ingress-nginx Inventory: Annotations, ConfigMaps, Flags

2026-07-18

Before comparing replacement controllers, find out what your ingress-nginx clusters actually depend on: annotations, the global ConfigMap, and controller flags.

Most migration guides start by comparing replacement Ingress controllers. That's the wrong first step. The first step is finding out which `ingress-nginx` behaviours your own clusters actually depend on — because the answer changes which controller (or Gateway API implementation) is even a candidate, and it's usually smaller than a full annotation reference makes it look.

► TL;DR

Before picking a replacement, inventory three surfaces: the `nginx.ingress.kubernetes.io/*` annotations actually set on your Ingress objects, the global controller `ConfigMap` (which affects every route and is invisible if you only look at annotations), and any controller flags that gate annotation behaviour. An annotation being present doesn't mean it's load-bearing — the only way to know is whether removing it changes observed behaviour, not whether it's copied into a Helm chart.

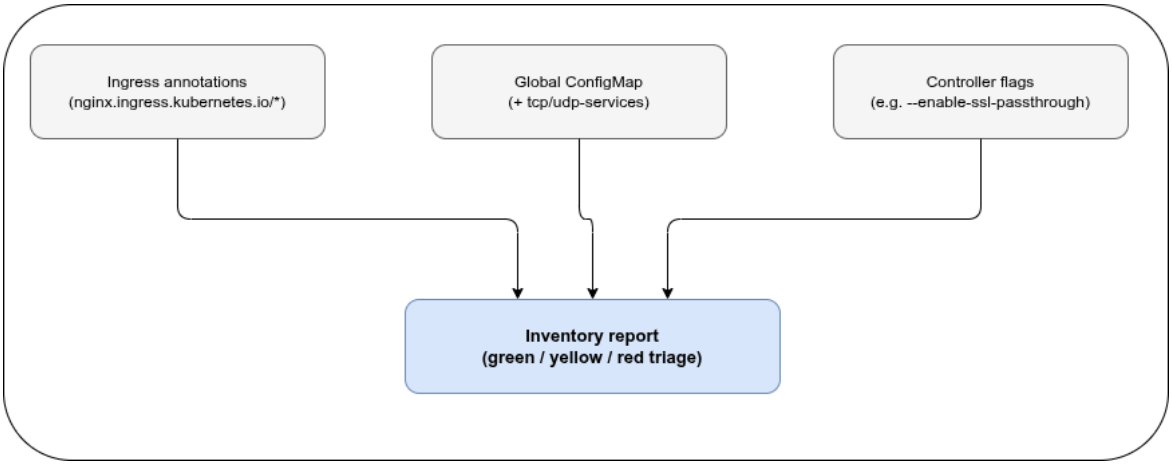

Figure 1: Three ingress-nginx inventory surfaces — annotations, global ConfigMap, and controller flags feed one inventory report

Surface 1: Ingress annotations

`ingress-nginx`'s behaviour is mostly configured through annotations on individual `Ingress` objects, all prefixed `nginx.ingress.kubernetes.io/`. The following `kubectl`/`jq` one-liner extracts every one of them across every namespace:

```
kubectl get ingress -A -o json \
  | jq -r '
    .items[]
    | .metadata as $m
    | ($m.annotations // {})
    | to_entries[]
    | select(.key | startswith("nginx.ingress.kubernetes.io/"))
    | [$m.namespace, $m.name, .key, .value]
    | @tsv'
```

Output is a four-column TSV: namespace, Ingress name, annotation key, annotation value. Redirect it to a file and you have a starting inventory:

```
shop   checkout          nginx.ingress.kubernetes.io/rewrite-
target      /$2
shop   checkout          nginx.ingress.kubernetes.io/proxy-body-size
50m
shop   checkout-canary   nginx.ingress.kubernetes.io/canary
true
shop   checkout-canary   nginx.ingress.kubernetes.io/canary-weight
10
```

This is exactly the script in the companion repo — `scripts/inventory-annotations.sh` — so you don't need to retype it.

