



We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking "Manage Cookies" at the bottom of the page. [Privacy Statement](#) [Third-Party Cookies](#)

Accept

Reject

Manage cookies



Microsoft

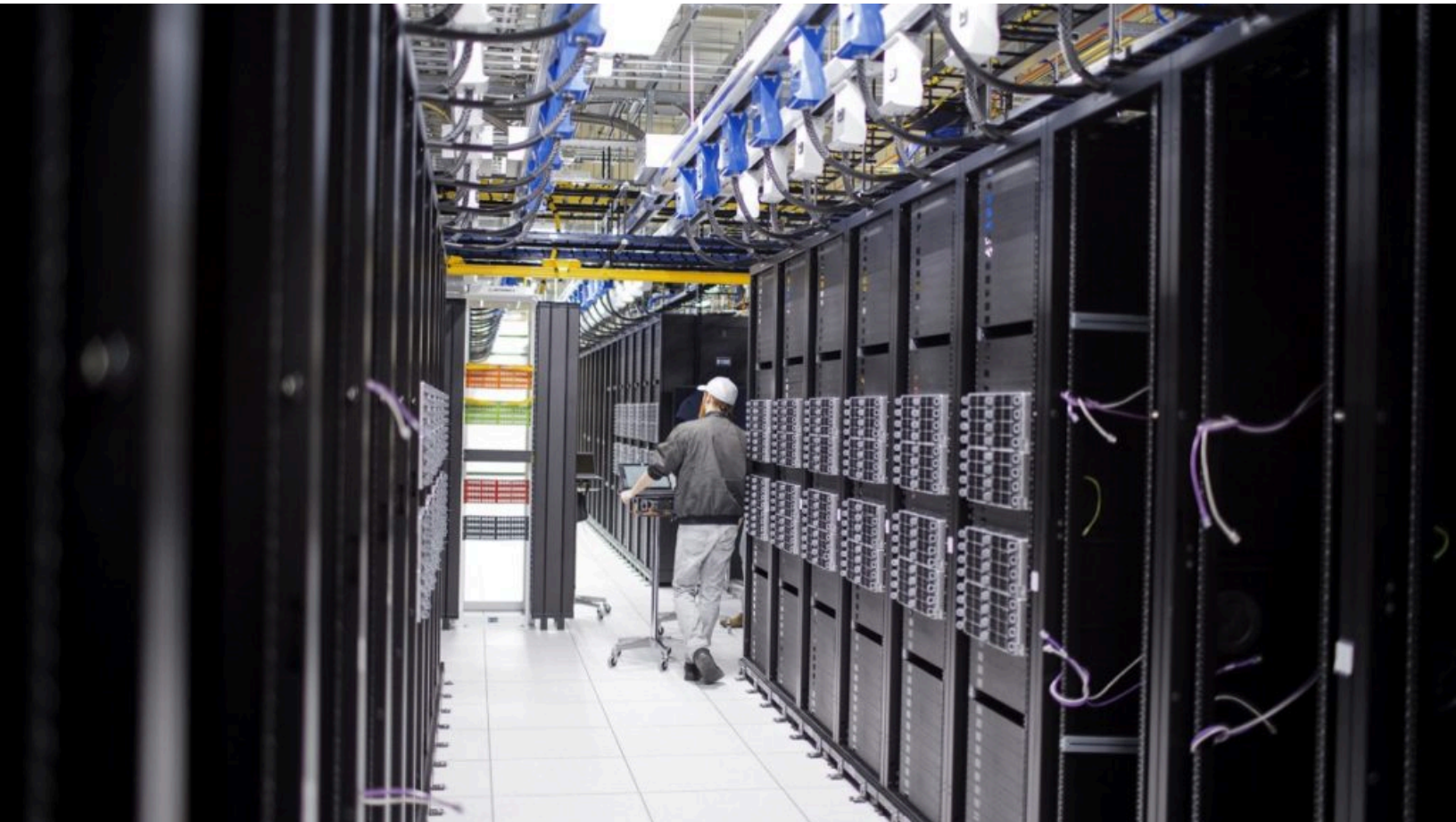
Light

Microsoft Security

[Blog home](#) / Security management

Search the blog





[News](#) [Security management](#) [Microsoft Defender](#)

