Cilium + eBPF Day

NORTH AMERICA

Live Migrating Production Clusters From Calico to Cilium

Cilium + eBPF Day
NORTH AMERICA

November 12, 2024
Salt Lake City

#Open To Work

Moh Ahmed
Staff Developer, SRE
SamsungAds

#Open To Work

Raymond Maika
Manager, Platform Engineering
SamsungAds

Agenda

Objective

Purpose

Different Migration Methods

Running the Migration

Demo

Closing Thoughts

Objective

- Calico is in-use across ~20 clusters in use by tenants
  - More than half are running production workload
  - Sizes vary from 10s to >500 nodes
- We want to swap to Cilium
  - Gateway API support
  - Light service-mesh features
  - Hubble for network visibility
- How do we minimize impact of this change and keep revenue-generating services running?
  - Secondary - limit effort required of tenants in particular

Purpose

- Swap to Cilium for the following improvements
  - eBPF forwarding, XDP datapath
  - Simplified network policies using Cilium nodePort implementation
  - Improved self-service capabilities and auditability for policies
  - New features
    - Gateway API
    - Enhanced load-balancing (Cilium service implementation)
    - Lightweight service mesh features

Different Migration Methods

1. Deploy new clusters that come freshly made with Cilium
2. Rip out the old, deploy the new
3. Bind multiple network interfaces to a pod
4. Attempt a hybrid, per-node migration

Deploy new clusters

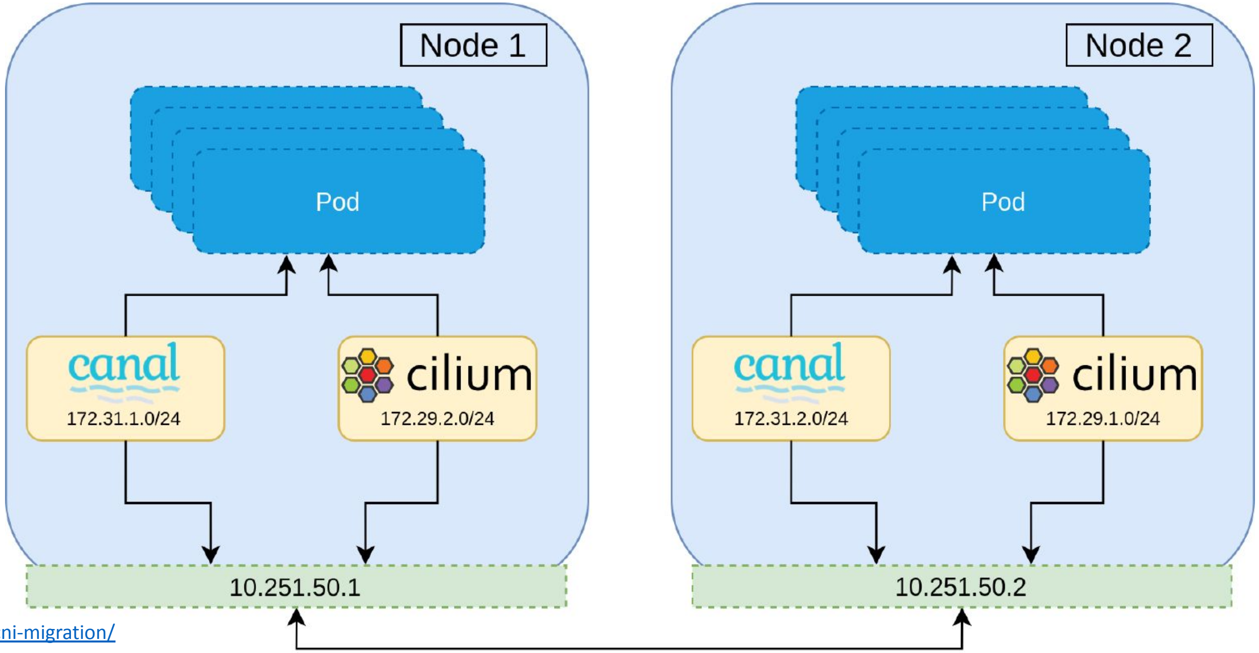
- Deploy a new cluster with Cilium as the default CNI
- This removes the dependency on legacy components
- Problems:
    - More work on the user side to migrate their workload and data
    - Stretches out the amount of time needed to complete the work
    - A long period of time where we're running two different networking and security solutions
- New clusters will be provisioned with Cilium, but can't clean up old ones until **everyone** migrates

