Technique	Description	Tactics
Abuse Elevation Control Mechanisms	Adversaries may abuse mechanisms designed to elevate privileges such as SETUID and SETGID, sudo, or file capabilities to gain or persist privileged access to a system. Privileged access can be gained by exploiting weaknesses in the elevation control mechanism. Attackers who obtained privileged access can persist it through Elevation Control Mechanisms. For example, they could plant a malicious SETUID binary or edit the sudoers file.	Privilege Escalation, Persistence
Access the Kubelet Main API	Adversaries may access the kubelet's main API to gather information on cluster resources or execute commands on running pods. The kubelet exposes a server, normally at port 10250, that can be used to execute commands in running pods, and could be configured to allow anonymous access. An adversary with network access and appropriate credentials to a properly configured kubelet, or a misconfigured kubelet, may access the kubelet API to move laterally and execute code in pods managed by the kubelet, or gather information on the pods managed by the kubelet.	Execution, Lateral Movement, Discovery
Access the Kubernetes API Server	Adversaries may access the Kubernetes API server to discover, create, compromise, or delete cluster resources for a variety of purposes, most notably Discovery, Lateral Movement, Persistence and Credential Access. The API server is the gateway to the cluster that exposes the Kuberntes API, allowing clients to manage the cluster according to their permissions. Adversaries with sufficient privileges in the cluster can access the API server to discover cluster resources and the cluster's version; execute code on existing pods to move laterally within the cluster; create privileged pods to compromise their underlying nodes; deploy persistent compute objects such as ReplicaSets, Deployments and CronJobs to maintain access to the cluster; set up Admission Controllers to intercept and possibly mutate new cluster objects; retrieve Secrets; create users, certificates, service accounts, (cluster) roles and (cluster) rolebindinges to maintain access to the cluster; collect or delete logs; or cause a denial-of-service by deleting existing compute or network resources.	Discovery, Lateral Movement, Persistence, Credential Access
Account Access Removal	Adversaries may interrupt the availability of system and network resources by inhibiting access to accounts utilized by legitimate users. Local accounts, cloud accounts, and Kubernetes user and service accounts could be deleted, locked, or manipulated (e.g. changing password or permissions) to remove access.	Impact

Account Discovery	Adversaries may attempt to get a listing of local accounts, cloud accounts, or Kubernetes service accounts. This information can help adversaries determine which accounts exist to aid in follow-on behavior.	Discovery
Account Manipulation	Adversaries may manipulate accounts to maintain access to victim systems. Account manipulation can consist of any action that preserves adversary access to a compromised environment, such as modifying or adding credentials to an account or changing permissions. In Kubernetes environments, adversaries can persist access by creating RoleBindings and ClusterRoleBindings that grant access to an adversary-controlled service account, user, or group.	Persistence
Application Exploit (RCE)	Adversaries may attempt to exploit Remote Code Execution (RCE) vulnerabilities in an application to gain code execution on the underlying VM or container.	Execution
Cloud Instance Metadata API	Adversaries may attempt to access the Cloud Instance Metadata API to collect credentials and other sensitive data. Most cloud service providers expose a Cloud Instance Metadata API at http[:]//169.254.169.254 to virtual instances that provides applications with information about the virtual instance. This information may include the instance's name, security groups, OS version, associated credentials, and startup scripts that could contain secrets. In most managed Kubernetes offerings, such as GKE, pods can access their node's metadata API by default. If adversaries have a presence on a running virtual instance, they may query the Instance Metadata API to identify credentials that grant access to additional resources. Additionally, attackers may exploit Server-Side Request Forgery (SSRF) vulnerabilities in public facing cloud applications to gain access to the Instance Metadata API.	Credential Access Discovery
CommandAnd Control/GENERAL	The technique refers to different general actions that adversaries may use to communicate with systems under their control within a victim network. There are many ways an adversary can establish command and control with various levels of stealth depending on the victim's network structure and defenses.	Command and Control
Compile After Delivery	Adversaries may attempt to make payloads difficult to discover and analyze by delivering files to victims as uncompiled code. Text-based source code files may subvert analysis and scrutiny from protections targeting executables/binaries. These payloads are compiled before execution on the infected system.	Defense Evasion

