K3s + Cloudflare Tunnel Deployment Report

End-to-End Architecture, Setup, Errors, Fixes & Final Working State

Author: Sandesh Pulami

Date: November 28, 2025

---

1. Introduction

This report documents the complete setup and debugging process for deploying the *Explorage Backend Application* inside a **K3s Kubernetes cluster**, secured and publicly exposed using a **Cloudflare Tunnel**, without exposing any ports on the home router. It includes:

- Installation of K3s
- Deployment of a backend workload (Explorage)
- Creation of Kubernetes Service, Ingress & HPA
- Installation and configuration of Cloudflared (Tunnel client)
- Cloudflare DNS + Tunnel routing
- Detailed analysis of errors encountered
- Final working architecture & security overview

This report was created AFTER testing, debugging, and successfully getting:

👉 **https://explorage.pulami.co.uk** running through Cloudflare Tunnel → Traefik → Kubernetes → Explorage Pod.

---

2. Infrastructure Overview

2.1 Cluster Environment

- K3s running on **Ubuntu 22.04 ARM64** inside Multipass VM
- Traefik installed automatically by K3s
- Traefik exposed via LoadBalancer → NodePort fallback
- Cloudflared installed as a systemd service
- Cloudflare DNS + Tunnel integrated into K3s networking

VM Information:

- Name: `k3s-master`
- IP Address: `192.168.2.18`
- CPU: 2 cores
- RAM: 2GB
- Disk: 20GB

---

3. Backend Deployment Summary

3.1 Namespace Created

```
kubectl create namespace explorage
```

3.2 Secrets Applied

Secrets for MongoDB connection & session secret:

```
kubectl apply -f secret.yaml -n explorage
```

3.3 Deployment Applied

```
kubectl apply -f deployment.yaml -n explorage
```

Backend pod status:

```
explorage-backend-b56f9df5b-xxxx   Running
```

3.4 Service (ClusterIP)

```
NAME                    TYPE        CLUSTER-IP      PORT(S)
explorage-backend-svc   ClusterIP   10.43.126.249   8080/TCP
```

This ensures service is **internal-only**, accessible only through Traefik.

3.5 Horizontal Pod Autoscaler

```
kubectl apply -f hpa.yaml -n explorage
```

3.6 Ingress Resource

Mapped two hosts: - `explorage.pulami.co.uk` (public) - `explorage.local` (local debugging)

Status:

```
Host: explorage.pulami.co.uk
service: explorage-backend-svc:8080
```

4. Traefik Configuration (Auto from K3s)

Traefik service discovered as:

```
traefik          LoadBalancer   10.43.221.108   192.168.2.18
80:32234/TCP,443:30787/TCP
```

Important mapping:

- **HTTP** → NodePort **32234**
- **HTTPS** → NodePort **30787**

This is critical for Cloudflare Tunnel.

5. Cloudflared Installation & Tunnel Setup

Cloudflared was installed using ARM package:

```
wget https://.../cloudflared-linux-arm64.deb
sudo dpkg -i cloudflared.deb
```

Login to Cloudflare:

```
cloudflared tunnel login
```

Create Tunnel:

```
cloudflared tunnel create explorage-tunnel
```

Tunnel ID:

```
ea050b77-e979-4b40-9ac3-929480a399af
```

Credentials File Discovered At:

```
/home/ubuntu/.cloudflared/ea050b77-...json
```

6. Cloudflare Tunnel Configuration

Final **correct** `/etc/cloudflared/config.yml`:

```
tunnel: ea050b77-e979-4b40-9ac3-929480a399af
credentials-file: /home/ubuntu/.cloudflared/ea050b77-e979-4b40-9ac3-
929480a399af.json
```

```
ingress:
  - hostname: explorage.pulami.co.uk
    service: http://192.168.2.18:32234
  - service: http_status:404
```

Why port 32234?

- Cloudflared sends **HTTP** traffic
- Traefik HTTPS port (30787) would reject HTTP (needs TLS)
- Traefik HTTP NodePort (32234) matches cloudflared tunnel expectations

Result: **Perfect routing**

---

7. Cloudflare DNS Configuration

DNS record required:

```
Type: CNAME
Name: explorage
Target: ea050b77-e979-4b40-9ac3-929480a399af.cfargotunnel.com
Proxy: ON
```

Final `dig` result:

```
explorage.pulami.co.uk.  IN A   104.21.31.87
explorage.pulami.co.uk.  IN A   172.67.175.193
```

This shows Cloudflare Anycast is routing properly.

---

8. Error Summary & Fixes

Error 1: cloudflared.service failing to start

CAUSE: Wrong credentials file path - Cloudflared stored creds at `/home/ubuntu/...` - Config pointed to `/root/...` → Service crash

FIX: Correct path in `config.yml`

Error 2: NXDOMAIN (DNS_PROBE_FINISHED_NXDOMAIN)

CAUSE: DNS record existed in UI but not published

FIX: Delete + recreate DNS record and run

```
cloudflared tunnel route dns explorage-tunnel explorage.pulami.co.uk
```

Error 3: 502 Bad Gateway

CAUSE: Tunnel attempted to forward to internal service DNS:

```
service: http://traefik.kube-system.svc.cluster.local:80
```

This fails because cloudflared runs *outside* the cluster.

FIX: Use VM IP + NodePort

```
service: http://192.168.2.18:32234
```

---

9. Final Working Architecture Diagram

```
Browser (HTTPS)
        ↓
Cloudflare Edge (SSL termination)
        ↓
Cloudflare Tunnel (Encrypted)
        ↓  HTTP
Cloudflared on VM → http://192.168.2.18:32234
        ↓
Traefik (HTTP EntryPoint)
        ↓
Kubernetes Ingress
        ↓
explorage-backend-svc (ClusterIP:8080)
        ↓
Pod (Express backend)
```

---

10. Security Verification

✓ No inbound ports exposed on router

✓ Secure HTTPS via Cloudflare Edge

✓ Cloudflared uses outbound-only tunnel

✓ Traefik routes only known hostnames

✓ Backend isolated with ClusterIP

✓ Systemd-managed cloudflared runs continuously

This setup is **safe, reliable, and production-grade**.

---

11. Final Confirmation

After all fixes: - Tunnel: **active & healthy** - DNS: **propagated & valid** - Traefik: **routing correctly** - Ingress: **matching hostname** - Backend: **responding at /listings**

★ FINAL RESULT:

👉 **https://explorage.pulami.co.uk — LIVE & SECURE**

Conclusion

This report captures the full setup of your K3s + Cloudflare Tunnel deployment, including: - Infrastructure setup - Routing - Debugging - Final architecture - Security verification

You're now running a fully secure homelab-based production environment using **Kubernetes, Traefik, Cloudflare Tunnel, and Node.js**.