Surface 2: The global ConfigMap

Annotations only cover per-`Ingress` behaviour. `ingress-nginx` also ships a cluster-wide `ConfigMap` that applies to every route the controller serves, regardless of what any individual `Ingress` object says. Worth checking specifically:

- proxy timeouts and buffer sizes (`proxy-connect-timeout`, `proxy-buffer-size`, and similar)
- log format (`log-format-upstream`, `log-format-escape-json`)
- SSL/TLS settings (`ssl-protocols`, `ssl-ciphers`, `ssl-session-cache`)
- proxy-protocol handling (`use-proxy-protocol`)
- real-IP extraction (`enable-real-ip`, `use-forwarded-headers`, `proxy-real-ip-cidr`)
- custom headers (`add-headers`, `proxy-set-headers`)

> ⚠ **TCP/UDP services are a blind spot**
>
> `ingress-nginx` can also expose raw TCP and UDP services through two *separate* ConfigMaps (`tcp-services`, `udp-services`) — a mechanism that exists specifically because the Ingress API itself only supports HTTP(S). These are invisible to the annotation inventory

above; they don't show up on any `Ingress` object at all. If your cluster exposes anything this way, it's a common red-migration case — treat it as its own inventory item, not a footnote.

Surface 3: Controller flags

Some annotations only take effect if the controller was started with a matching command-line flag. The clearest example is `--enable-ssl-passthrough`, which defaults to `false`. The corresponding `ssl-passthrough` annotation has no effect at all unless that flag is set — and the flag itself carries a real cost: it works by intercepting all traffic on the HTTPS port and handing it to a local TCP proxy, bypassing NGINX's normal request handling with a non-negligible performance penalty. Check your controller's actual deployment args, not just the annotations on your `Ingress` objects — do not assume a given annotation is active just because it's set. Where a specific flag/annotation pairing isn't confirmed here, verify it against the current `ingress-nginx` documentation before relying on it; several such pairings exist beyond this one example.

The cargo-cult warning

An annotation being present in a Helm chart or an `Ingress` manifest does not mean it's load-bearing. Annotations get copied from one chart to the next, get left behind after the config that needed them changes, or get set once "to be safe" and never revisited. The right question isn't whether an annotation is set — it's whether removing or changing it changes observed behaviour. Presence is not evidence. That's also exactly what the companion repo's planned test suite is for: measuring observed behaviour before and after a migration, instead of trusting what an annotation's name implies.

Risk triage

Once you have the raw inventory, sort it into three buckets rather than treating every row as equally urgent:

- **Green** — likely portable, or a standard migration path already exists (e.g. a direct Gateway API equivalent).
- **Yellow** — implementation-specific policy that needs manual validation against whatever you migrate to; behaviour may differ subtly even where a nominal equivalent exists.
- **Red** — no safe equivalent, or a redesign is required (snippet annotations and TCP/UDP services both tend to land here).

Part 3 of this series builds the detailed annotation-by-annotation version of this triage. Don't overstate certainty at this stage — an item classified green here can still turn yellow once you check the actual value, not just the annotation's name.

Companion repo

The inventory script above, and the annotation matrix schema this inventory feeds into, live in the `ingress-nginx-migration-lab` companion repo on Codeberg. It's a skeleton for now — the inventory script is there, along with the planned matrix schema and test-suite layout — and grows into a reproducible lab as later parts of this series get published.

What's next

This inventory is the input to Part 3's annotation compatibility matrix: once you know which annotations, ConfigMap settings, and flags you actually depend on, the matrix tells you which of them translate cleanly to a given target and which don't. The series hub tracks what's published so far; Part 1 covers whether you need to act at all, if you haven't already read it.

Sources

- Annotations — kubernetes.github.io/ingress-nginx
- ConfigMaps — kubernetes.github.io/ingress-nginx
- Exposing TCP and UDP services — kubernetes.github.io/ingress-nginx
- Command line arguments — kubernetes.github.io/ingress-nginx
- SSL Passthrough — kubernetes.github.io/ingress-nginx