GitHub Scenraio Threats and Ground Truth									
ID	**QUALTRICS ID**	**Threats description**	**Assumptions**		**STRIDE type**	**Affected Components**	**Real threat?**	**Justification**	
1-Training	EXPLOIT-REMOTE-REPO	Exploit a vulnerability in the code in the remote repository to attack the server running the code.	1. Remote repo & running server reachable by the attacker; 2. The source code has security vulnerabilities.		Information disclosure	Victim's server	Yes	Remote repositories and GitHub servers can be reached by anyone if the owner of the repo has not set enough restrictions e.g., setting its visibility to public. https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/managing-repository-settings/setting-repository-visibility	Use for training and excluded from actual task
3	STOLEN-AUTH-INFO	Spoofing a remote repo admin by stealing the authentication credentials via a social engineering attack.	1. The attacker carries out a successful social engineering attack (attackers communicate legitimately with others, manipulating and exploiting human qualities to achieve their attack) and gets authentication credentials; 2. The credentials are valid.		Spoofing	The remote code repository	Yes	Using simple alert-style email notifications, attackers are able to steal credentials to gain access to development code, intellectual property, and project details. Here is an example of a phishing attack on GitHub users https://github.blog/2020-04-14-sawfish-phishing-campaign-targets-github-users/	
2	LEAKED-CONFIG-FILE	An attacker uses leaked (e.g. on Github) configuration file to interact with the remote repo, potentially jeopardize running code and submitting malicious code.	1. The attacker finds the leaked information; 2. The attacker can (remotely) interact with the remote repo.		Information disclosure	Victim's server	Yes	Git's configuration files are plain-text, when leaked an attacker can view personal information of the repo owner, leading to information disclosure attack. In addition, the attacker can alter the default bahavior set by the owner by manually editting these values. https://www.git-scm.com/book/en/v2/Customizing-Git-Git-Configuration	
4	DOS-SERVER	An attacker submit malicious code can crash the server running the code, causing a Denial of Service (DoS).	1. The attacker submits malicious code that can cause Denial of Service (DoS) to remote repo; 2. The server has pulled and been running the code submitted by the attacker.		DoS	Victim's server	yes	An attacker can exploit vulnerabilities (i.e., lack of authentication) in the git protocol to send multiple requests that overlaod/crash the server causing a DoS attack. https://git-scm.com/book/en/v2/Git-on-the-Server-The-Protocols	
5	MALICIOUS-CODE-GITHUB	An attacker submit malicious code to the project on GitHub and get a shell of the server running the code.	1. The attacker can submit code to remote repo; 2. The server has pulled and been running the malicious code submitted by the attacker.		Elevation of privilege	Victim's server	Yes	An attacker can get server and /path/to/repo from git remote (only if the remote repo is an ssh type). This permits the execution of server-side Git commands implementing the pull/push functionality. https://git-scm.com/docs/git-shell	
6	ELEVATION-PRIVILEDGED-ACCESS	The attacker uses the malicious code to gain root access to the server.	1. The attacker can submit code to remote repo; 2. The server has pulled and been running the code submitted by the attacker; 3. Code run as root on the server.		Elevation of privilege	Victim's server	Yes	With root access, the attacker is able to perform any actions on GitHub, such as creating a pull/push request in a repository. https://docs.github.com/en/get-started/learning-about-github/access-permissions-on-github	
7	DOS-REMOTE-REPO	The attacker submits the same fake push request to other remote code repositories, causing a Denial of Service attack.	1. Permission management is properly implemented in the remote (GitHub) provided repository hosting service to differentiate user access to the code repository; 2. An attacker has the right to submit a bogus push request to a remote repository.		DoS	Other remote code repositories	No	It is possible for an attacker to submit a fake push request to other remote code repositories if they have access to the necessary credentials or exploit a vulnerability in the system. This could potentially cause a Denial of Service attack. But the assumptions clearly states that permision mamagement is properly implemented.	