5 min read

Secure containerized environments with updated threat matrix for Kubernetes

By [Yossi Weizman](#), Senior Security Researcher, Microsoft Defender for Cloud

March 23, 2021



Security strategies

Last April, we released the first version of the [threat matrix for Kubernetes](#). It was the first attempt to systematically map the threat landscape of Kubernetes. As we described in the previous post, we chose to adapt the structure of MITRE ATT&CK® framework which, became almost an industry standard for describing threats.

Since the publication of the threat matrix last year, things have changed:

- New threats were discovered as attackers targeted more and more Kubernetes workloads.
- We were glad to see that the security community adopted the matrix and added more techniques.
- As Kubernetes evolves, it becomes more secure by default and some techniques are no longer relevant.

Today, we are releasing the second version of the threat matrix for Kubernetes, which considers these changes. The updated matrix adds new techniques that were found by Microsoft researchers, as well as techniques that were suggested by the community. We also deprecate several techniques, which do not apply anymore to newer versions of Kubernetes. In this version, we also add a new tactic taken from MITRE ATT&CK®: collection.

| Initial Access | Execution | Persistence | Privilege Escalation | Defense Evasion | Credential Access | Discovery | Lateral Movement | Collection | Impact |
|--------------------------------|-------------------------------------|--------------------------------|------------------------|---------------------------------|---|-----------------------------|---|--------------------------------|--------------------|
| Using Cloud credentials | Exec into container | Backdoor container | Privileged container | Clear container logs | List K8S secrets | Access the K8S API server | Access cloud resources | Images from a private registry | Data Destruction |
| Compromised images in registry | bash/cmd inside container | Writable hostPath mount | Cluster-admin binding | Delete K8S events | Mount service principal | Access Kubelet API | Container service account | | Resource Hijacking |
| Kubeconfig file | New container | Kubernetes CronJob | hostPath mount | Pod / container name similarity | Access container service account | Network mapping | Cluster internal networking | | Denial of service |
| Application vulnerability | Application exploit (RCE) | Malicious admission controller | Access cloud resources | Connect from Proxy server | Applications credentials in configuration files | Access Kubernetes dashboard | Applications credentials in configuration files | | |
| Exposed Dashboard | SSH server running inside container | | | | Access managed identity credential | Instance Metadata API | Writable volume mounts on the host | | |
| Exposed sensitive interfaces | Sidecar injection | | | | Malicious admission controller | | Access Kubernetes dashboard | | |
| | | | | | | | Access tiller endpoint | | |
| | | | | | | | CoreDNS poisoning | | |
| | | | | | | | ARP poisoning and IP spoofing | | |

= New technique

= Deprecated technique

What has deprecated?

Kubernetes evolved and became more secure by default; techniques that appeared in last year's matrix aren't relevant to newer environments. Therefore, we decided to deprecate some of the techniques:

Exposed Kubernetes Dashboard

The Kubernetes dashboard's usage has been in decline for a while now. Cloud-managed clusters, such as Microsoft's AKS and Google's GKE, deprecated this service and moved to a centralized interface in their portals. Moreover, the recent versions of the Kubernetes dashboard require authentication and it's less likely to find exposed dashboards that do not require authentication. Consequently, we also removed the technique, "**Access Kubernetes dashboard**," under lateral movement tactic. Older versions of Kubernetes, including newer clusters that have the dashboard manually installed, are still affected by this technique. The concept of this technique was generalized in the new technique: **exposed sensitive interfaces** (see below).

Access tiller endpoint

As of version 3, Helm doesn't use its server-side component, Tiller. This is a major improvement in the security of Helm. Currently, Helm operates (by default) on behalf of the user's credentials as they appear in the kubeconfig file. Users of older versions of Helm are still affected by this technique.

