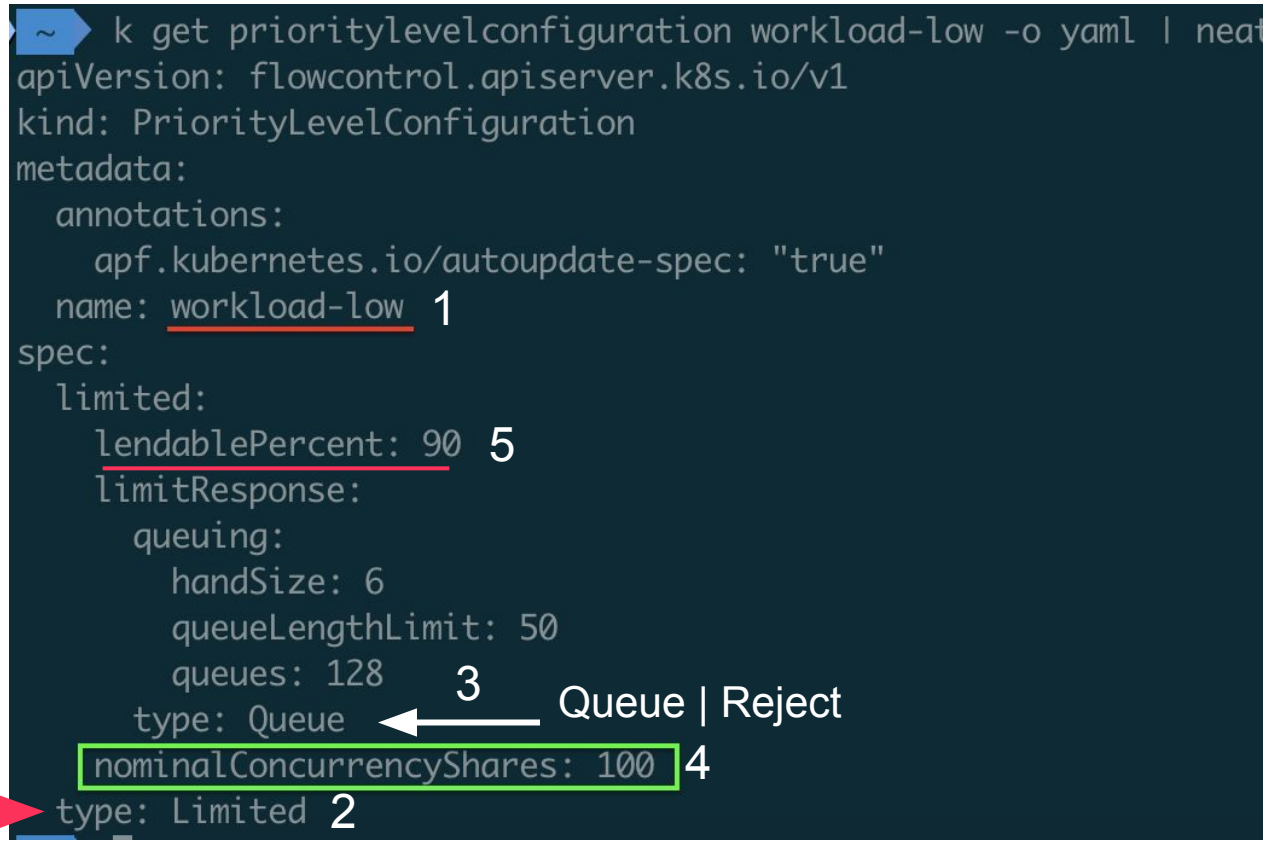
PriorityLevelConfiguration

Defines a priority tier, concurrency share, and queues.

