The most complicated comparison I’ve ever attempted. From a 10,000 ft perspective it should be simple. We have CNI as a common standard in Kubernetes and all these plugins need to do is assign an IP address to each pod so they can talk to each other, both on the same host, and across hosts.

As you can see from the table the devil is in the detail. In this blog we’ll examine the key differences between Flannel, Calico, Weave, Cilium, Kube Router, Romana and Contiv.



[You can view and edit the Google sheet here.](https://docs.google.com/spreadsheets/d/1qCOlor16Wp5mHd6MQxB5gUEQILnijyDLIExEpqmee2k/edit?usp=sharing)

**Update:** the Google sheet now also includes Tungsten Fabric, kopeio and amazon-vpc-cni-k8s.

Unfortunately, I wasn’t clever enough to figure out a way to compare combinations of features directly across all plugins. Depending on the plugin, and depending on your underlying network, you will configure things differently. We would need a column for each variation of settings and I think that may be overkill. What we’ll do instead is simplify by grouping plugins by type and then I’ll attempt to give my opinion on the key differentiators.

Before we do that let’s take a segway into some background information. I was interested in what each cloud Kubernetes service used, as well as defaults for the On-Prem Kubernetes distributions.

| Service / Distro | Network |
| --- | --- |
| No data available in table | |

When you provision a Kubernetes cluster with GKE, EKS or AKS the network just works. Each of the cloud providers have their own CNI plugins. AWS have open sourced their plugin which is quite a friendly thing to do. You may notice that all 3 clouds are adopting Calico specifically to handle network policy.

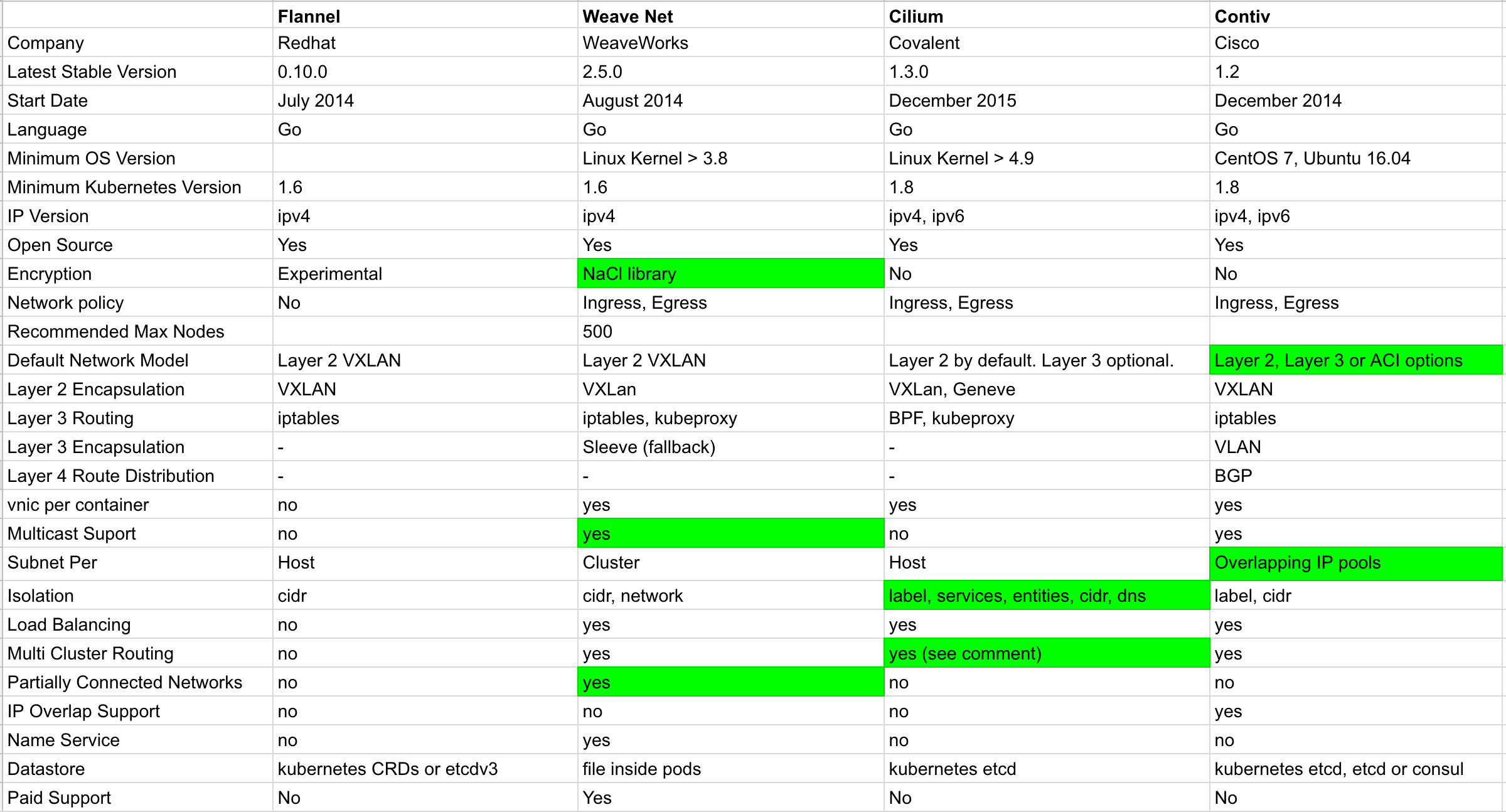
Amongst the On-Prem distributions Calico again reigns supreme. This time for both networking and network policy. OpenShift and Cisco have their own plugins as you would expect.