Create Account	Adversaries may create an account to maintain access to victim systems. Accounts could be local accounts, cloud provider accounts and service accounts, and Kubernetes service accounts.	Persistence
Create Container	Adversaries with appropriate permissions may deploy new containers to the environment to execute their malicious code. In Kubernetes environments, adversaries could use controllers such as Deployments, ReplicaSets, CronJobs, or DaemonSets to create Backdoor Containers that persist in the cluster.	Execution, Persistence
Credential Dumping	Adversaries may attempt to dump credentials to obtain account login and credential material, normally in the form of hashes or clear text passwords, from the compromised system. On Linux, /etc/passwd and /etc/shadow store user account information and password hashes, which can be used for offline password cracking. Additionally, given root access, adversaries can abuse the procfs filesystem to scan and harvest credentials from the memory of all running processes on the system.	Credential Access
Endpoint Denial-of-Service	Adversaries may perform Endpoint Denial of Service (DoS) attacks to degrade or block the availability of services to users. An Endpoint DoS blocks the availability of a service without saturating the network that provides access to the service. Adversaries can target various layers of the hosting system's application stack. These layers include the operating systems, server applications, DNS servers, databases, and the (typically web-based) applications that sit on top of them. Adversaries could exhaust resources, abuse bottlenecks, and exploit persistent crash conditions in a target service, either through a single request or a flood of requests. In Kubernetes environments, launching a DoS attack against the API server can significantly reduce the availability of the cluster.	Impact
Event Triggered Execution	Adversaries may establish persistence using system mechanisms that trigger execution based on specific events. Shell initialization scripts such as .bash_profile and .bashrc execute upon execution of a shell process. Adversaries may plant malicious code in those scripts to persist on a target machine.	Persistence
Exec Into Container	Adversaries with appropriate permissions may run malicious commands in containers with the exec command (“kubectl exec” and "docker exec").	Execution
Exfiltration	Adversaries may steal data by exfiltrating it over the Command and Control channel or over an alternative, separate channel. Stolen data may be encoded, encrypted or otherwise obfuscated.	Exfiltration

Exploit Public-Facing Application	Adveseries may attempt to exploit one-day or zero-day vulnerabilities in a public facing application to gain Initial Access. If an application is hosted on cloud-based infrastructure or in a Kuberntes cluster, then exploiting it could lead to compromise of the underlying instance or pod, which may have credentials attached.	Initial Access, Execution
Exploitation for Privilege Escalation	Adversaries may exploit software vulnerabilities in an attempt to elevate privileges. Exploitation of a software vulnerability occurs when an adversary takes advantage of a programming error in a program, service, or in the kernel itself, to execute adversary-controlled code. Security constructs, such as permission levels, will often hinder access to information and the use of certain techniques, so adversaries will likely need to perform privilege escalation through software exploitation to circumvent those restrictions.	Privilege Escalation
Exploitation of Remote Services	Adversaries may exploit vulnerable remote services to gain unauthorized access to remote hosts or containers within a network or cluster. Exploitation of a vulnerability in a remote service occurs when an adversary takes advantage of a programming error in a service to execute attacker controlled code.	Lateral Movement
File and Directory Discovery	Adversaries may enumerate files and directories or search specific locations for certain information within a file system. Adversaries could use the information obtained during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions. Utilities like 'ls' and 'find' can be used to obtain this information, as well as custom binaries and scripts.	Discovery
Foreign Binary Execution	Adversaries may install and run utilities, malware, or third-party applications to gain custom execution on a target. Foreign binaries can be brought to a compromised system through Ingress Tool Transfer or Lateral Tool Transfer.	Execution
Hijack Execution Flow	Adversaries may execute their own malicious payloads by hijacking the way a system runs programs. Hijacking execution flow can be employed for the purpose of persistence, since hijacked execution can recur over time. Adversaries may also use these mechanisms to elevate privileges or evade defenses. There are many ways an adversary could hijack the flow of execution, including by manipulating how the operating system locates programs to be executed or libraries to be loaded. Common techniques include modifying the dynamic linker configuration (e.g. through /etc/ld.preload), planting malicious versions of a binary or library under a directory placed early in the search path, and modifying the binary search path itself.	Persistence, Defense Evasion, Privilege Escalation