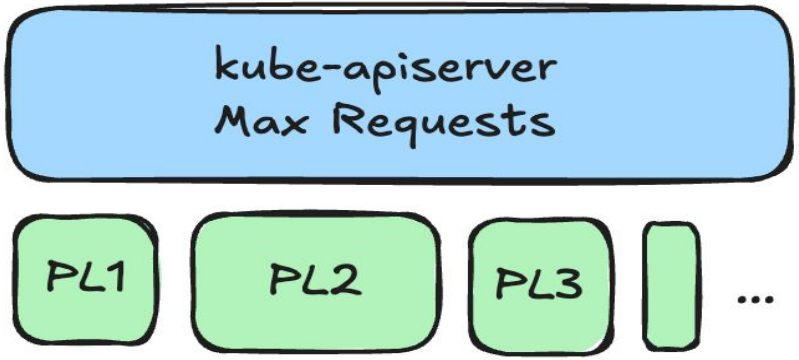
How to handle requests when over capacity. I.e. admit, queue, or reject.

```
→  ~ kubectl get prioritylevelconfiguration
NAME              TYPE      NOMINALCONCURRENCYSHARES   QUEUES   HANDSIZE   QUEUELENGTHLIMIT   AGE
catch-all         Limited   5                          <none>   <none>     <none>             2m50s
exempt            Exempt    <none>                     <none>   <none>     <none>             2m50s
global-default    Limited   20                         128      6          50                 2m50s
leader-election   Limited   10                         16       4          50                 2m50s
node-high         Limited   40                         64       6          50                 2m50s
system            Limited   30                         64       6          50                 2m50s
workload-high     Limited   40                         128      6          50                 2m50s
workload-low      Limited   100                        128      6          50                 2m50s
```

PriorityLevelConfiguration RESOURCE SPEC



CONCURRENCY / SEATS



concurrency (**seats**) =

apiserver-max-requests * **nominalConcurrencyShares** / **sum(nominalConcurrencyShares)**

CONCURRENCY / SEATS

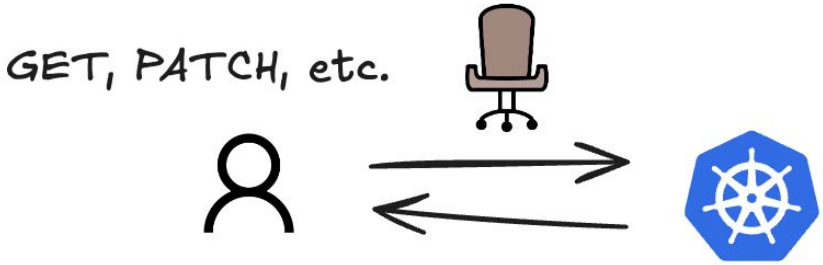
```
spec:
  limited:
    lendablePercent: 0
    limitResponse:
      type: Reject
    nominalConcurrencyShares: 5
  type: Limited
```

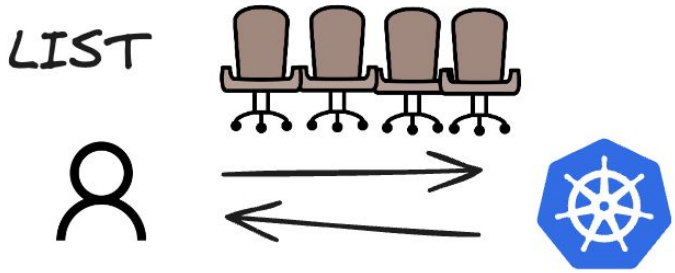
For example, to calculate concurrency for PriorityLevel **catch-all** having **5 nominalConcurrencyShares**

