Secure Systems Group, Aalto University

Jacopo Bufalino, Mario Di Francesco, Tuomas Aura

Analyzing network-layer access and isolation between microservices

Abstract

Cloud-based applications are composed of several microservices, loosely coupled containers interacting over a network. These applications, and the underlying infrastructure, are configured and deployed using a declarative language to be easily reused and shared. However, reusability often introduces misconfigurations, especially when the components come from unfamiliar third parties. Additionally, the high number of microservices makes it difficult to understand the access control of a cluster. Our work introduces a methodology to analyze the network connectivity of a cluster and the Kubesonde tool, a Kubernetes-based implementation of the methodology. We use Kubesonde to analyze the different sources of misconfiguration of 453 applications.

Analyzing network connectivity with Kubesonde

Our methodology consists of probing TCP and UDP endpoints. Probing is performed from each network namespace to every other network namespace. We discover the targets by combining the cluster APIs and a sidecar agent running alongside each application. Probing is performed when new applications are deployed and is repeated at regular intervals.

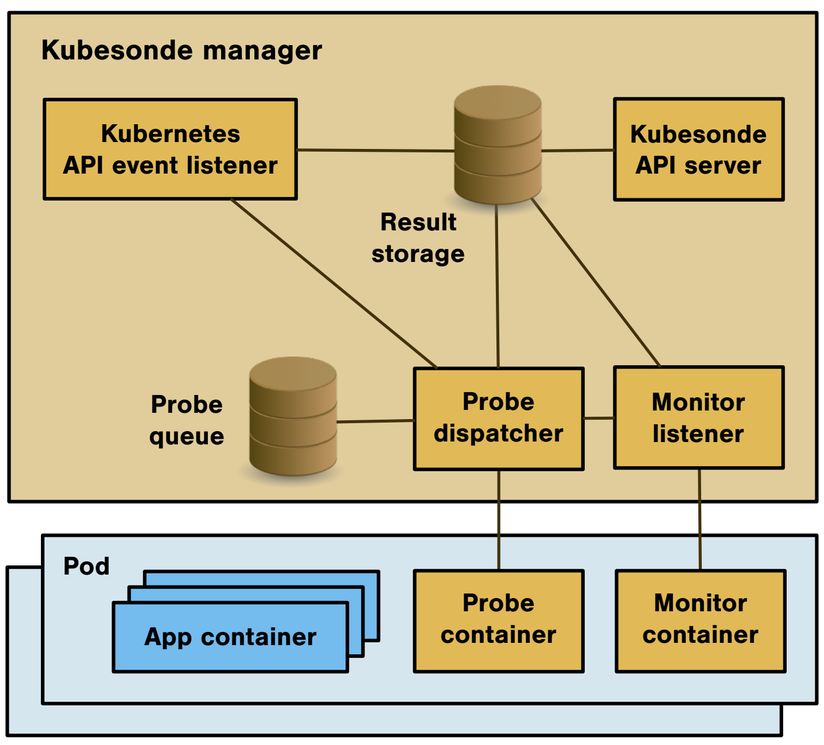

Figure 1: Kubesonde architecture

The different components of Kubesonde (Figure 1) are:

- Monitor containers: one per Pod, record the listening processes
- Probe containers: one per Pod, run probes
- Manager service: create, distribute, schedule, and collect probes

Results

Based on our methodology, we analyzed 453 popular Kubernetes applications deployed as Helm charts. We found several misconfigurations, summarized in Table 1.

Table 1: Summary of the misconfigurations

Misconfiguration	No. applications
Listen to all network interfaces	304
HostNetwork flag enabled	22
Inaccurate port specification	271
Application uses dynamic ports	69
Network policies not enforced	444

In addition to those misconfigurations, other possible attack vectors manifest when an application does not declare network policies (Figure 2 shows an example of those applications). In detail, they are: unnecessary connectivity to the Internet and between microservices, access to public DNS, and lack of access control between Pods. Based on our analysis, we discovered a critical vulnerability affecting a popular CI/CD platform and reported it.

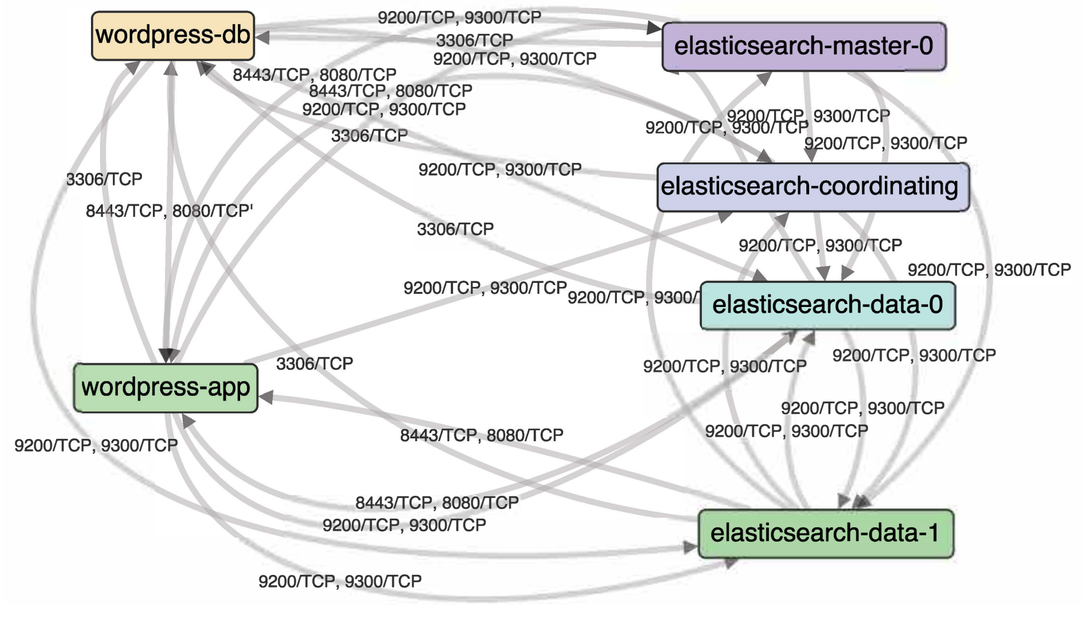

Figure 2: Sample cloud infrastructure without access control

Contact: jacopo.bufalino@aalto.fi, mario.di.francesco@aalto.fi, tuomas.aura@aalto.fi

Aalto University
School of Science