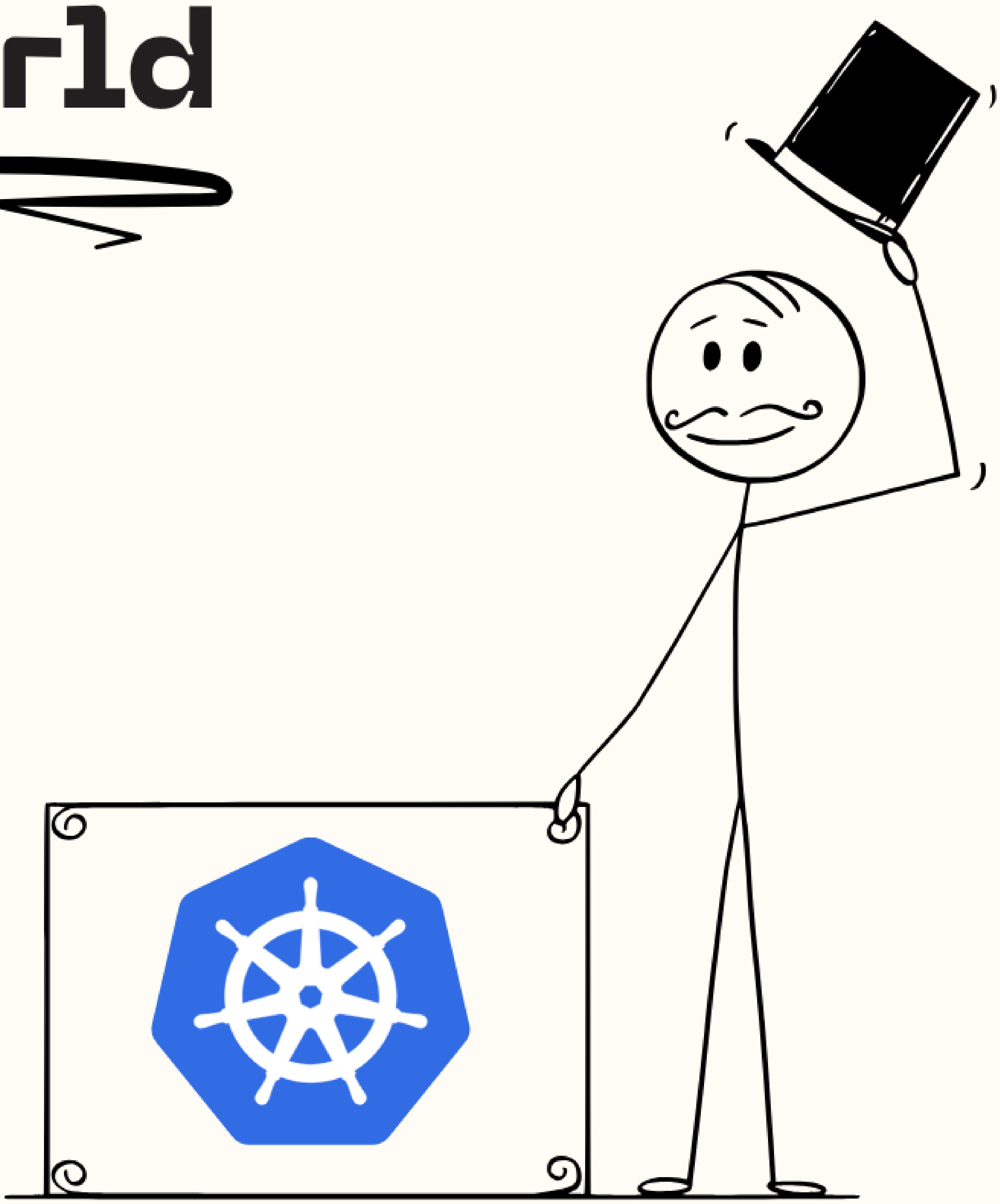
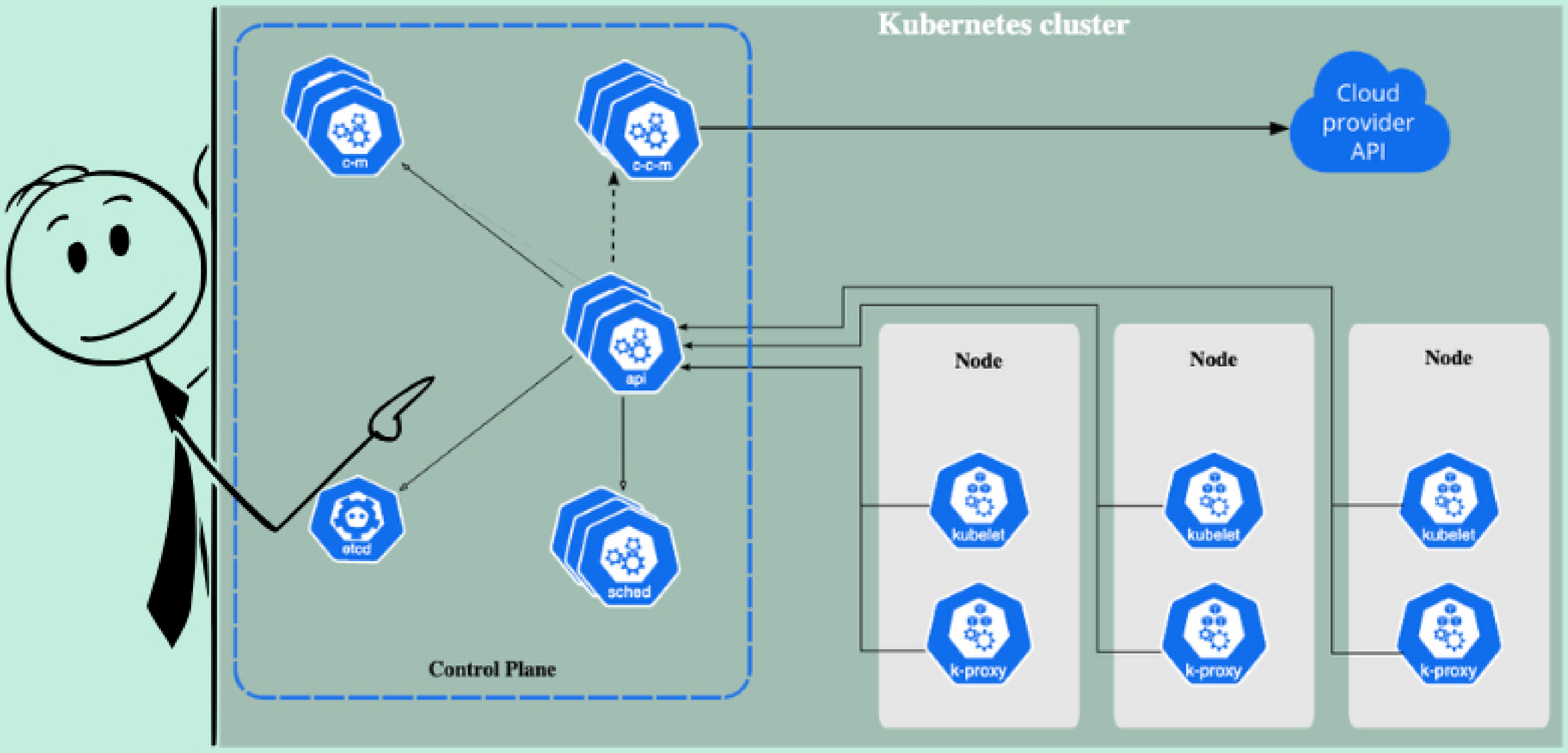
Troubleshooting in the Kubernetes World