Max allowed requests = 600
Sum(nominalConcurrencyShares) = 245

concurrency = 600 * 5/245 = 12

LIST REQUEST SEATS ASSIGNMENT





kubectl get pods -A

Queue

Queues

More Queues -> Fewer collisions

Queues: 1 -> Disables fair queuing

queueLengthLimit

Low limit

Sudden bursts -> requests get dropped

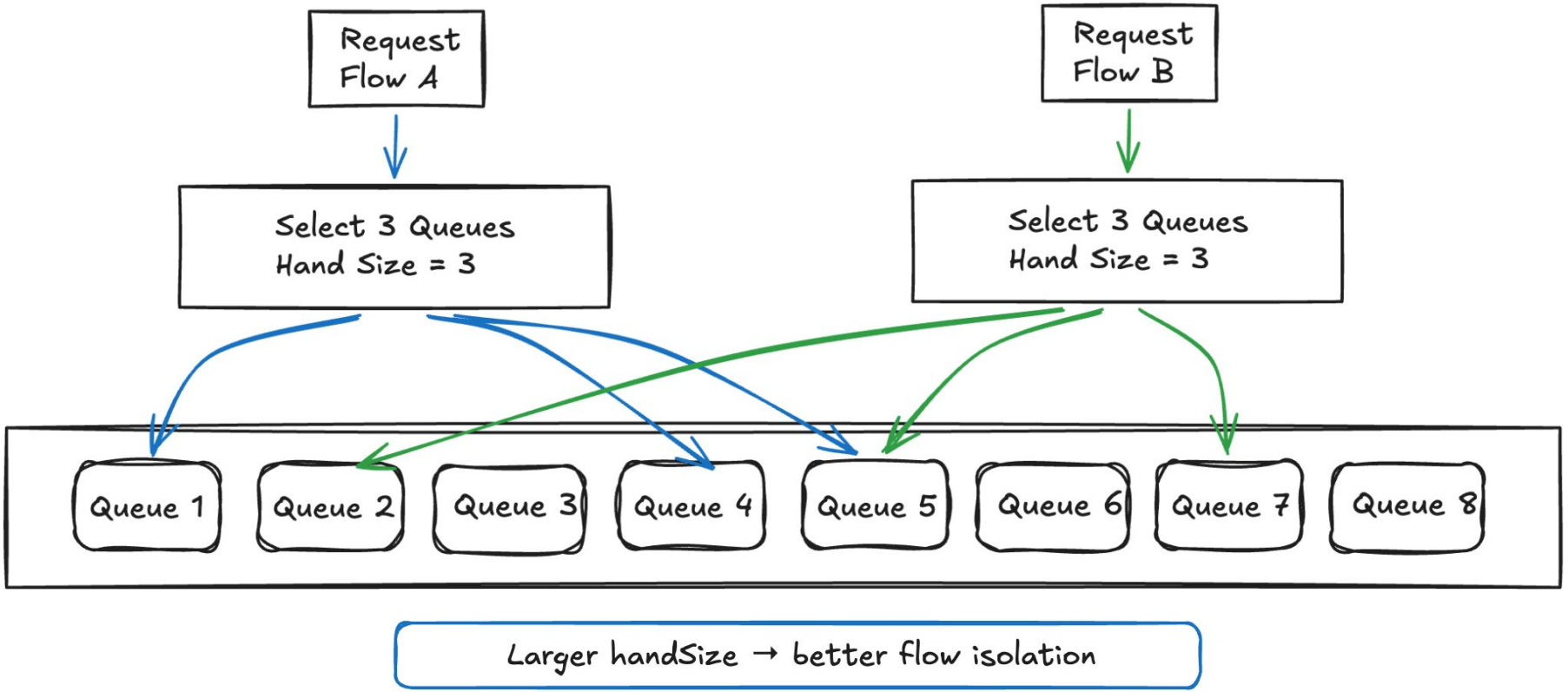
High limit

Sudden bursts -> requests are queued

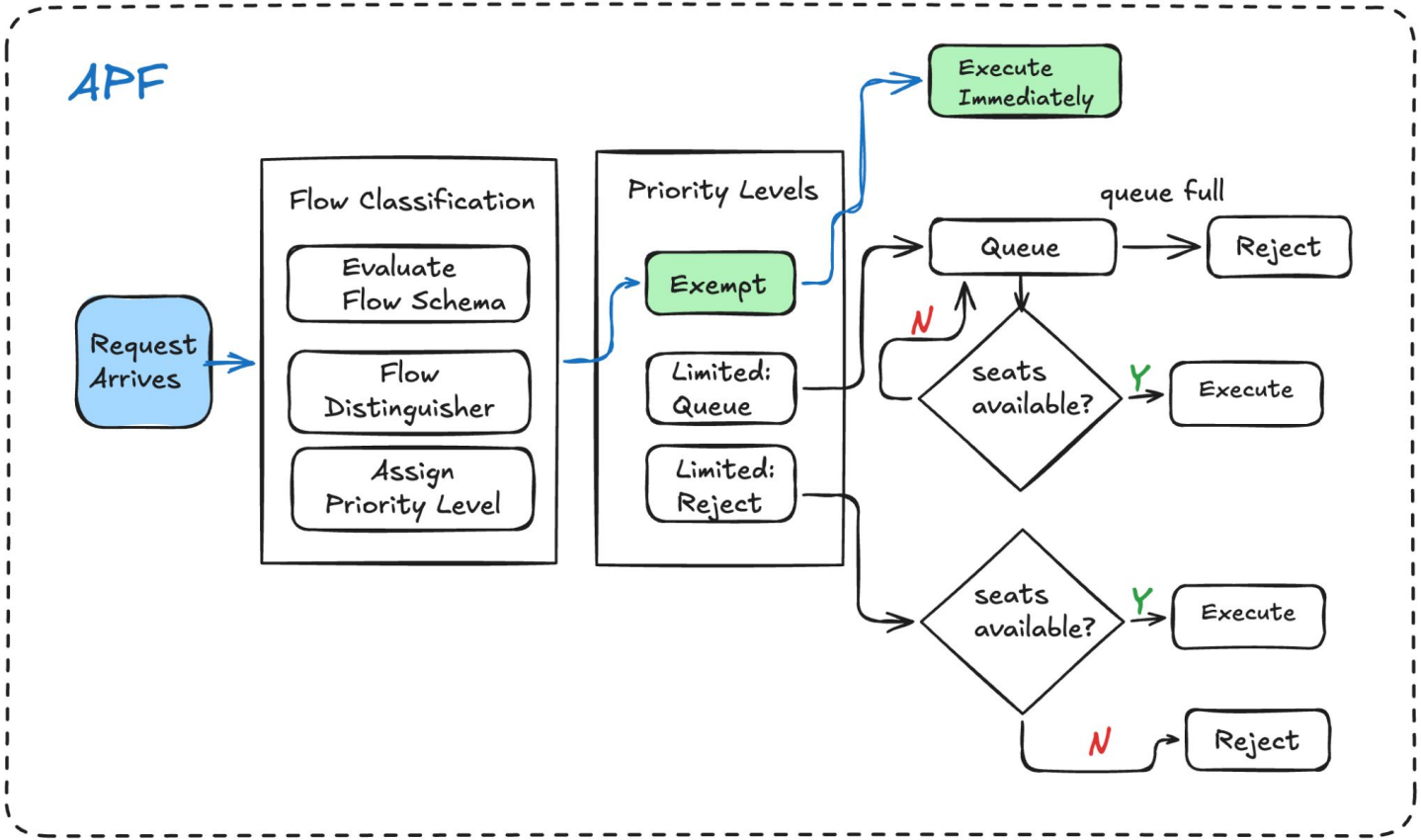
