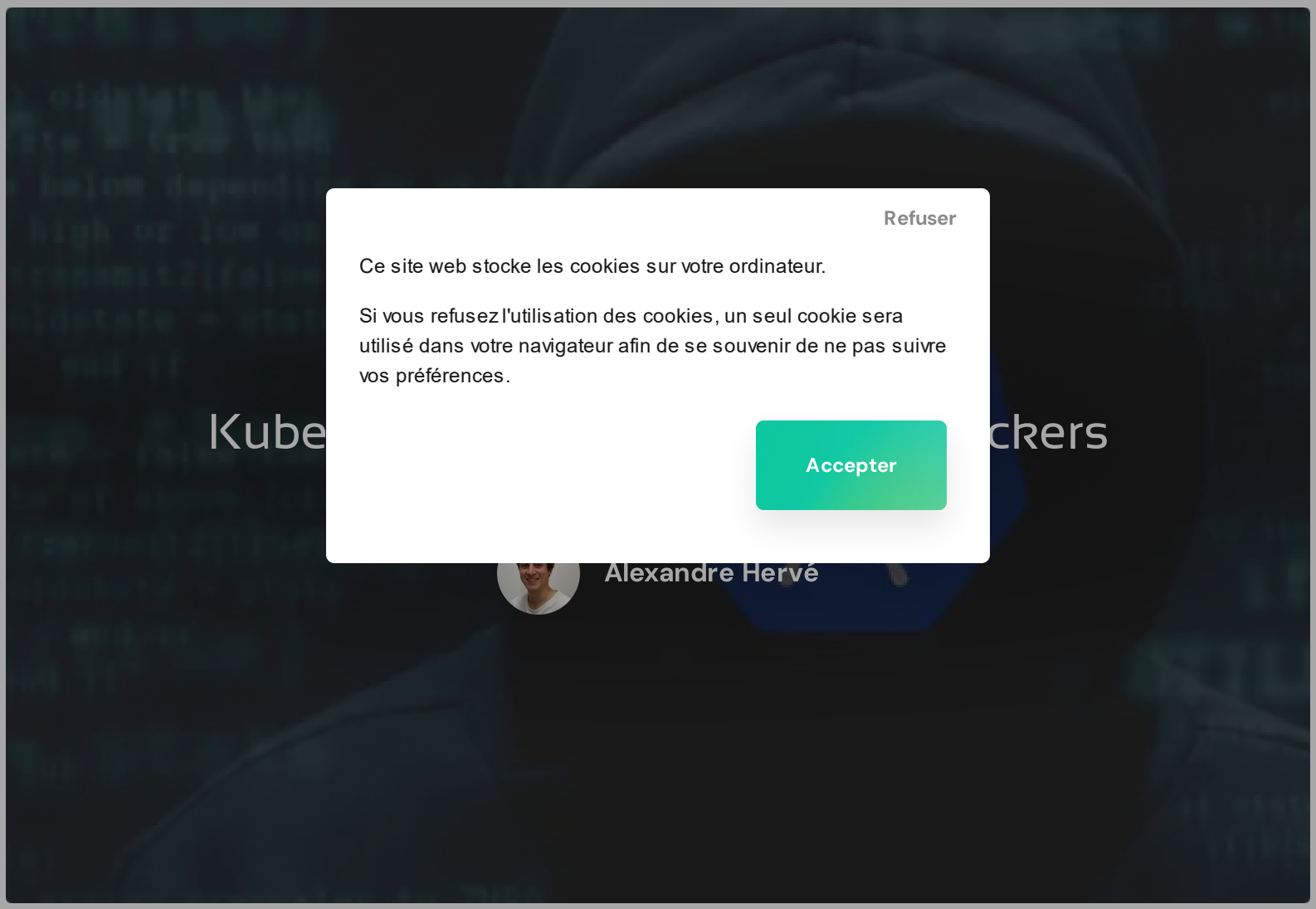
theodo. Cloud



14 December 2023

For an attacker, the main step after compromising a system is to establish persistent access to it. It means that even if you remove his initial access, he could easily come back thanks to the multiple backdoors he installed in your system. Some hackers are even specialized in selling backdoors on the darknet. This threat is also present in your **Kubernetes** clusters.