Rip out the old, deploy the new

- Uninstall the old CNI plugin, install the new one
- Problems:
  - Disrupts the existing workloads
  - Cluster-wide maintenance that can last a long time
  - Hard to revert back since it’s an all or nothing approach
- We set out to find a solution that we can apply in a controlled way

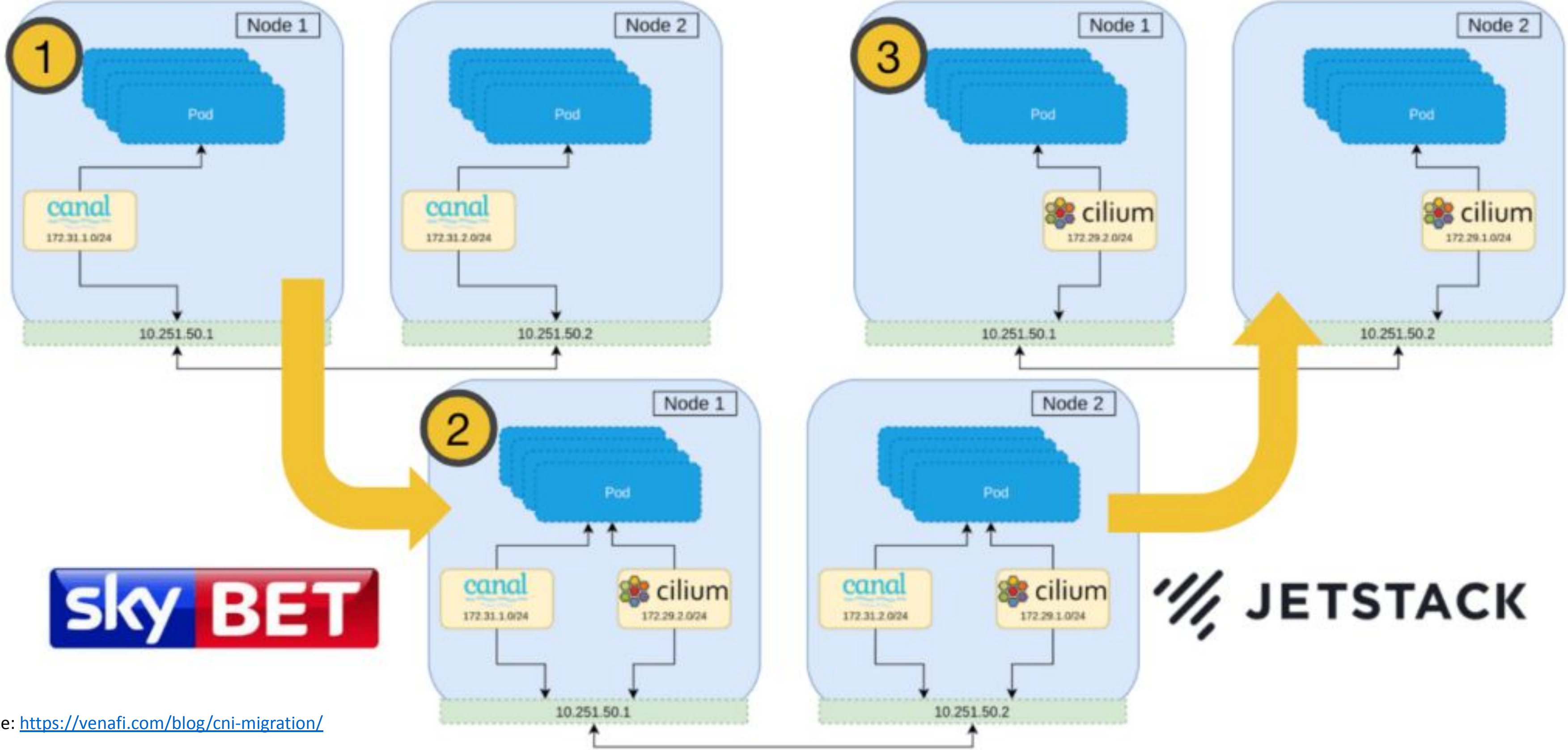
Bind multiple network interfaces to a pod