Memory and Latency

QUEUE - Hand Size - Shuffle Sharding

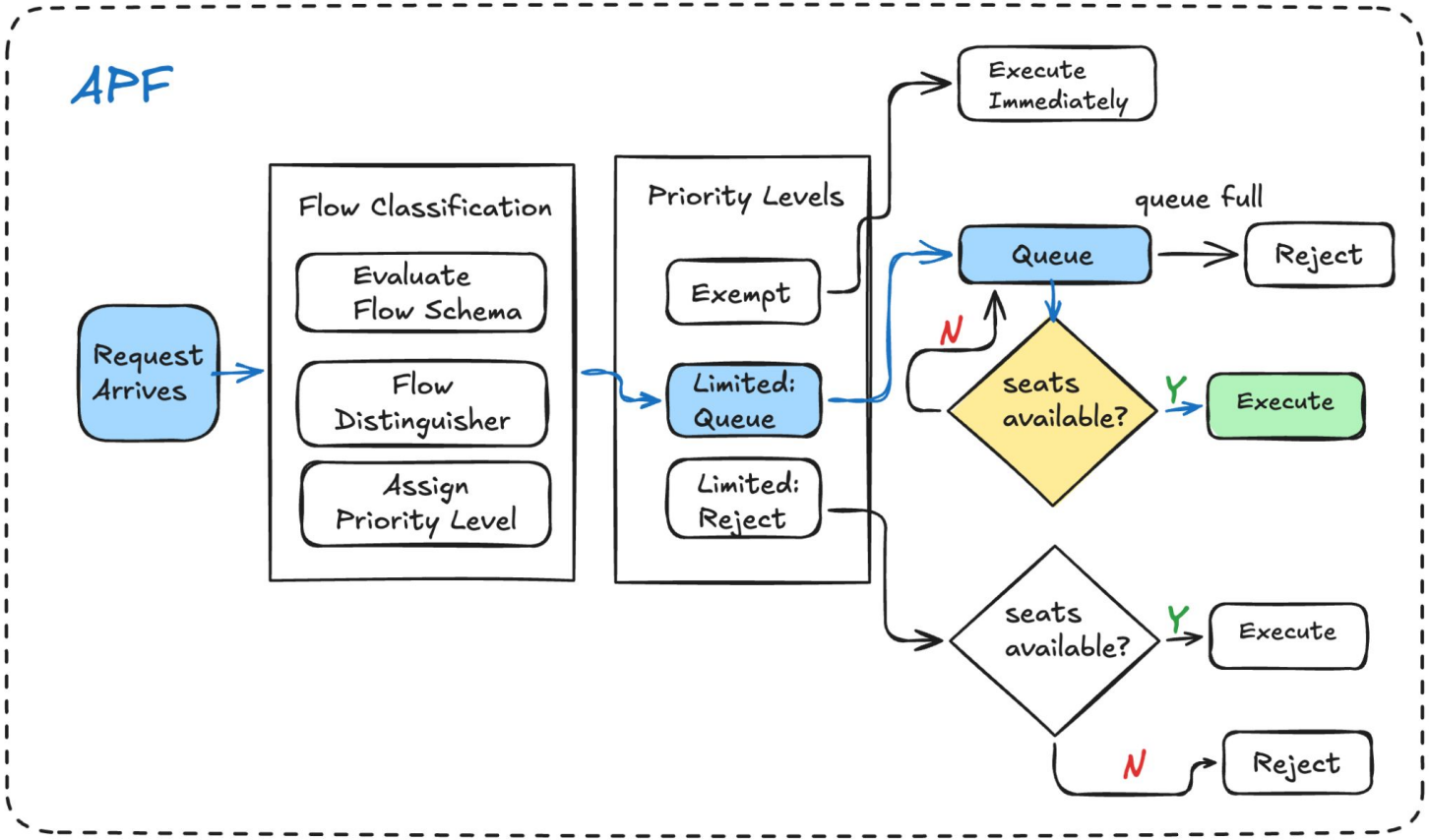
```
    limitResponse:
      queuing:
        handSize: 3
        queueLengthLimit: 20
        queues: 8
      type: Queue
```