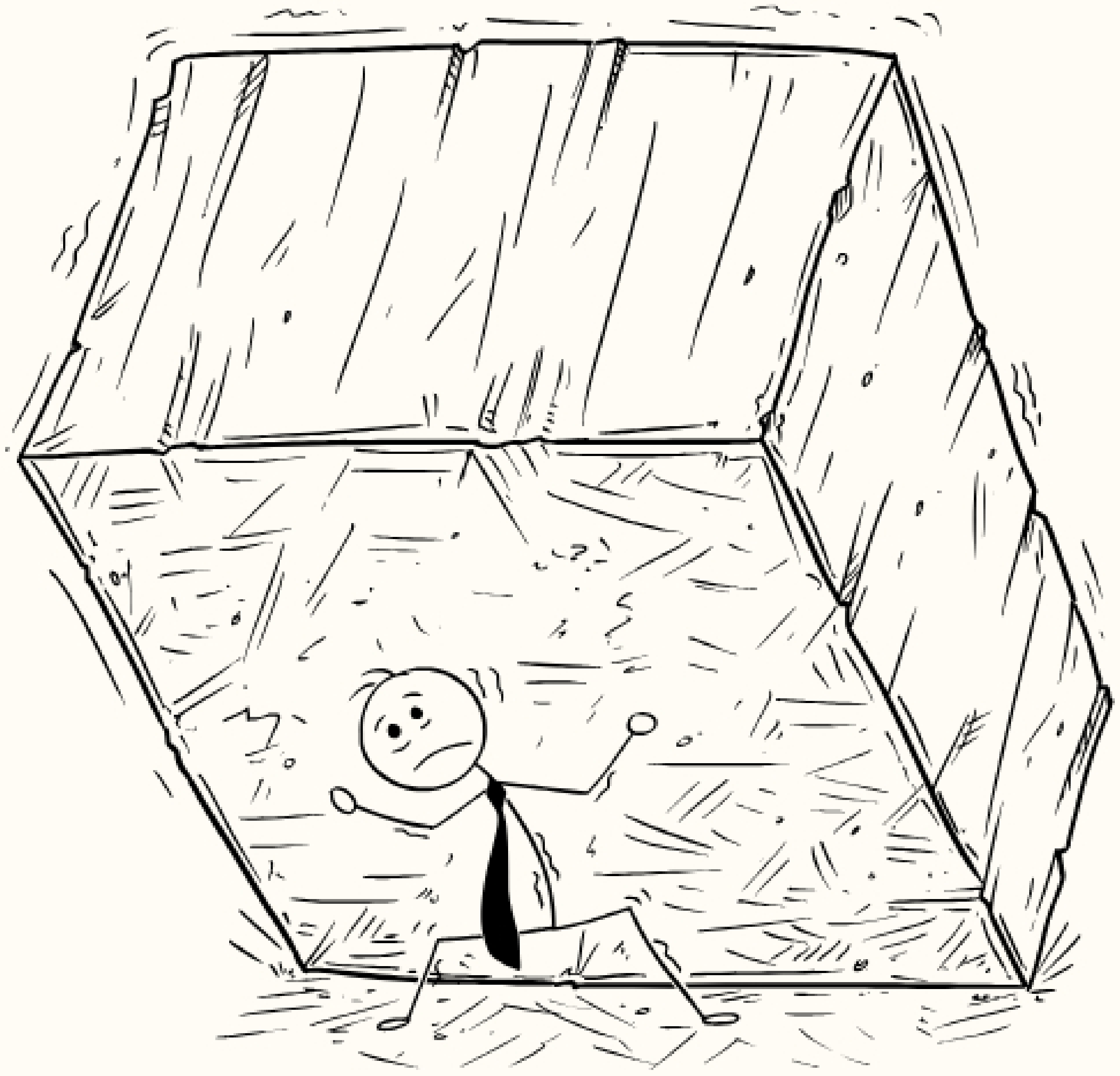
Kubernetes troubleshooting would be easy if K8s were a straightforward, uncomplicated system.