New techniques for the threat matrix

1. Initial access

Exposed sensitive interfaces

Exposing a sensitive interface to the internet poses a security risk. Some popular frameworks were not intended to be exposed to the internet, and therefore don't require authentication by default. Thus, exposing them to the internet allows unauthenticated access to a sensitive interface which might enable running code or deploying containers in the cluster by a malicious actor. Examples of such interfaces that were seen exploited include Apache NiFi, Kubeflow, Argo Workflows, Weave Scope, and the Kubernetes dashboard.

2. Execution

Sidcar injection

A Kubernetes Pod is a group of one or more containers with shared storage and network resources. Sidecar container is a term that is used to describe an additional container that resides alongside the main container. For example, service-mesh proxies are operating as sidecars in the applications' pods. Attackers can run their code and hide their activity by injecting a sidecar container to a legitimate pod in the cluster instead of running their own separated pod in the cluster.

3. Persistence

Malicious admission controller

Admission controller is a Kubernetes component that intercepts, and possibly modifies, requests to the Kubernetes API server. There are two types of admissions controllers: validating and mutating controllers. As the name implies, a mutating admission controller can modify the intercepted request and change its properties. Kubernetes has a built-in generic admission controller named *MutatingAdmissionWebhook*. The behavior of this admission controller is determined by an admission webhook that the user deploys in the cluster. Attackers can use such webhooks for gaining persistence in the cluster. For example, attackers can intercept and modify the pod creation operations in the cluster and add their malicious container to every created pod.

4. Credential access

Access managed identity credential

Managed identities are identities that are managed by the cloud provider and can be allocated to cloud resources, such as virtual machines. Those identities are used to authenticate with cloud services. The identity's secret is fully managed by the cloud provider, which eliminates the need to manage the credentials. Applications can obtain the identity's token by accessing the Instance Metadata Service (IMDS). Attackers who get access to a Kubernetes pod can leverage their access to the IMDS endpoint to get the managed identity's token. With a token, the attackers can access cloud resources.

Malicious admission controller

In addition to persistency, a malicious admission controller can be used to access credentials. One of the built-in admission controllers in Kubernetes is *ValidatingAdmissionWebhook*. Like *MutatingAdmissionWebhook*, this admission controller is also generic, and its behavior is determined by an admission webhook that is deployed in the cluster. Attackers can use this webhook to intercept the requests to the API server, record secrets, and other sensitive information.

5. Lateral movement

CoreDNS poisoning

CoreDNS is a modular Domain Name System (DNS) server written in Go, hosted by Cloud Native Computing Foundation (CNCF). CoreDNS is the main DNS service that is being used in Kubernetes. The configuration of CoreDNS can be modified by a file named *corefile*. In Kubernetes, this file is stored in a ConfigMap object, located at the *kube-system* namespace. If attackers have permissions to modify the ConfigMap, for example by using the container's service account, they can change the behavior of the cluster's DNS, poison it, and take the network identity of other services.

ARP poisoning and IP spoofing

Kubernetes has numerous network plugins (Container Network Interfaces or CNIs) that can be used in the cluster. Kubenet is the basic, and in many cases the default, network plugin. In this configuration, a bridge is created on each node (*cbr0*) to which the various pods are connected using veth pairs. The fact that cross-pod traffic is through a bridge, a level-2 component, means that performing ARP poisoning in the cluster is possible. Therefore, if attackers get access to a pod in the cluster, they can perform ARP poisoning, and spoof the traffic of other pods. By using this technique, attackers can perform several attacks at the network-level which can lead to lateral movements, such as DNS spoofing or stealing cloud identities of other pods (CVE-2021-1677).

6. Collection

In this update, we also add a new tactic to the threat matrix: collection. In Kubernetes, collection consists of techniques that are used by attackers to collect data from the cluster or through using the cluster.

