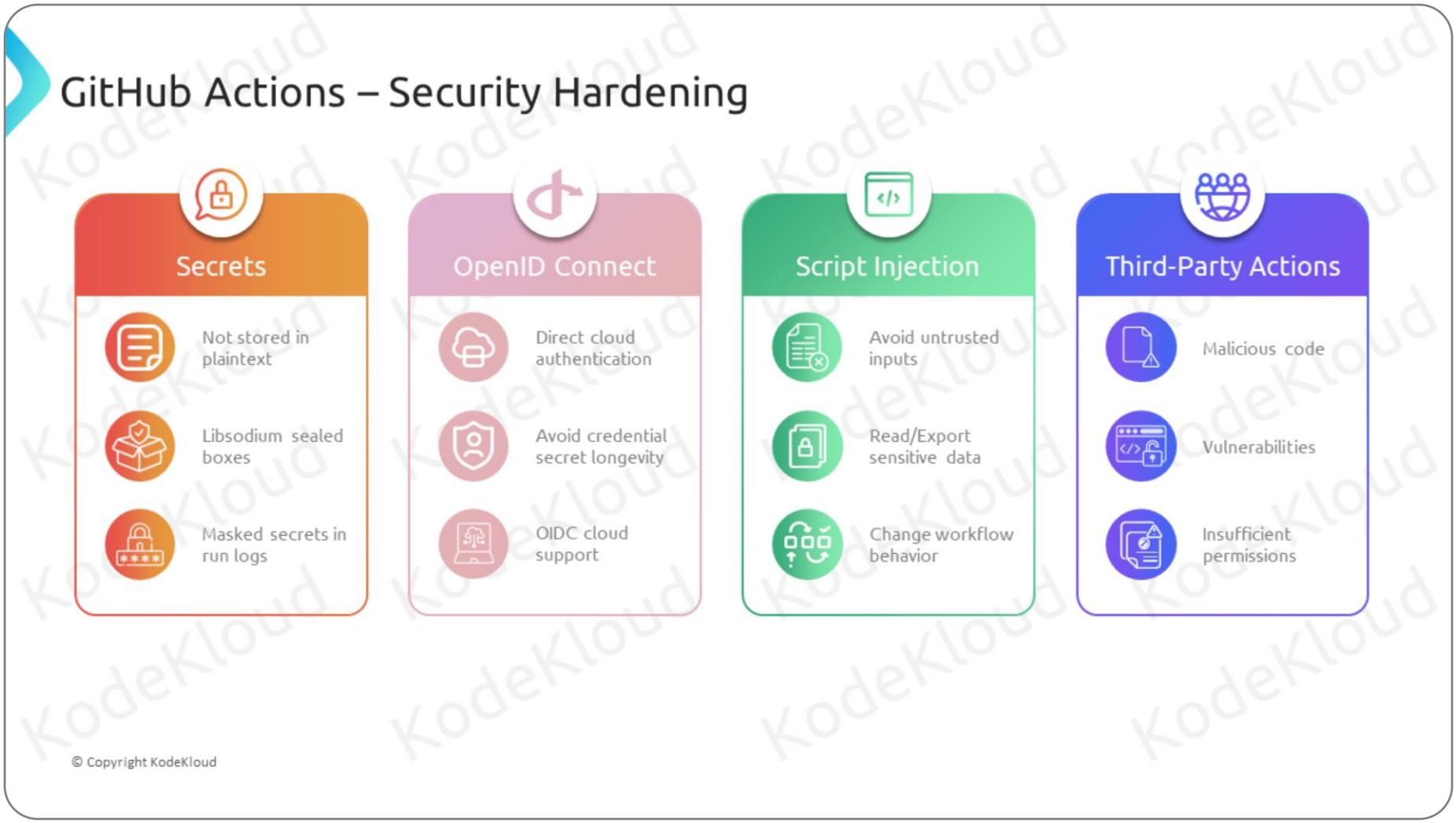
KodeKloud

Follow us on https://kodekloud.com/ to learn more about us.

GitHub Actions – Security Hardening



Sensitive values should never be stored as plaintext in workflow files, but rather as secrets. Secrets can be configured at the organization, repository, or environment level, and allow you to store sensitive information in GitHub.

Secrets use Libsodium sealed boxes, so that they are encrypted before reaching GitHub. This occurs when the secret is submitted using the UI or through the REST API.

To help prevent accidental disclosure, GitHub uses a mechanism that attempts to mask any secrets that appear in run logs.

===

OpenID Connect allows your workflows to exchange short-lived tokens directly from your cloud provider.

GitHub Actions workflows are often designed to access a cloud provider (such as AWS, Azure, GCP,) in order to deploy software or use the cloud's services. Before the workflow can access these resources, it will supply credentials, such as a password or token, to the cloud provider. These credentials are usually stored as a secret in GitHub.

However, using hardcoded secrets requires you to create credentials in the cloud provider and then duplicate them in GitHub as a secret.

With OpenID Connect (OIDC), you can take a different approach by configuring your workflow to request a short-lived access token directly from the cloud provider. Your cloud provider also needs to support OIDC on their end.