Impair Defences	Adversaries may maliciously modify components of a victim environment in order to hinder or disable defensive mechanisms. This not only involves impairing preventative defenses, such as firewalls and anti-virus, but also detection capabilities that defenders use to audit activity and identify malicious behavior.	Defense Evasion
Ingress Tool Transfer	Adversaries may transfer tools or other files from an external system into a compromised environment. Files could be copied from an external adversary controlled system through the command and control channel to bring tools into the victim network. Alternatively, files could be copied using alternative protocols, such as FTP or through native tools like scp, rsync, and sftp.	Command and Control
Kubernetes Secrets	A Kubernetes secret is an object that lets users store and manage sensitive information in the cluster, such as passwords and connection strings. Adversaries with appropriate permissions may retrieve secrets from the API server (by using the pod service account, for example) and access the sensitive information stored in them.	Credential Access
Lateral Tool Transfer	Adversaries could transfer tools or files between systems in a compromised environment. Files could be copied from one system to another to stage adversary tools. Adversaries could copy files laterally between internal victim systems to support lateral movement using file sharing protocols and native tools, like scp, rsync, and sftp.	Lateral Movement
Man-in-the-Middle	Adversaries may attempt to position themselves between two or more networked devices using a man-in-the-middle (MiTM) technique to support follow-on activity such as Network Sniffing or Transmitted Data Manipulation. By abusing common networking protocol features which control network traffic flow (e.g. ARP spoofing), adversaries may force a device to communicate through an adversary-controlled system so they could collect information and credentials or manipulate transmitted data.	Collection Credential Access
Masquerading	Adversaries may attempt to manipulate features of their artifacts to make them appear legitimate or benign to users and/or security tools. Masquerading occurs when the name or location of an object is manipulated for the sake of evading defenses and observation. This may include manipulating names and metadata of files, services or cloud instances. Adversaries may name their malware after commonly used utilities, or place it under directories used for executables such as /bin. In Kubernetes, controllers such as Deployments and DaemonSets attach a random suffix to pods created by them. Attackers may match this behavior to masquerade malicious pods as legitimate ones created by a controller.	Defense Evasion

Native Binary Execution	Adversaries may use installed binaries, like curl and apt, or interpreters, such as bash and Python, to gain execution on a target. Using binaries native to an environment can help conceal the attack from defenders.	Execution
Network Service Scanning	Adversaries may attempt to get a listing of services running on remote hosts and pods, including those that may be vulnerable to remote software exploitation. Methods to acquire this information include port scanning and vulnerability scanning with tools brought onto a system. By default, Kubernetes doesn't restrict pod communications, meaning an attacker with access to a Kubernetes pod can discover and map other pods on the cluster.	Discovery
Obfuscated Files	Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, compressing, padding, or otherwise obfuscating its contents on the system or in transit. Adversaries may strip file identifiers such as ELF headers from payloads to make them harder to identify and analyze.	Defense Evasion
Privileged Container	Adversaries who gain access to a privileged container or can create a privileged container may use its elevated privileges to compromise the underlying host. A privileged container isn't necessarily one that runs with the infamous privileged flag. It can be any container configured with elevated privileges, such as additional kernel capabilities, shared host namespace, exposed devices, or lack of cgroups isolation, that allow it to compromise the underlying host.	Privilege Escalation
Query the Kubelet Readonly API	Adversaries may query the kubelet's read-only API to discover the configuration of the kubelet's node and the pods running on it. By default, the kubelet exposes a read-only API at port 10255 that doesn't enforce authentication. An adversary with network access to a node can query the kubelet's read-only API to retrieve the configuration of all pods running on the kubelet's node, as well as node information and metrics. A pod's configuration may include sensitive information, such as the containers' image, environment variables, and command.	Discovery
Resource Hijacking	Adversaries may leverage the resources of compromised hosts and containers in order to solve resource intensive problems which may impact system and service availability. One common purpose of Resource Hijacking is to earn virtual currency through cryptomining.	Impact
Scheduled Task/Job	Adversaries may abuse task scheduling functionality to facilitate initial or recurring execution of malicious code. Utilities such as systemd timers, at, and cron could be abused.	Execution, Persistence