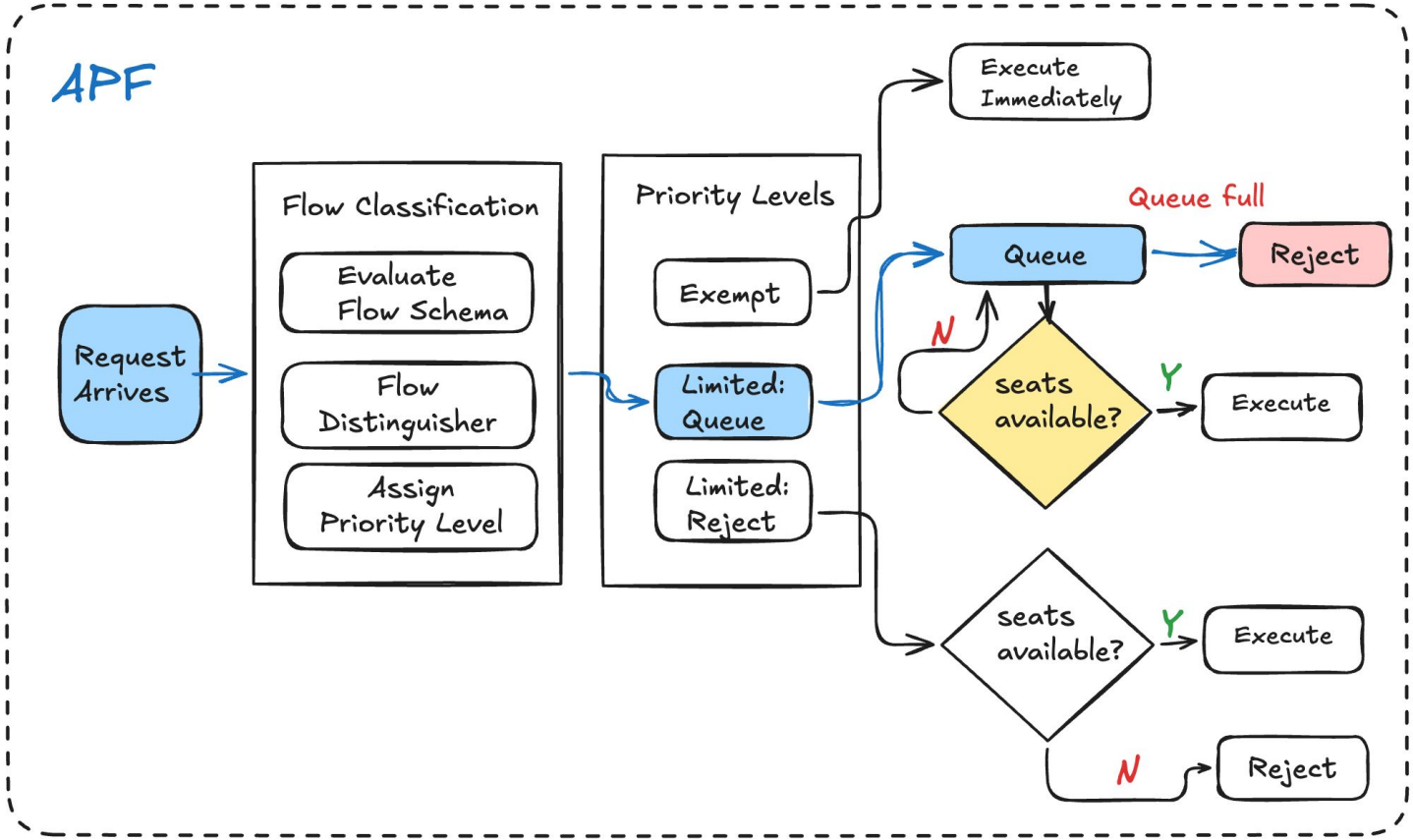
REQUEST FLOW - Exempt



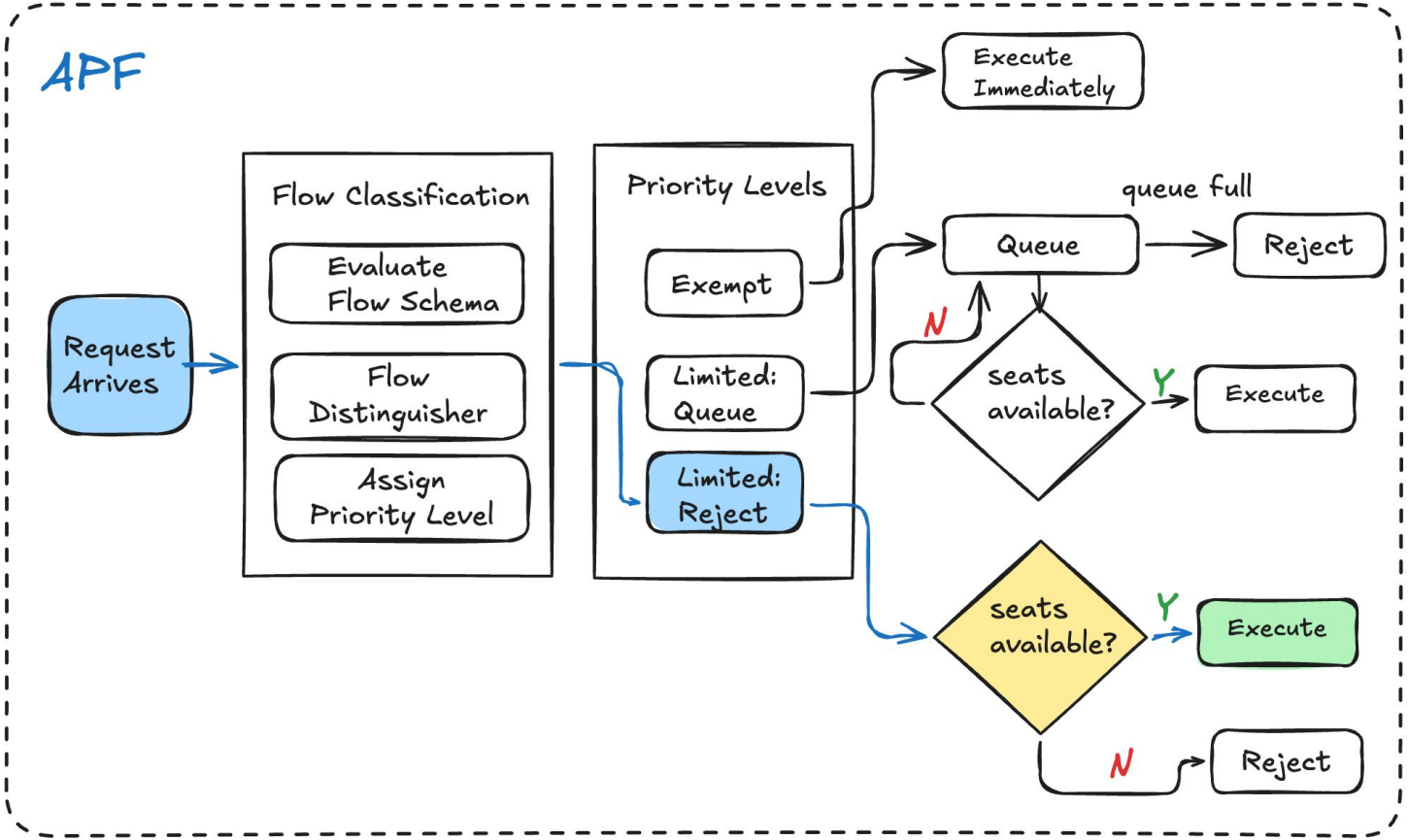
REQUEST FLOW - Queue



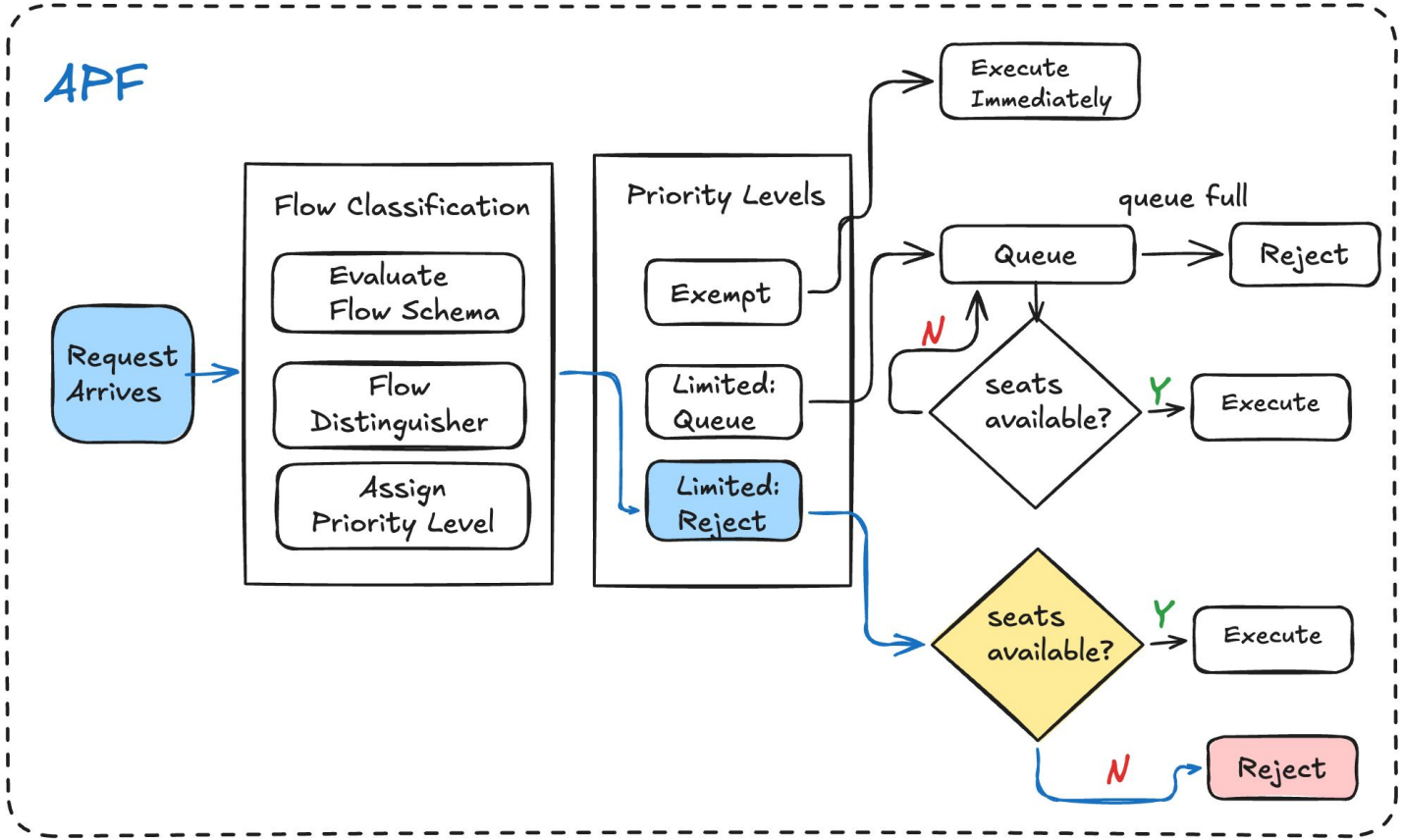
REQUEST FLOW - Queue



REQUEST FLOW - Reject



REQUEST FLOW - Reject



APF Metrics

1. apiserver_flowcontrol_dispatched_requests_total
2. apiserver_flowcontrol_current_inqueue_requests
3. apiserver_flowcontrol_rejected_requests_total
4. apiserver_flowcontrol_current_executing_requests
5. apiserver_flowcontrol_current_executing_seats
6. apiserver_flowcontrol_request_wait_duration_seconds

APF DEBUG ENDPOINTS

```
~ kubectl get --raw /debug/api_priority_and_fairness/dump_priority_levels
PriorityLevelName, ActiveQueues, IsIdle, IsQuiescing, WaitingRequests, ExecutingRequests, DispatchedRequests, RejectedRe
catch-all,         0,            true,   false,       0,              0,                 0,                  0,
exempt,            0,            true,   false,       0,              0,                 248685,             0,
global-default,    0,            false,  false,       0,              1,                 85,                 0,
leader-election,   0,            true,   false,       0,              0,                 0,                  0,
node-high,         0,            true,   false,       0,              0,                 133,                0,
system,            0,            true,   false,       0,              0,                 1165,               0,