- This idea came from a blogpost from Jetstack (now Venafi)
- Using Multus, we should be able to have multiple network interfaces in a pod



source: https://venafi.com/blog/cni-migration/

Bind multiple network interfaces to a pod



source: https://venafi.com/blog/cni-migration/

Bind multiple network interfaces to a pod

- But we couldn’t get it to work:
    - Multus was installed and was set to be the default CNI
    - Pods had both interfaces configured (one interface serving Calico and the other serving Cilium)
    - Calico was set to be the primary CNI
- However, we couldn’t get pods to talk to each other on either interfaces
- There would be a period of time where workloads were unreachable
- We also tried using the source based routing (SBR) meta plugin but saw:
    - If SBR was enabled, only Cilium interfaces worked
    - If SBR was disabled, only Calico interfaces worked

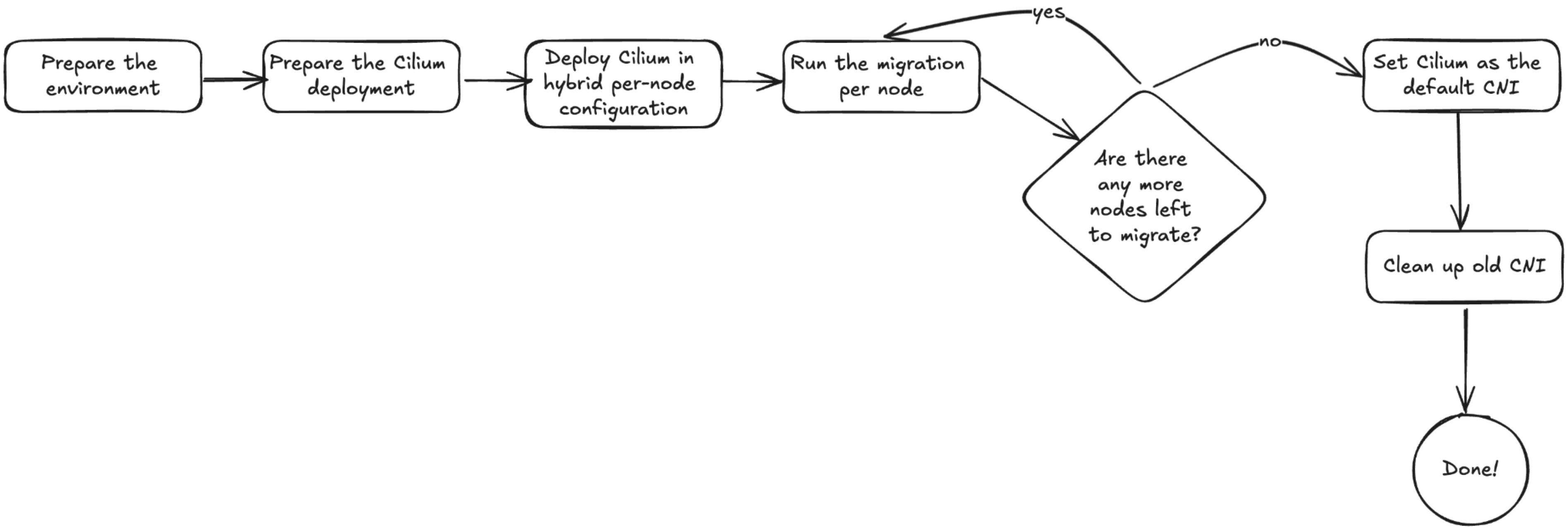
We learned a lot

- We had a valid and reusable rollback strategy for when things didn’t work
- We updated our Rancher cluster agents to use `hostNetwork` to maintain connection while the CNI was being upgraded
- We understood CNI configuration in much more detail (`/etc/cni.d/`)
- We learned about Cilium configuration taking over CNI path unless specifically disabled (`--cni-exclusive=false`)

Hybrid, per-node migration

- We took advantage of a new Cilium feature for per-node configuration
  - Allows for setting up a specific Cilium configuration on a per-node basis, using node labels
- With this, we can have a CNI configuration that uses Calico as a default CNI
- Once the node is labeled, Cilium takes over the CNI directory and becomes the default CNI
- Whitelisting the two pod network CIDRs in both CNI firewalls allows for open communication between the two

Running the Migration



Prepare the environment

Prepare the Cilium deployment

Deploy Cilium in hybrid per-node configuration

Run the migration per node

Are there any more nodes left to migrate?