**Layer 2 vs Layer 3**

This is something that you’ll need to decide first. Layer 3 is more desirable but often the world is not fair and there are circumstances where you will be forced to use a Layer 2 solution.

**Layer 2**

Let’s examine the Layer 2 solutions.



Flannel, Weave, Cilium and Contiv can all run in Layer 2 mode which means they create a VXLAN that encapsulates all traffic between hosts. Encapsulation adds a cost in terms of performance. However, most modern network cards will offload this so perhaps it’s not an issue.

You’ll often use a Layer 2 solution to cover over deficiencies in the underlying network. For example maybe you have address space limitations. In the case of Weave, Cilium and Contiv they additionally provide Layer 3 functionality as an all in one solution.

This is where it gets tricky. You can mix and match plugins and use combinations together. Technically you could run Flannel as a Layer 2 network under many of the Layer 3 options in this comparison. Let’s assume that you want to use the fewest number of plugins for the sake of sanity.

Flannel is the oldest and arguably most mature plugin but it has the fewest features. It’s really common for people to combine Flannel and Calico together into what used to be called ‘Canal’. It seems the Canal project has died and both Flannel and Calico develop separately but maintain good documentation for combining together.

Weave has an encryption library baked in and has some unique fast data path features. The other stand-out feature for Weave is the support for partially connected networks. All Weave nodes operate as a mesh and so you only need a route between one other connected segment to achieve full connectivity. It’s also the only plugin with built in name resolution which can be used for load balancing services. The fact that Weave operates as a full mesh is both a massive strength and a weakness. Past a few hundred nodes the overhead of distributing routes becomes a bottleneck and you then need to disable auto-discovery and start to manually manage connections.