workload-high,     0,            true,   false,       0,              0,                 162697,             0,
workload-low,      0,            true,   false,       0,              0,                 2044303,            0,
```

```
~ kubectl get --raw /debug/api_priority_and_fairness/dump_queues
PriorityLevelName, Index,  PendingRequests, ExecutingRequests, SeatsInUse, NextDispatchR,      InitialSeatsSum,
system,            0,      0,               0,                 0,          189.34575101ss,    0,
system,            1,      0,               0,                 0,          0.00000000ss,      0,
system,            2,      0,               0,                 0,          0.00000000ss,      0,
system,            3,      0,               0,                 0,          0.00000000ss,      0,
system,            4,      0,               0,                 0,          190.27955853ss,    0,
system,            5,      0,               0,                 0,          190.29421813ss,    0,
system,            6,      0,               0,                 0,          0.00000000ss,      0,
system,            7,      0,               0,                 0,          1414.77944926ss,   0,
system,            8,      0,               0,                 0,          0.00000000ss,      0,
```

APF Logs - Client Side

APF adds the following two headers to each HTTP response message

```
I0305 23:42:17.258315   72399 round_trippers.go:632] "Response" status="200 OK" h
        Audit-Id: 17885de7-01dc-4d52-bb9c-7323dd2a4b73
        Cache-Control: no-cache, private
        Content-Type: application/json
        Date: Fri, 06 Mar 2026 07:42:19 GMT
        X-Kubernetes-Pf-Flowschema-Uid: 478a779e-88d2-4f4d-a246-cc175d6a2687
        X-Kubernetes-Pf-Prioritylevel-Uid: 68222f25-8e1a-4ec8-b032-58135e0dd646
```

Waited for 1.033772408s due to **client-side throttling, not priority and fairness**, request: GET:https://api.example.org/apis/pkg.crossplane.io/v1?timeout=32s

APF Logs - Server Side

At -v=3 or more verbosity, the API server outputs an httplog line for every request

```
I0517 17:10:42.517920      11 httplog.go:132] "HTTP"
verb="GET" URI="/metrics" latency="554.668µs"
userAgent="pod_nanny/1.8.14"
audit-ID="5b4e4c63-bb9f-4293-bd2d-2e62409adc95" srcIP=""
apf_pl="workload-high"
apf_fs="kube-system-service-accounts" apf_iseats=1
apf_fseats=0 apf_additionalLatency="0s"
apf_execution_time="133.04µs" resp=403
```

APF Best Practices

- Use separate FlowSchemas for mutating
- Throttle / limit expensive LIST requests
- Ensure non-critical clients have lower priority
- Assign higher priority levels to essential system components
- Set appropriate FlowSchema for critical requests (e.g., Kubernetes administrators, leader elections, leases, etc.)
- Ensure requests from non-critical DaemonSets are within limits
- Careful in recursive requests scenarios

THANK YOU

Q & A

We are hiring!

Contact: linkedin.com/in/nareshku