SOMMAIRE

Microsoft Kubernetes Threat Matrice

Microsoft released a **threat matrix** for Kubernetes based on the MITRE ATT&CK **framework. There are multiple ways to establish persistence in a **Kubernetes cluster** but in this article, we will deep dive into the technique involving malicious admission controllers.

What are Admission Controllers?

Kubernetes Admission Controllers are elements that intercept Kubernetes API requests and can modify, accept, or deny them. This operation occurs before the persistence of a reso oks. A webhook is basically an HTTP ser sends a response to it. There are 2 types

- The **mutating w** step. Its role is to intercept the request, modify it or not, and respond with the patched request. For example, you can add labels, add a sidecar container, change the docker image used, and so on.
- The **validating webhook** is triggered at the end of the flow. It receives the request and can respond if the request is accepted or rejected but can't modify it. For example, you can only validate a deployment that doesn't run a privileged container, check the signature of a container, and so on.

Using admission webhook to implement a backdoor

Now, imagine that you are an **attacker** and you compromised a Kubernetes cluster. Your next step would be to deploy backdoors in order to sell it or to come back if your initial access is discovered. A simple way to do this would be to deploy a backdoored containe ent, for example. But there is a big chance

As myself being a **Kul** ply one of the commands I use the cious because it has been created by someone from outside. Let's try something clever: what if a cluster user is himself deploying backdoor for you? One way to achieve this would be to install a mutating webhook in the cluster.

Now, we won't create any pod manually. Instead, we are going to tell the cluster that each time a new pod is created, we want to inject a backdoor into it. As we saw earlier, the mutating webhook is a perfect candidate for this. Let's use the sidecar injector from this **repo** to inject a backdoored container in all pods.

After the installation of this webhook, every pod of your **Kubernetes cluster** will have a backdoored sidecar container. So the attacker will be able to come back to

your Kubernetes cluster through any of the pod deployed. Of course, it is very noisy to inject a backdoor into every pod.

We can modify the **mutating webhook** to adjust the injection rate or target some specific pods to get something less obvious. This technique has a lot of advantages for the attacker. You c ontainer from backdoor to cryptom ealth. The cluster is itself deploying backd

We can also imagine an external server. In this case, the attacker can change the behavior of the webhook on the fly. He can remain silent for months and activate the payload only when he needs it. That's why detecting this kind of attack can be really hard.

Protect your Kubernetes cluster

A good **RBAC** is not easy to set up and maintain in your Kubernetes cluster and it is not always enough to prevent this kind of attack. Indeed, this attack can be deployed by a malicious cluster user, who can create a webhook from scratch.

There is no perfect way to protect yourself against this. You can't deactivate **webhooks** because they are used by a lot of other elements such as the nginx-ingress controller. You can't really rely on validating webhook because the user able to create a mutating webhook is probably also able to modify validating webhook. Surveillance is probably the best way to deal with this kind of attack. Tools like **Falco** can notify you when it detects malicious behavior in legit pods which gives you a clue that your Kubernetes [illegible]e, inspecting webhooks might be i[illegible]tion every time a mutating webhook is

Conclusion

The strength of this attack might also be a good way to detect it. The attacker introduces a differential between what we want to deploy and what is really deployed by your Kubernetes cluster. This differential can be handled by the **gitops approach**. Indeed, by using tools like **ArgoCD**, you can check if what you have in your Kubernetes cluster is actually what you asked to deploy from your repo. With this mechanism, no mutating webhook would be able to modify your pods to install backdoors in it; otherwise, ArgoCD will correct the deployment.

14 December 2023

Alexandre Hervé

Alexandre is a SecOps Engineer at Theodo Cloud. He assesses the security level of cloud infrastructures and helps protect them against malicious behaviors. He enjoys CTF and playing the piano.

Vous avez un

Contactez-nous →

Partager →

Articles similaires

TECHNOLOGY • 4 MIN

How to master network policies in a Kubernetes cluster?

Learn how to easily restrict network traffic between yo pods in a kubernetes clust using Network Policies.

TECHNOLOGY • 5 MIN

Isolate your sensitive workloads with taints, tolerations and affinities

KUBERNETES • 3 MIN

Kubernetes webhook used by attackers

How can an attacker use malicious admission controllers to settle in your Kubernetes without you being aware

Theodo Cloud Security,
Expert en Cybersécurité à Paris
1 rue de Saint-Pétersbourg
Paris 75008