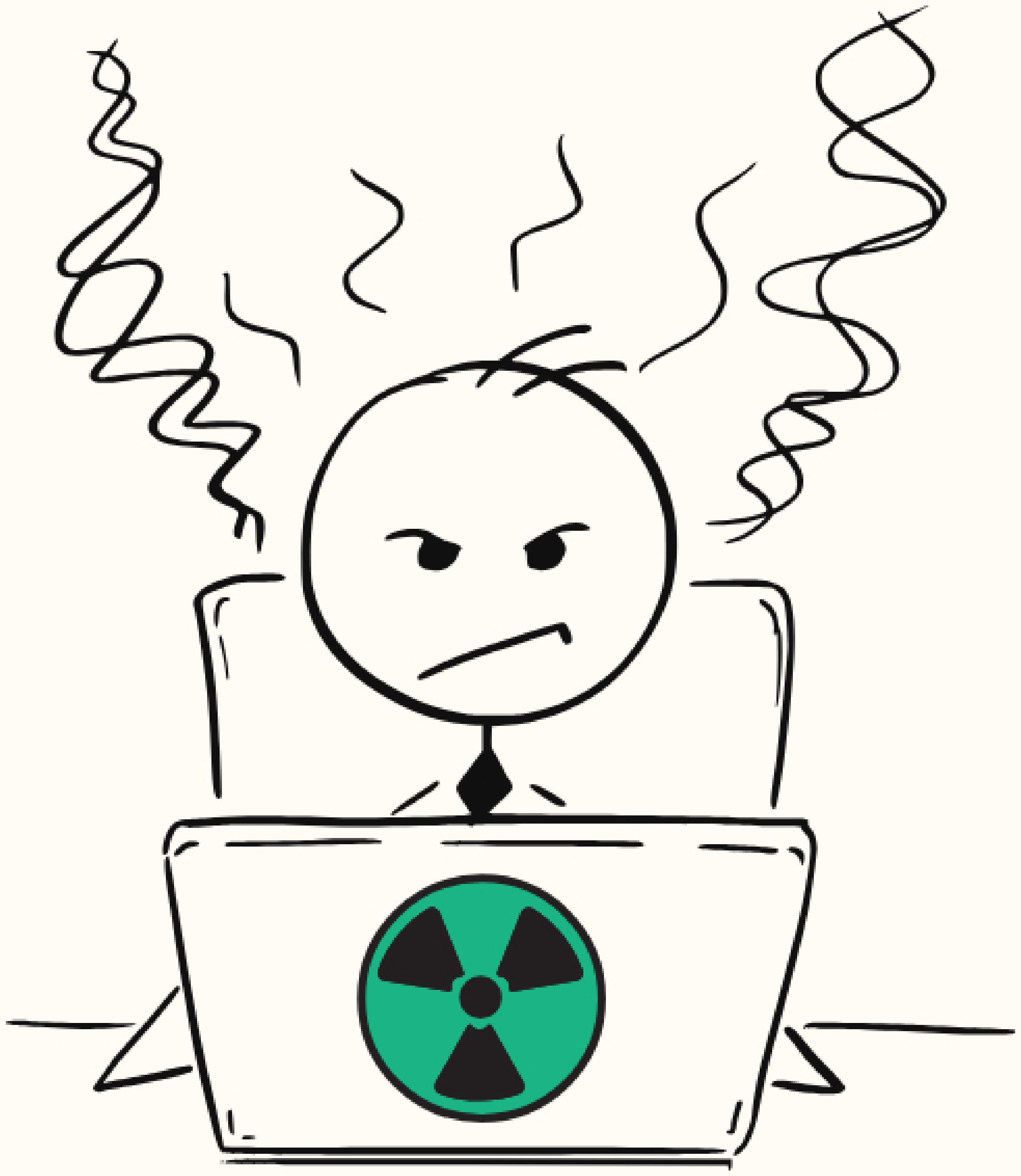


Unfortunately, it's not.