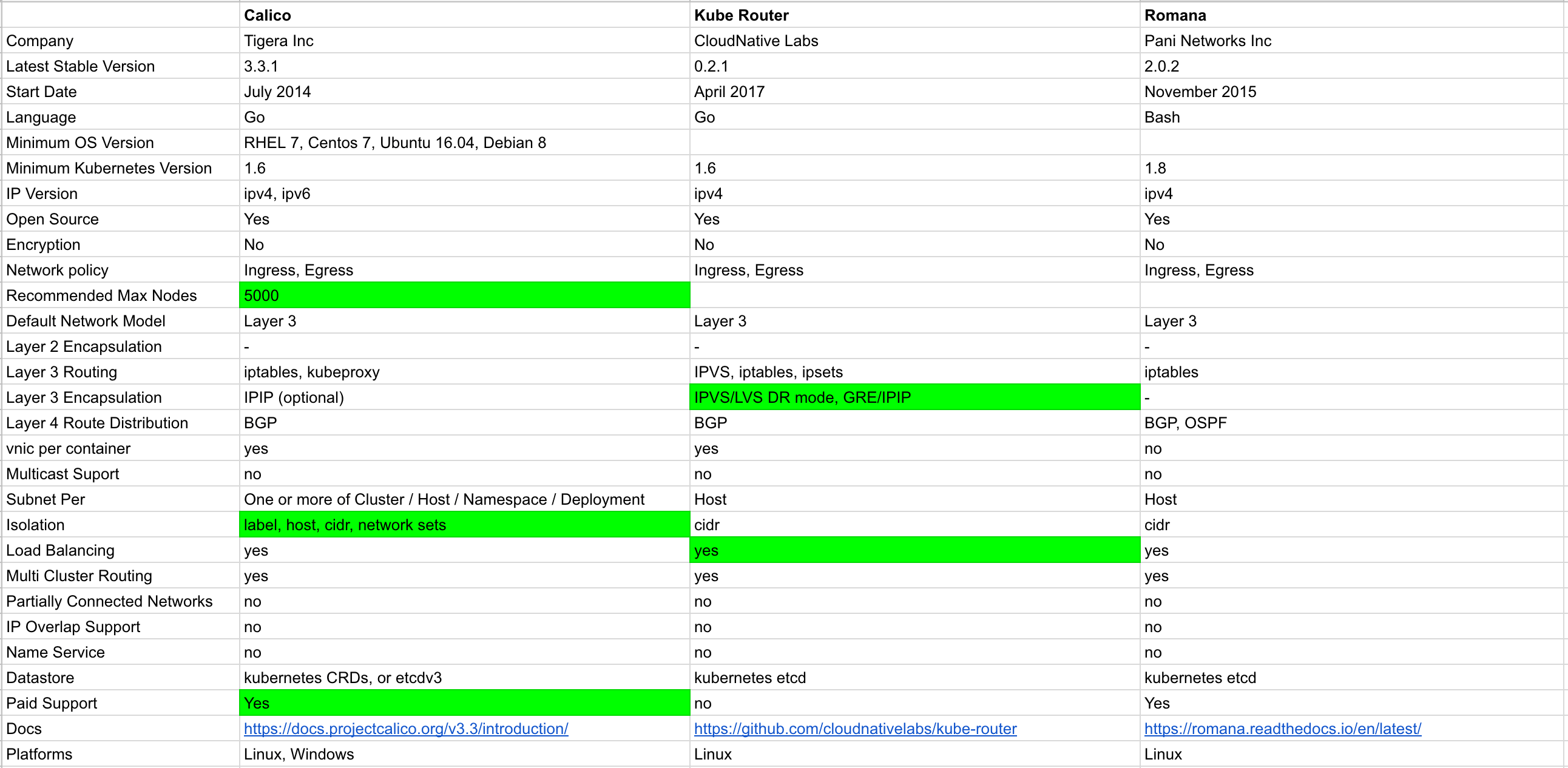
Cilium differentiates itself by providing Layer 7 policies and using BPF to process the rules inside the kernel which should scale a lot better than iptables rules. The only down side is it requires quite a new kernel version. I quite like Cilium and I’ve received a number of comments from people who are using it happily in production.

Contiv has many of the same features as the other options but also has some funky stuff that allows overlapping IP’s, for if you need multiple pods to share the same IP address. If you have a significant On-Prem Cisco investment then this is probably one to trial.

**Layer 3**

Let’s say you don’t need Layer 2 functionality at all and you’d prefer a pure Layer 3 plugin. For many this will absolutely be the case. Layer 2 VXLAN’s are a bit of a black box when trying to debug. Layer 3 plugins on the other hand allow you to quickly print a routing table on an instance to view what’s going on.

The pure Layer 3 plugins are Calico, Kube Router and Romana. Although, remember, you can also make Cilium and Contiv Layer 3 only by changing settings but we’ve covered those already.