yes

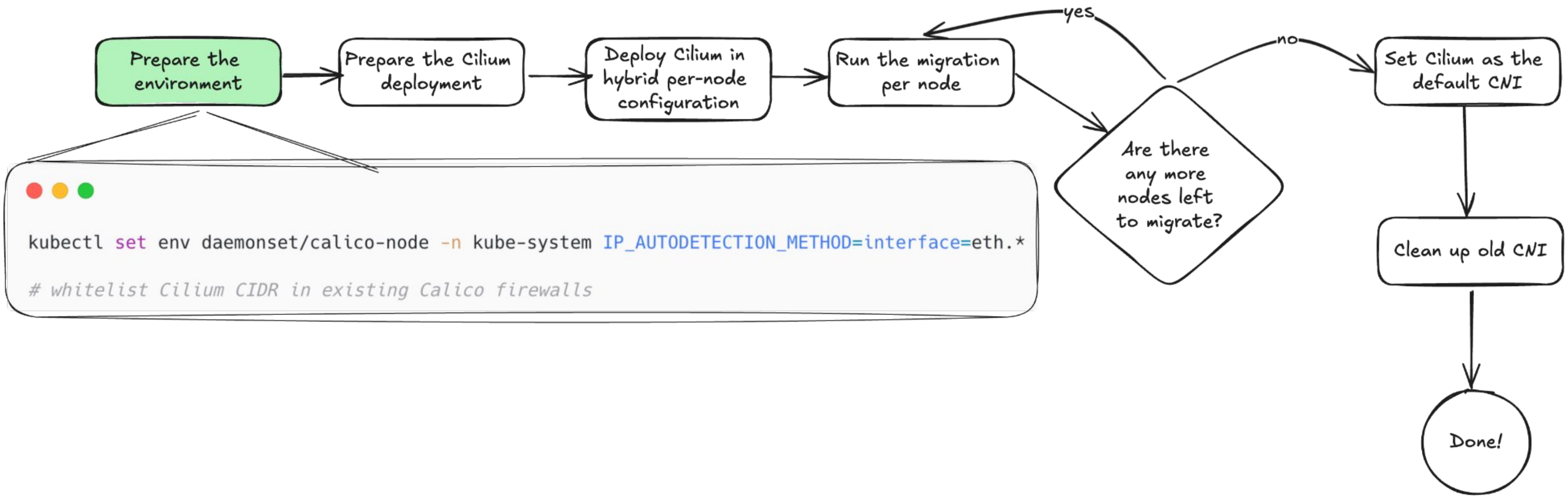
no

Set Cilium as the default CNI

Clean up old CNI

Done!

