Vetting Process for 3rd Party Dependencies

This process was approved by TSC vote on 7/22/20.

The process to objectively assess the security risk of 3rd party open source components or dependencies is outlined with consideration of the legacy way of performing the assessment, as well as the new process discussed within the project during the Hanoi development time frame.

The Process

The process should take into consideration relevant data such as the project's age, popularity / maturity, evidence of security practices, recent commit history, diversity of committers, established CVE practices, or other observable evidence. In terms of licensing compliance, ideal process should also consider the license associated to the component as well.

Additionally, the ideal process should take into account the following scenarios:

1. Existing Code (Skeletons in the Closet)
2. Code in Holding (Analysis of code before it is accepted)
3. Pull Request with new dependency

Developer Process for Each Use Case

Use Case	Process	Tools	Applicability
1.) Existing Code (Skeletons in the Closet)	Automated scan within build automation via Snyk CLI Scans of the published Docker images via Snyk with notifications to SIR Team members / Snyk Administrators Working groups should review any issues identified via the build automation tools and address within the context of the working group reviews.	Snyk Community Bridge Advanced Snyk Reports Clair	Scan automation occurs within the build - PR merge to master
2.) Code in Holding (Analysis of code before it is accepted)	Review catalog of approved packages in this wiki. Compare that list to the list being submitted. Provide the summary of differences to include the list of new packages and why they are needed.	Complete the paper study for each package. See paper study process as written for use case 3.	When considering code that is under consideration for moving into the main EdgeX Foundry Org, out of holding

3.) Pull Request with new dependency	Submitter of a Pull Request (PR) will complete the Pull Request template to include any new changes that introduce dependency changes (e.g. imports or go module dependencies)
	
	The standard Pull Request template includes a question that asks - *"Are there any new imports or modules? If so, what are they used for and why?"*
	
	Submitter of the PR will add a `dependency` label to the pull request.
	
	If the dependency is security related, the submitter will add the `security-review` label to the PR so a member of the Security WG can help review.
	
	Submitter should include scan results which include consideration of compliance (license) as well as security vulnerability (e.g. CVE) data, that can be reviewed by a Security WG member.
	
	Note: Reviewers will see one of the changed files is go.mod for Go projects.
		Run this command at the root of your repo
		
		GO111MODULE=on go list -m all
		
		GO111MODULE=on go mod graph
		
		For a PR with new dependencies, the submitter of the PR will complete a manual paper study to collect the following data points for review:
		
		• **Total increase in new imports**: (count)
		Does the new import introduce additional import dependencies, if so, how many?
		  ◦ Ensure that **every** one of the new dependencies is checked for the same criteria.
		• **Releases/Tags**: (count)
		  ◦ We should avoid new imports that have never had a release and/or tag. How many is too few, this is a judgement call and probably involves also considering how long ago the last release was, and how far apart releases have been done.
		• **Contributors**: (count)
		• **License** - what is the license, and is it Apache 2.0 compatible?
		• **Stars/Forks/Watchers**: (count)
		  ◦ These are all indications of how wide-spread the package is used.
		• **godoc.org metrics**: (count)
		  ◦ The individual godoc pages hosted by godocs.org include metrics at the base of the page which indicate how many packages import the package
		• **Subjective opinion of the reviewers** – at the end of the day, we rely on our reviewers to vet new code. Reviewers should give thought to whether the code is improving our project, whether we'd be better off to implement the functionality ourselves, and at the same time considering whether this new import itself comes with too many dependencies (e.g. go-kit).
		
		When submitting the PR, complete the PR template and set the labels using both - `dependency` , `security-review (security components only)`
		• On approval, notify the working group chair to update the catalog of approved packages if required.
			On a Pull Request, whenever there's a new dependency introduced as shown through changes to the go.mod

Approved Go Modules (those in Red are being investigated for replacement - avoid them if possible

See Approved Go Modules/Packages

Process Research

Explore Documentation: Issue-1947