===

Lets understand the risks of script injections

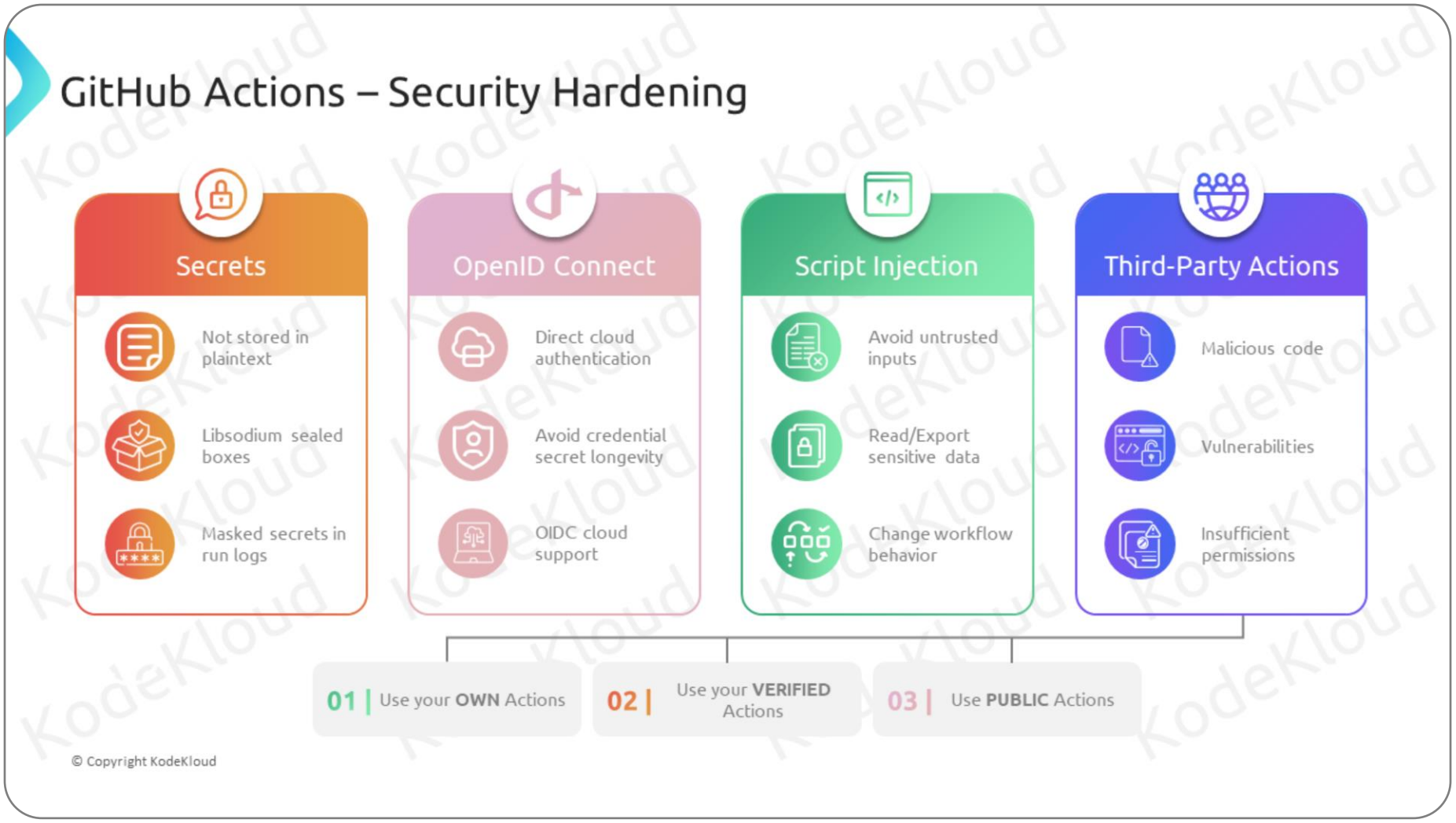
When creating workflows, actions, you should always consider whether your code might execute untrusted input from attackers. This can occur when an attacker adds malicious commands and scripts to a context. When your workflow runs, those strings might be interpreted as code which is then executed on the runner.

To expose secrets and sensitive data and could also change behaviour of workflows/actions

There are a number of different approaches available to help you mitigate the risk of script injection, we will see one of it in an upcoming session.

===

There are a number of security issues that can arise when using third-party actions in GitHub Actions. Some of the most common issues include:
•**Malicious code:** Third-party actions could be malicious and contain code that could compromise your repository.
•**Vulnerabilities**: Third-party actions could have vulnerabilities that could be exploited by attackers.
•**Insufficient permissions:** Third-party actions may request more permissions than they need which might again compromise your Github account.



To mitigate the issues w.r.t actions, it is always recommended to

Use your own actions where were applicable, but this approach requires considerable amount of effort as the actions needs to created and maintained by you.

The 2nd option is to use a combination of own actions and actions which are verified by Github.
Verified actions are third-party actions that have been verified by GitHub. This means that GitHub has reviewed the action code and determined that it is safe and does what it is supposed to do.
Verified actions are identified by a blue checkmark next to the action name in the GitHub Marketplace.

Finally if there is no option left, we can use the public/community actions with caution.
•it is a good idea to review the action code to make sure that you understand what it is doing.
•Pin the action to a specific commit to prevent malicious updates.
•Use the least privilege principle to Only grant the action the permissions that it needs.

KodeKloud

Follow us on https://kodekloud.com/ to learn more about us.