Kubernetes is a very complex platform that includes a variety of distinct components – an API server, an etcd key-value store, control plane nodes, worker nodes, Pods, various network resources and more.

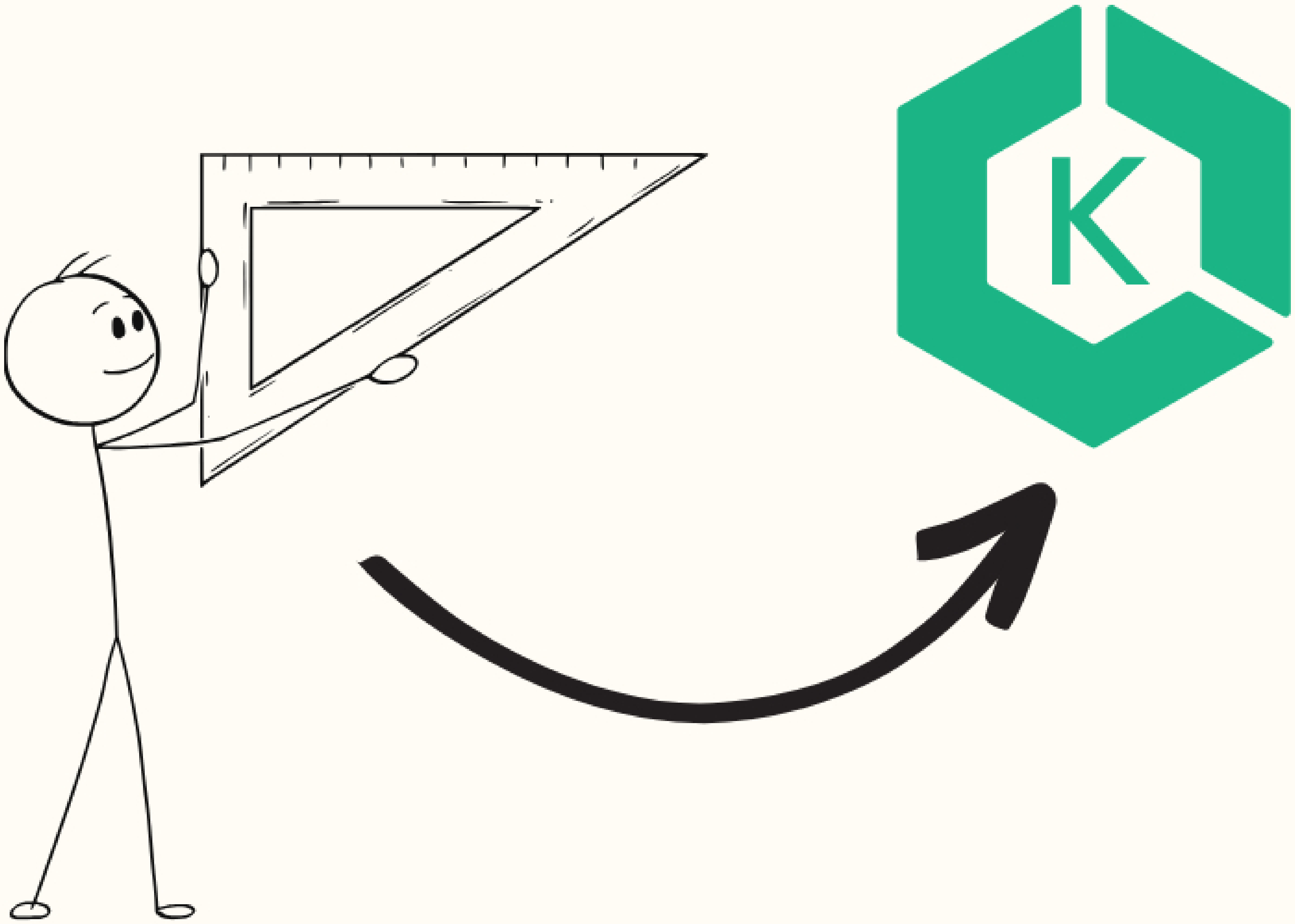
kubernetes





These components interact in complex ways, making it non-obvious to identify the root cause of a performance issue solely based on its surface-level manifestation.