Calico is the oldest and most mature. I don’t have any hard facts to substantiate this assumption, but, I think Calico is probably the most popular plugin overall. The guys from Tigera on the Sig-Network channel on Kubernetes Slack are awesomely responsive and left me feeling that paid support would be worth every penny.

Kube Router is a relatively new project. It’s simple and uses the IPVS/LVS kernel feature to speed up load balancing and routing. It also has direct server return which is supposed to decrease latency.

Romana aims to achieve host level network performance by working without any encapsulation at all. Calico and Kube Router both encapsulate traffic at Layer 3 (usually with IPIP) in order to route traffic between subnets which comes at a slight performance cost. The only negative thing I saw with Romana is lack of community.

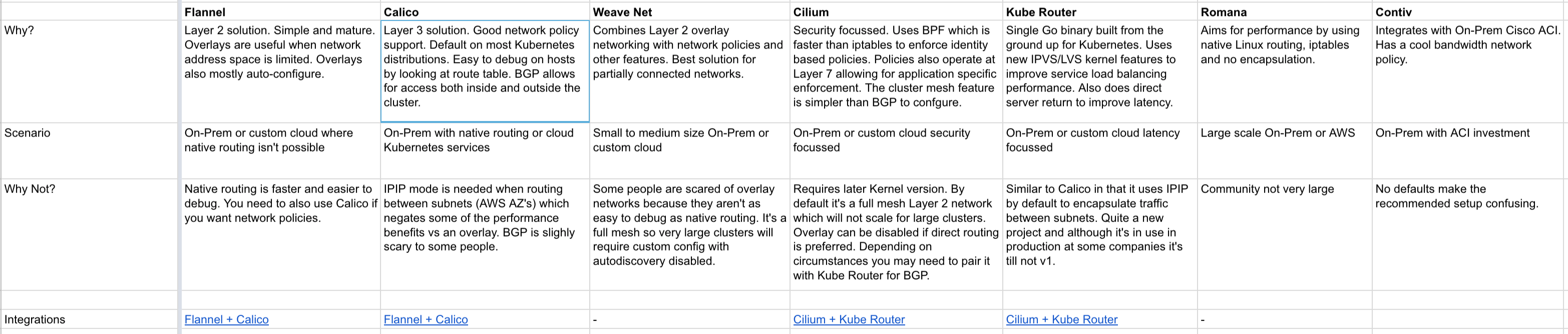
**Recommendations**

Let’s start with the simplest first. If you’re using a cloud service like GKE, EKS or AKS you are covered already and should simply use their Calico integration for network policy.

For OpenShift, PKS or Cisco use their built in SDN.

What about custom cloud? For example Rancher, Kops or just Kubeadm on AWS. Here’s my thought process: start with Calico and only deviate if you need something that it does not provide.

Custom On-Prem is where things get trickier. Here’s a quick table that shows how I think each options deviate.



Again, I’d start with Calico as the default choice and work out if you need any other features. If you are extremely performance sensitive I’d benchmark Calico vs Kube Router and Romana. Each of those have features that should reduce latency, or at least remove plugin performance overhead.

I think Calico has become the default choice for many because of the way it uses BGP. Networks can be extremely difficult to troubleshoot and calico makes it relatively easy. Lots of people look at the choices and say ‘oh I can peer it into my normal network easily’. They rarely do, but thats’s still a big plus when you’re evaluating something.

If security is your main concern than I’d trial Calico vs Cilium. I also read a blog that showed [Cilium may be a very good choice for networking pods across Kubernetes clusters](https://medium.com/@olemarkus/kubernetes-clustermesh-with-cilium-and-kops-7a8670551725) without the pain of complicated BGP configuration.

If you need something to cover over your On-Prem network deficiencies then Weave is probably a good choice.

To summarise my recommendation is Calico 9 times out of 10. It’s simple, fast, has lots of features and a good company behind it.

As ever, if I have made an error please let me know and I’ll update this post. I’m interested to get feedback on what network plugins you are using and what made you choose it.