8	DISCLOSE-THIRD-PARTY	Not only can the attacker use these third-party services, but the attacker can also push code to the remote code repository where the yml file is located.	1. An attacker can browse the project's yml file; 2. The developer has left the keys to the third-party services they need to use in the yml file that configures GitHub Actions.		Information disclosure	Third-party services The remote code repository	No	It is possible for an attacker to browse the project yml file, especially if the repository is public. However, an attcker with access to the yml files can ***only view*** the workflow of the repo.	

9	ELEVATION-PRIVILEDGED-REPO	The attacker can launch a push request to a repository on GitHub and can delete the repository.	The attacker has gained push access to the code repository on GitHub.		Elevation of privilege	The remote code repository	No	To perform actions on GitHub, such as creating a pull request in a respository, a person must have sufficient access to the repo. This access is controlled by permisions. The only way to gain push access when the admin/owner of the repository assigns these permisions to you. Hence, it is not possible for an attacker to gain access if they are not directly assigned to them. Making the application of the threat impossible.	
10	ELEVATION-PRIVILEDGED-CODE	An unauthenticated and non privileged attacker can still submit custom code into the remote repo to prepare the first step of another attack, e.g. turning off logging service or cause a Denial of Service.	The attacker can reach the remote repo (e.g. through internet).		Elevation of privilege, Tampering	The remote code	No	It is possible for an unauthenticated and non-privileged attacker to submit custom code into a remote repository. This might also be made possible if there are vulnerabilities in the repository's access controls or security features. However, most repositories have security measures in place to prevent unauthorized access and modification of code. Moreover, changes made to code in a repository are often review before being approved and merged to the master node. The approval process can reduce the risk of unauthorised changes. Making this threat fake.	
11	EXPLOIT-HTTP-PROTOCOL	If an attacker compromises a http protocol on GitHub, the attacker can steal other users' identities by using HTTP by running a man in the middle attack.	1. GitHub uses the HTTPS protocol; 2. The attacker has compromised the HTTP protocol on GitHub.		Spoofing	GitHub engine	No	It is generally known that HTTP is an insecure protocol and can be vulnerable to such attacks. However, GitHub uses HTTPS, which encrypts traffic between the server and client, which can ultimatley prevent unauthorized access. The HTTPS protocol provides an additional layer of security for users.	

Kubernetes Scenraio Threats and Ground Truth

ID	QUALTRICS ID	Threat Description	Assumptions		STRIDE Threat	Affected Components	Real Threat	Justification	
1-Training	EXPLOIT-PRIVILEGES-WEB-PODS	Exploit a vulnerability in an Internet-facing web application to get a reverse shell on a pod.	- Web app reachable by the attacker (e.g. available on the Internet); - The web app is vulnerable (e.g. to remote code execution).		Elevation of privilege	- Pod / Container and the application running on top.	Yes	Used fr walkthrough, also removed from the task Running a public-facing vulnerable application in a cluster can enable initial access to the cluster. A container that runs an application that is vulnerable to remote code execution vulnerability (RCE) may be exploited; this can allow an attacker to get a reverse shell on the container and, if a service account is mounted, move laterally within the cluster.	Use for training and excluded from actual task
2	LEAKED-PRIVILEGE-REMOTE	An attacker uses leaked (e.g. on Github) cluster configuration file (e.g. /etc/kubernetes/admin.conf) to interact with the cluster, potentially jeopardize running workloads and creating malicious pods.	- The attacker finds the leaked information; - The attacker can (remotely) interact with the cluster.		Elevation of privilege Information Disclosure	- Not present in the DFD.	Yes	In cases where the Kubernetes cluster is deployed in a public cloud (e.g., AKS in Azure, GKE in GCP, or EKS in AWS), compromised cloud credential can lead to cluster takeover. Attackers who have access to the cloud account credentials can get access to the cluster's management layer.	
3	SPOOFING-AUTH-WORKLOAD	Spoofing a cluster admin by stealing the authentication credentials via a social engineering attack.	- The attacker carries out a successful spoofing attack; - The credentials are valid.		Spoofing	- Cluster configuration - Running workloads (pods, container, etc.) - Access control (of users and services) In general, he/she can take over the all cluster.	Yes	Same as the second threat, by gaining access to a cluster using leaked credential configuration files, or by using interacting with an unauthenticated API server of a cluster.	