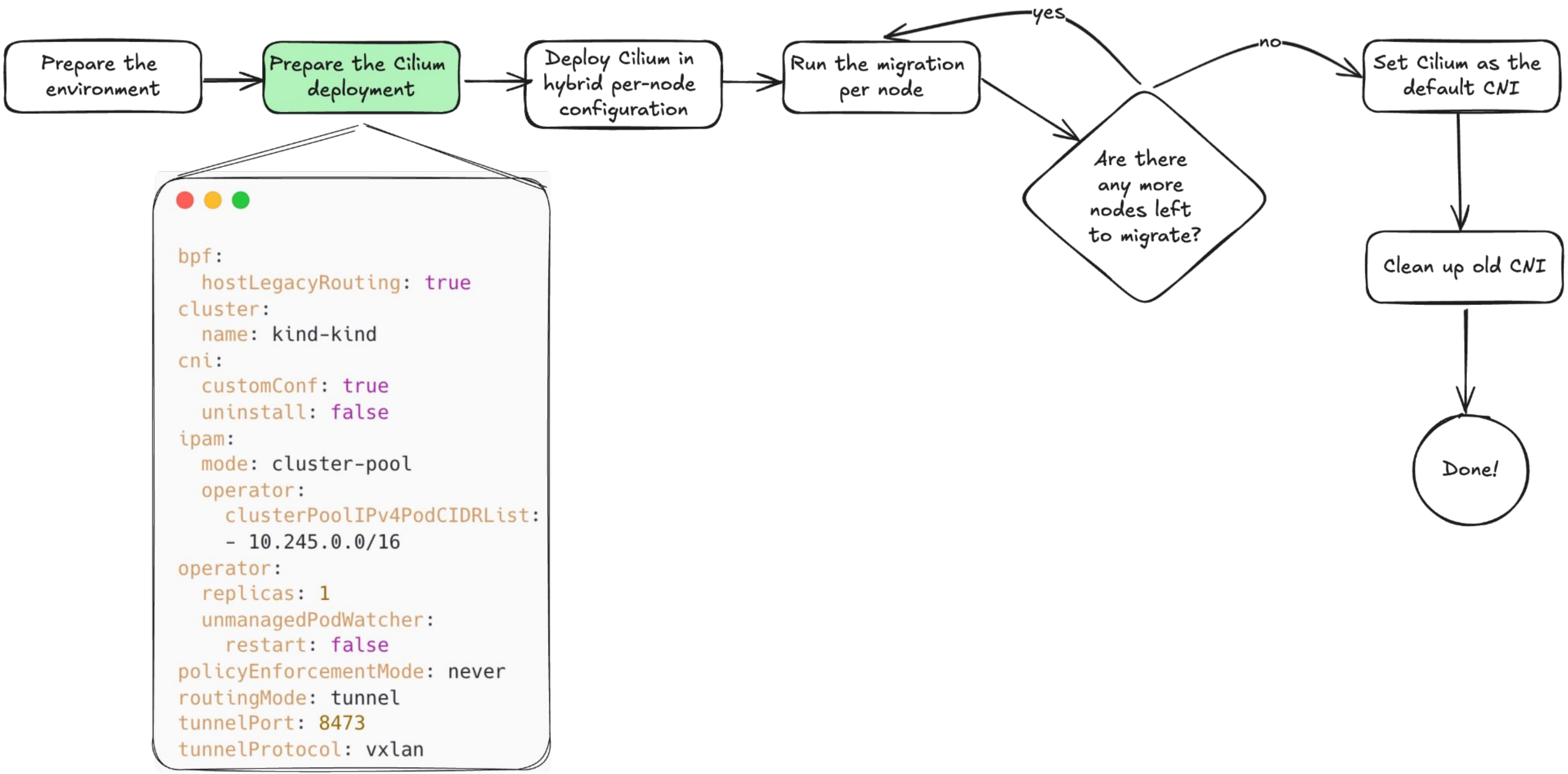
Running the Migration


Prepare the environment
Prepare the Cilium deployment
Deploy Cilium in hybrid per-node configuration
Run the migration per node
Are there any more nodes left to migrate?
yes
no
Set Cilium as the default CNI
Clean up old CNI
Done!

```
kubectl set env daemonset/calico-node -n kube-system IP_AUTODETECTION_METHOD=interface=eth.*

# whitelist Cilium CIDR in existing Calico firewalls
```

Running the Migration



Prepare the environment

Prepare the Cilium deployment

Deploy Cilium in hybrid per-node configuration

Run the migration per node

Are there any more nodes left to migrate?