Software Deployment Tools	Adversaries may gain access to software deployment tools and administrative tools, such as the Docker daemon, the Kubernetes API Server, the Kubernetes Dashboard, and the Helm v1 Tiller, to move laterally within an environment. Abusing software deployment tools could enable adversaries to gain remote code execution on all entities connected to the system by deploying payloads in the form of binaries, services, or containers. The permissions required for this action vary by system configuration. If the deployment or administrative tools are misconfigured, they might allow unauthenticated access, meaning any adversary with network access to the tool could control it.	Lateral Movement
Software Discovery	Adversaries may attempt to get a listing of software and software versions that are installed on a local or remote system. Adversaries could use the information from Software Discovery during automated discovery to shape follow-on behaviors, including whether or not the adversary attacks the target and/or attempts specific actions. Adversaries may attempt to enumerate software for a variety of reasons, such as determining what security measures are present or if the compromised system has a version of software that is vulnerable to Exploitation for Privilege Escalation.	Discovery
Supply Chain Compromise	Adversaries may manipulate software, software dependencies, deployment artifacts, or infastracture-as-code files prior to receipt by a final consumer to gain Initial Access to a system. Adversaries may plant malware in binaries, libraries, container images, Kubernetes YAML files, and Helm charts.	Initial Access
System Information Discovery	An adversary may attempt to get detailed information about the operating system and hardware, including version, patches, hotfixes, service packs, and architecture through utilities such as uname and special files like lsb-release or /boot/config. Adversaries may use this information to shape follow-on behaviors, such as exploitation of one-day vulnerabilities in unpatched operating systems.	Discovery
System Network Configuration Discovery	Adversaries may attempt to get the network properties of a system to formulate follow-on behaviour. Adversaries can use tools, such as 'ifconfig' and 'ip'. They may also query the operating system directly, for example, by reading procfs files, such as /proc/net/route.	Discovery
System Network Connections Discovery	Adversaries may attempt to get a listing of network connections of the compromised system they are currently accessing, or of remote systems by querying for information over the network. In Linux, the netstat, lsof and ss utilities can be used to list	Discovery

	current connections.	
System Owner/User Discovery	Adversaries may attempt to identify the primary user, currently logged in user, set of users that commonly uses a system, or whether a user is actively using the system. They may do this, for example, by retrieving account usernames or by using OS Credential Dumping. The information may be collected in a number of different ways using other Discovery techniques, because user and username details are prevalent throughout a system and include running process ownership, file/directory ownership, session information, and system logs. Adversaries may use the information from System Owner/User Discovery during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions. In Linux, the whoami utility can retrieve the currently logged in user, and w and who utilities can list all currently logged in users.	Discovery
Unsecured Credentials	Adversaries may search compromised environments for insecurely stored credentials. These credentials can be stored and/or misplaced in many locations, including local files, such as bash history, private keys, Kubernetes YAML files, source code repositories, artifacts, such as container and VM images, and in environment variables of containers and VMs.	Credential Access
Web Shell	Adversaries may backdoor web servers with web shells to establish persistent access to systems. A web shell is a web script that is placed on an openly accessible web server to allow an adversary to use the web server as a gateway into a network. A web shell can provide a set of functions to execute or a command-line interface on the system that hosts the web server.	Persistence