4	DOS-WORKERNODE	An attacker with shell access to only one pod which has no resource limits applied (e.g. max memory and cpus usage) can crash the worker node on which that pod is running, causing a Denial of Service (DoS).	- The attacker got a remote shell on a pod.		DoS	- Compromised pod, worker node, and all the other pods running on the same node.	Yes	A container without CPU and memory resource limits can use up to all the worker node's resources (e.g., cryptocurrency mining scripts); this will likely cause crashing of other containers running on the same node, and in the worst case, of the node itself.	
5	ELEVATION-PRIVILEGE-MALICIOUS-IMG	An attacker able to upload images to a container registry from which a Kubernetes cluster retrieves pods images, he or she can potentially execute a malicious pod inside the cluster and get a shell from the malicious container.	- The attacker has permissions to upload or modify images in the registry used by the Kubernetes cluster;		Elevation of privilege	- Images Registry - Running pod	Yes	Running a compromised image in a cluster can compromise the cluster. Attackers who get access to a private registry can plant their own compromised images in the registry. The latter can then be pulled by a user. In addition, users often use untrusted images from public registries (such as Docker Hub) that may be malicious (supply-chain attacks).	
6	EXPLOIT-PRIVILEGED-CONTAINER	Privileged containers run as root on the host, thus an attacker compromising one of such containers, gets automatically root access on the host.	- The attacker get acccess to a privileged container running in the cluster; - The attacker escapes the container.		Elevation of privilege	- Pod - Worker node	Yes	Either by exploiting a vulnerability and getting a reverse shell (as in the first threat), or by laterally move to a privileged container, if an attacker gets a reverse shell on a privileged container, that is equivalent to have access to the container's worker node, which means cluster take-over.	
7	PORT-JAMMING-NETWORK-POLICIES	If an attacker deploys a malicious pod into a namespace of the cluster with network policies in place, the attacker can use such a pod to send bogus network packages to pods in different namespaces to jam the exposed ports, causing a disruption in the cluster performance.	- The network policies are implemented correctly to segment the namespaces.		DoS	- Pods in different namespaces.	No	The threat is fake because the network policies are implemented correctly, so no network traffic is allowed between the two namespaces.	
8	LEAKED-SECRET-DOCKERFILE	If a developer embeds a secret in a Dockerfile A and then builds an image (image A), an attacker with access to image A can reverse the image's layers to not only observe the embedded secret in Dockerfile A but also from other images that were built from different Dockerfiles.	- The attacker has access to the image		Information Disclosure	- Not present in the DFD - Cluster secrets/credentials	No	The attacker with access to image A, he/she can only reverse image A and Dockerfile A, but not other images or Dockerfiles.	
9	CHAIN-ATTACK-MALICIOUS-INPUTS	If an attacker compromises a running container, he/she can only continue exploiting the container by running the software specified in Dockerfile (by the developers) with malicious inputs.	- The attacker got access to a running container.		Elevation of privilege Tampering	- Running pod/container	No	If an attacker compromises a running container, he/she can run any kind of software/tool/process, independently from what is written in the Dockerfile.	
10	UNAUTH-CONFIG-TAMPERING	An unauthenticated and non privileged attacker can still upload custom pod configuration into the cluster to prepare the first step of another attack, e.g. turning off logging service or cause a denial of service.	- The attacker can reach the cluster API server (e.g. through internet)		Elevation of privilege Tampering	- New pods	No	This threat may have been true if anonymous access was enabled in the API server. However, this is not specified, so the threat is fake because the attacker does not have enough privileges to upload custom configurations.	
11	SPOOFING-LAYER-3	If an attacker compromises a pod in a Kubernetes cluster using a Layer 3 network solution, he/she can steal other pod's identities and laterally move within the cluster using the network bridge.	- The attacker exploits one container - The CNI works at layer 3		Spoofing	- Lateral movement between pods	No	This threat is fake because a network bridge is a Layer 2 network devide. If the cluster is using a Layer 3 network plugin, the network bridge is not present, thus it can not be exploited.	