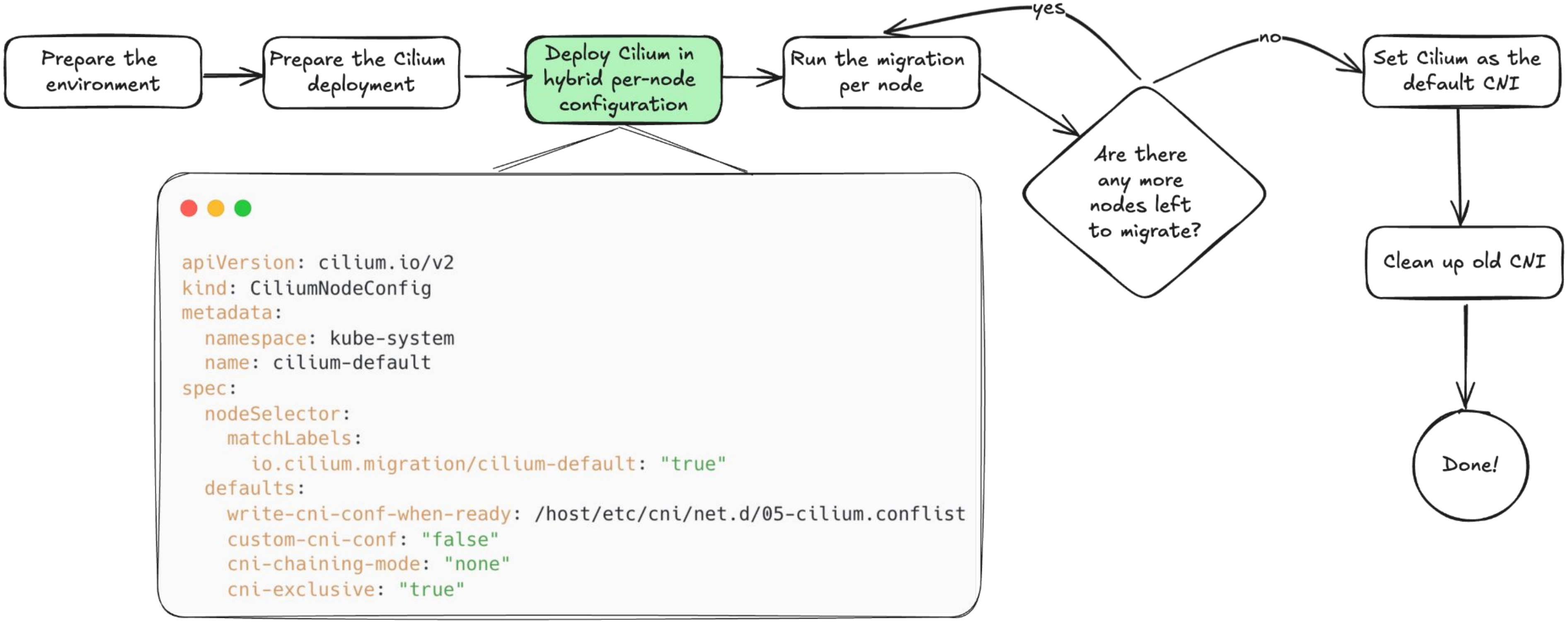
Set Cilium as the default CNI

Clean up old CNI

Done!

```yaml
bpf:
  hostLegacyRouting: true
cluster:
  name: kind-kind
cni:
  customConf: true
  uninstall: false
ipam:
  mode: cluster-pool
  operator:
    clusterPoolIPv4PodCIDRList:
    - 10.245.0.0/16
operator:
  replicas: 1
  unmanagedPodWatcher:
    restart: false
policyEnforcementMode: never
routingMode: tunnel
tunnelPort: 8473
tunnelProtocol: vxlan
```

Running the Migration



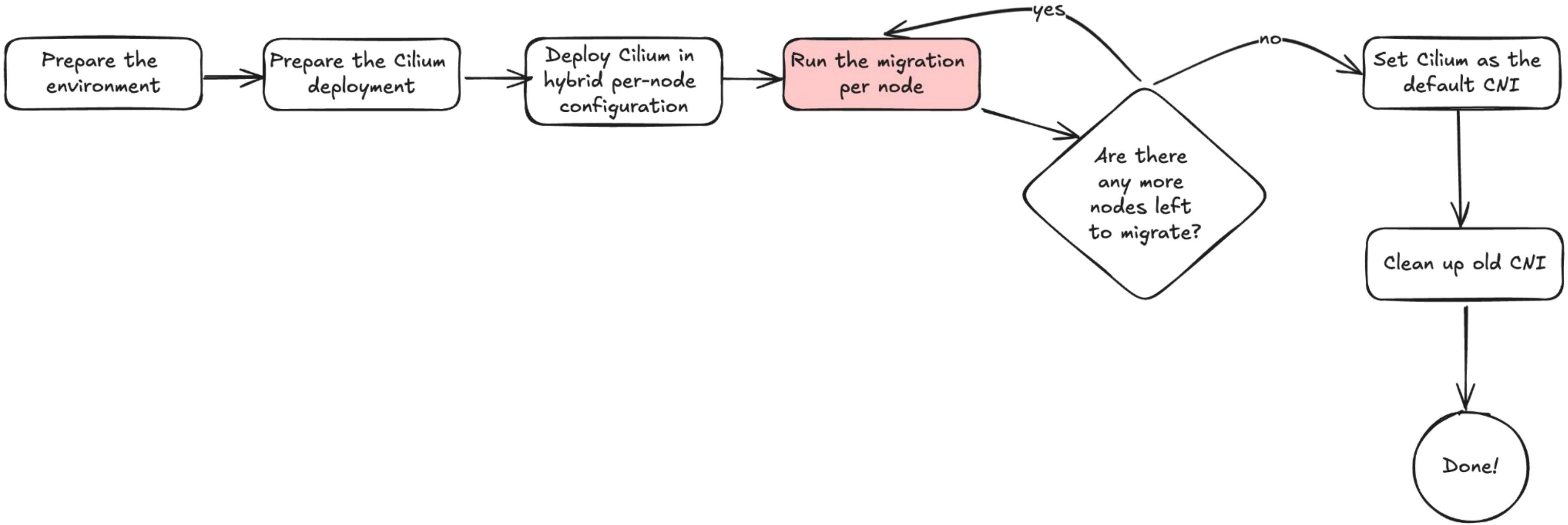
Prepare the environment → Prepare the Cilium deployment → Deploy Cilium in hybrid per-node configuration → Run the migration per node → Are there any more nodes left to migrate?