Images from private registry

The images that are running in the cluster can be stored in a private registry. For pulling those images, the container runtime engine (such as Docker or containerd) needs to have valid credentials to those registries. If the registry is hosted by the cloud provider, in services like Azure Container Registry (ACR) or Amazon Elastic Container Registry (ECR), cloud credentials are used to authenticate to the registry. If attackers get access to the cluster, in some cases they can obtain access to the private registry and pull its images. For example, attackers can use the managed identity token as described in the “Access managed identity credential” technique. Similarly, in EKS, attackers can use the AmazonEC2ContainerRegistryReadOnly policy that is bound by default to the node’s IAM role.

Protect your containerized environments

Understanding the attack surface of containerized environments is the first step of building security solutions for these environments. The revised threat matrix for Kubernetes can help organizations identify the current gaps in their defenses’ coverage against the different threats that target Kubernetes.

We recommend that you start protecting your containerized environment with Azure Defender today. Learn more about Azure Defender’s [support for container security](#).

To learn more about Microsoft Security solutions [visit our website](#). Bookmark the [Security blog](#) to keep up with our expert coverage on security matters. Also, follow us at [@MSFTSecurity](#) for the latest news and updates on cybersecurity.

Related Posts

[News](#) [Endpoint security](#) [Microsoft Intune](#) ·
Feb 1 · 8 min read

3 new ways the Microsoft Intune Suite offers security, simplification, and savings >

The main components of the Microsoft Intune Suite are now generally available. Read about how consolidated endpoint management adds value and functionality for security teams.

[Best practices](#) [Identity and access management](#)
[Microsoft Entra](#)

Jan 10 · 9 min read

5 ways to secure identity and access for 2024 >

To confidently secure identity and access at your organization, here are five areas Microsoft recommends prioritizing in the new year.

[Best practices](#) [Incident response](#)
[Microsoft Security Experts](#)
Jun 6, 2023 · 6 min read

Why a proactive detection and incident response plan is crucial for your organization >

[News](#) [Identity and access management](#)
[Microsoft Entra](#)
May 31, 2023 · 5 min read

XDR meets IAM: Comprehensive identity threat detection and response with Microsoft >

Matt Suiche of Magnet Forensics talks about top security threats for organizations and strategies for effective incident response.

Identity-based attacks are on the rise, making identity protection more important than ever. Explore our blog post to learn how Microsoft's Identity Threat Detection and Response can help.

Get started with Microsoft Security

Microsoft is a leader in cybersecurity, and we embrace our responsibility to make the world a safer place.

[Learn more](#)

Connect with us on social



What's new

[Surface Pro](#)

[Surface Laptop](#)

Microsoft Store

[Account profile](#)

[Download Center](#)

Education

[Microsoft in education](#)

[Devices for education](#)

| | | |
|--|---|---|
| Surface Laptop Studio 2 | Microsoft Store support | Microsoft Teams for Education |
| Surface Laptop Go 3 | Returns | Microsoft 365 Education |
| Microsoft Copilot | Order tracking | How to buy for your school |
| AI in Windows | Certified Refurbished | Educator training and development |
| Explore Microsoft products | Microsoft Store Promise | Deals for students and parents |
| Windows 11 apps | Flexible Payments | Azure for students |

Business

- [Microsoft Cloud](#)
- [Microsoft Security](#)
- [Dynamics 365](#)
- [Microsoft 365](#)
- [Microsoft Power Platform](#)
- [Microsoft Teams](#)
- [Microsoft 365 Copilot](#)
- [Small Business](#)


Developer & IT

- [Azure](#)
- [Developer Center](#)
- [Documentation](#)
- [Microsoft Learn](#)
- [Microsoft Tech Community](#)
- [Azure Marketplace](#)
- [AppSource](#)
- [Visual Studio](#)

Company

- [Careers](#)
- [About Microsoft](#)
- [Company news](#)
- [Privacy at Microsoft](#)
- [Investors](#)
- [Diversity and inclusion](#)
- [Accessibility](#)
- [Sustainability](#)

 [English \(United States\)](#)

 [Your Privacy Choices](#)

[Consumer Health Privacy](#)