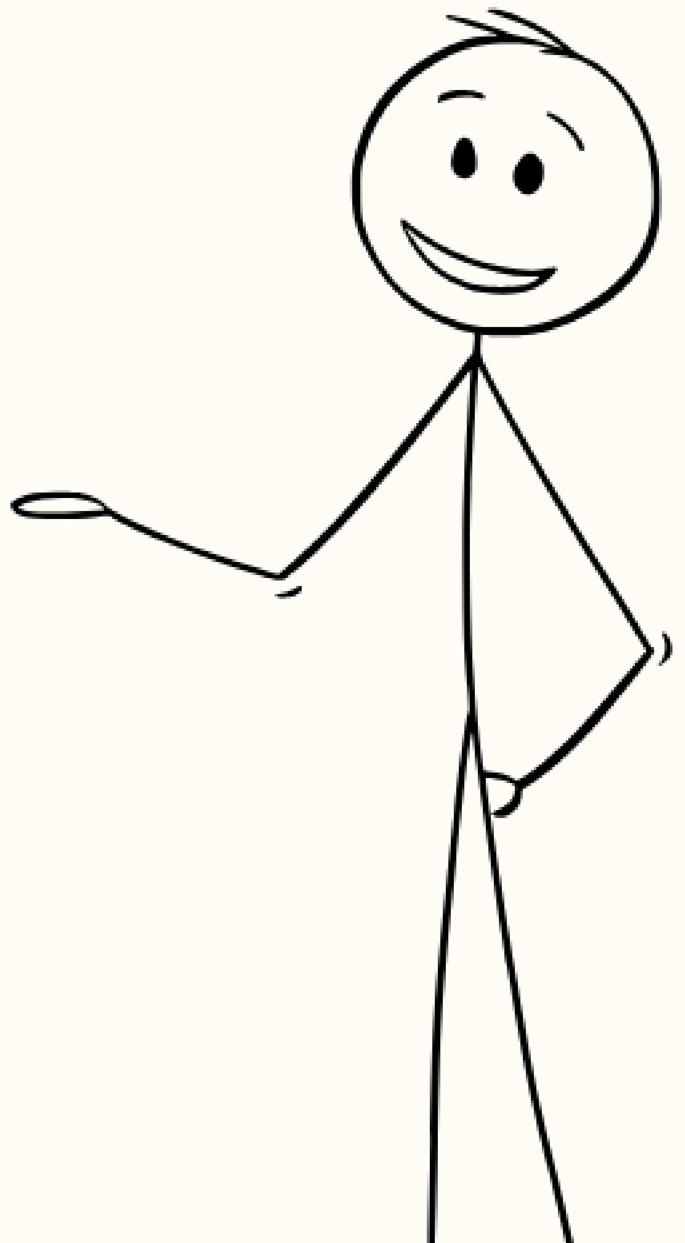
For example, imagine you're troubleshooting an application hosted on K8s that is experiencing high latency.



You know the latency rate for the app...

But that information alone tells you little about the root cause of the problem, which could be any number of the following:

1. Congestion on the network.
2. A problematic configuration with the networking plugin you're using.
3. Buggy code in the application that’s causing it not to respond to requests quickly.
4. Insufficient resources for the Pod that hosts the app, causing slow performance.



We could go on, but the point is this:

Kubernetes troubleshooting is tricky because there are so many potential root causes to sort through...

And so many individual resources that you need to monitor and observe to maintain the visibility necessary to trace problems to their root cause.