yes → Run the migration per node

no → Set Cilium as the default CNI → Clean up old CNI → Done!

```
apiVersion: cilium.io/v2
kind: CiliumNodeConfig
metadata:
  namespace: kube-system
  name: cilium-default
spec:
  nodeSelector:
    matchLabels:
      io.cilium.migration/cilium-default: "true"
  defaults:
    write-cni-conf-when-ready: /host/etc/cni/net.d/05-cilium.conflist
    custom-cni-conf: "false"
    cni-chaining-mode: "none"
    cni-exclusive: "true"
```

Running the Migration



Prepare the environment

Prepare the Cilium deployment

Deploy Cilium in hybrid per-node configuration

Run the migration per node

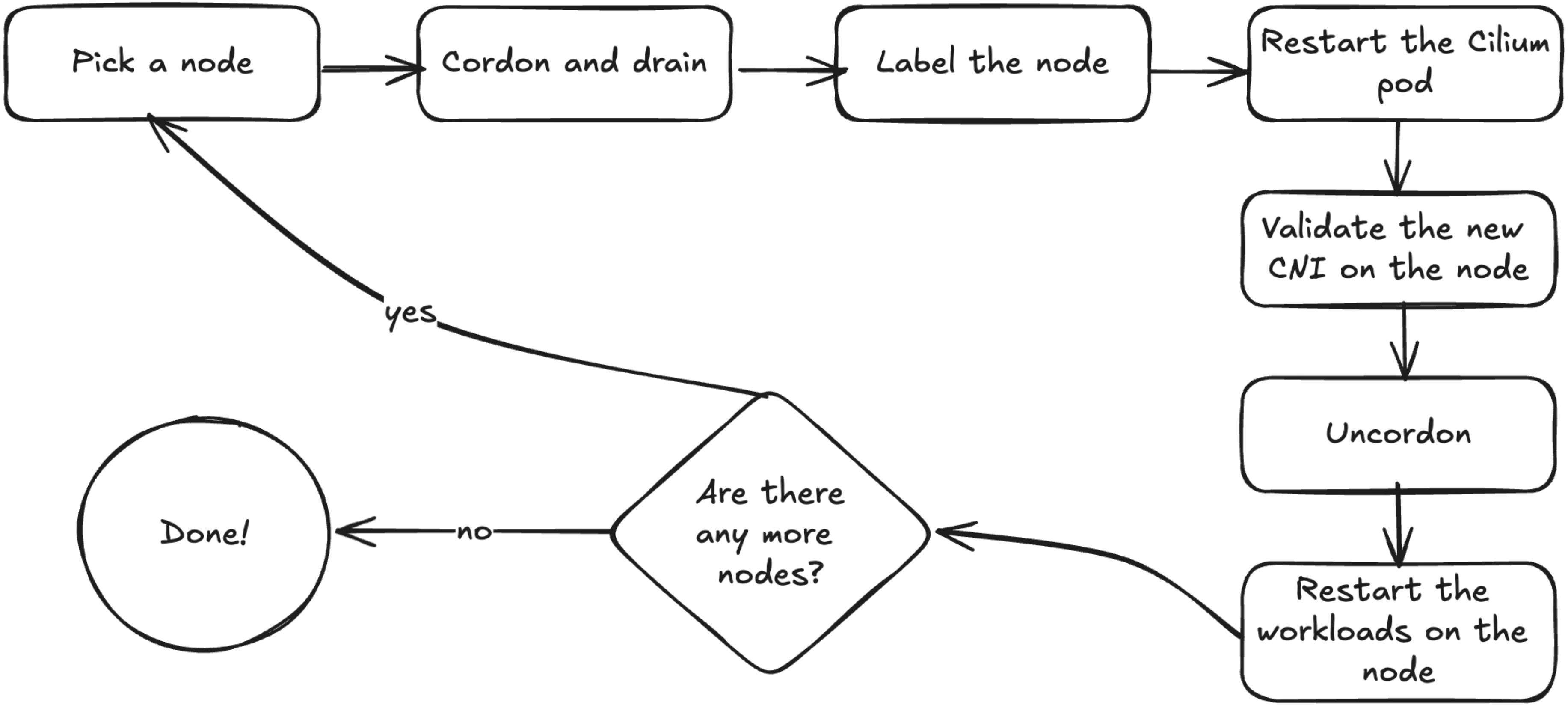
Are there any more nodes left to migrate?

Set Cilium as the default CNI

Clean up old CNI

Done!

Run the migration per node



Pick a node

Cordon and drain

Label the node

Restart the Cilium pod

Validate the new CNI on the node

Uncordon

Restart the workloads on the node

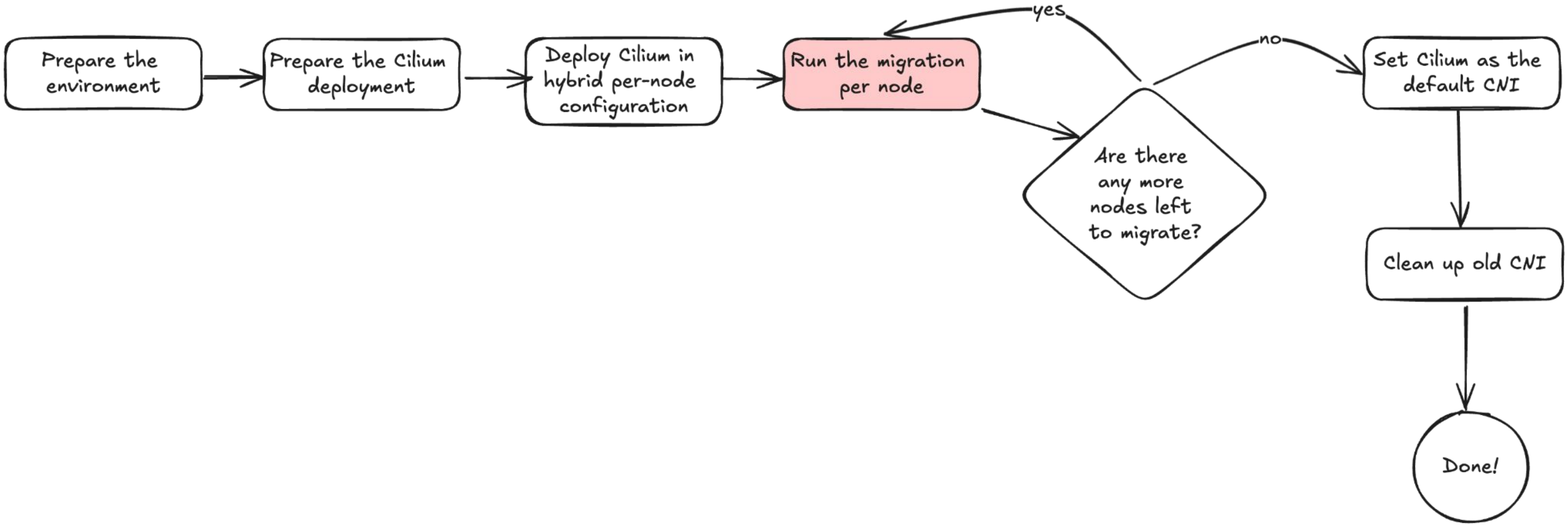
Are there any more nodes?

yes

no

Done!

Completing the migration



Prepare the environment

Prepare the Cilium deployment

Deploy Cilium in hybrid per-node configuration

Run the migration per node

Are there any more nodes left to migrate?

yes

no

Set Cilium as the default CNI

Clean up old CNI

Done!

Cilium + eBPF Day NORTH AMERICA

Demo

Some things to consider

- Our clusters are running IPv4 only, your mileage may vary for IPv6
- We were running an old version of Calico (v3.x), some extra clean up was required
- We migrated to Cilium v1.13.x at the time
- We didn’t enable Cilium’s kube-proxy replacement at the time of the migration to reduce complexity
- We ended up using Portmap (`hostPort`) CNI chaining as we had an application using a host port
  - This was needed while the migration was happening until we could enable Cilium’s kube-proxy replacement post-migration
- More details available on the blog post we published on this

Closing thoughts

- We migrated ~20 clusters (development, staging, production)
- Some clusters as small as 10 nodes, many with hundreds of nodes
- Number of PRs needed to deploy a service with firewalling was reduced by 50%
- New firewall rules are more readable and are based on identities rather than ports
- `CiliumNodeConfig` is a very useful CRD
    - Re-used this to rollout our kube-proxy replacement configs
- Thanks to strong best practices in the org, our critical systems have minimum levels of fault tolerance, allowing this migration to complete with no service interruptions

Thanks to

This was a team effort, with the migrations and troubleshooting done by everyone on the team, especially on large clusters

- Alexander Ratte
- Alexis Vanier
- Benoit Caron
- Thibaut Charry
- Yann David
- Vlad Paciu

Please rate and provide feedback