Filling the Gaps in Kubernetes Flavored SLSA with Threat Modeling

Christie Wilson, Google & Priya Wadhwa, Chainguard

KubeCon | CloudNativeCon

Europe 2023

@priyawadhwa16

@bobcatwilson

Intros

Priya Wadhwa
Software Engineer
Chainguard

Christie Wilson
Software Engineer
Google

Running CI/CD on kubernetes?

- Need to know the threats and address them
- SLSA gets you most of the way there
- K8s introduces some new unique threats

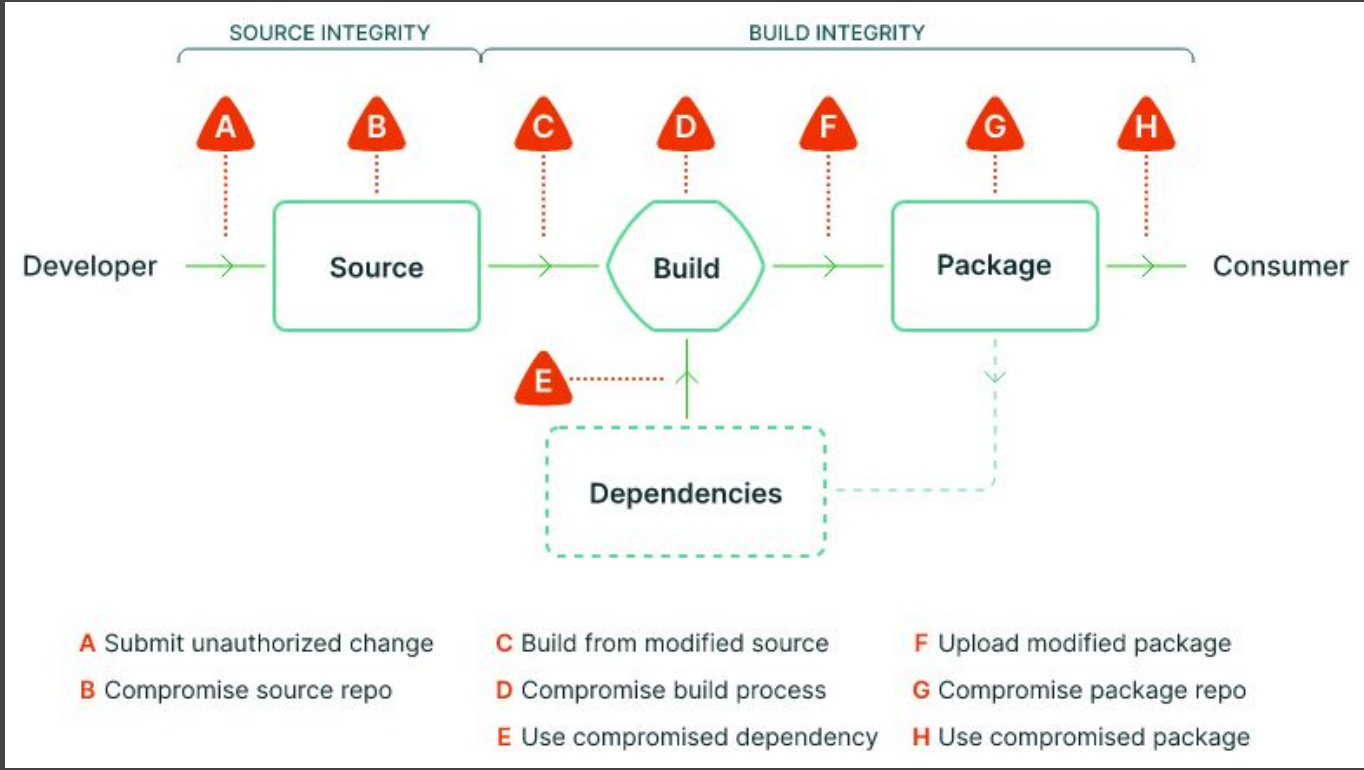
SLSA + k8s Threat Model

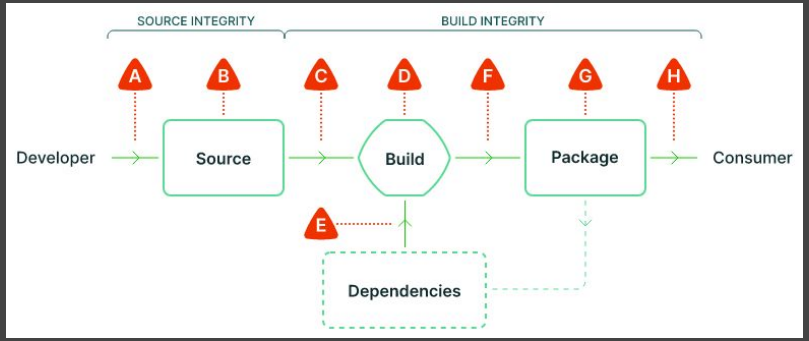
= Secure supply chain

- Comply with SLSA
- Fill the rest of the gaps

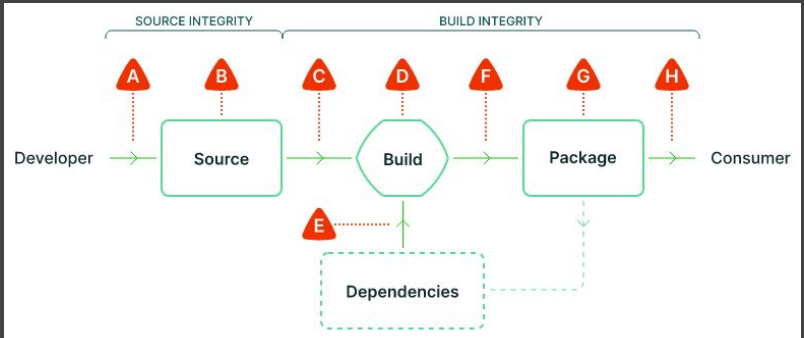
SLSA's Threat Model

SLSA Threat Model





A	Unauthorized changes in repo	• Former SLSA 0.1 requirement for two-person review helped • Future SLSA Source track may help
B	Compromised source repo	• Need to protect the source repo
C	Build from modified source	• SLSA compliant provenance will identify the sources used
D	Compromise build process	• Make it difficult to compromise the build system
E	Compromised dependencies	• Can mitigate by verifying dependency provenance before use
F	Upload a package modified outside of build process	• Verifiable SLSA provenance will be provided for packages that are built by the build process
G	Compromised package repo	• Can mitigate by verifying dependency provenance before use
H	Use a compromised package	• Can mitigate by verifying package provenance before use



When you're using stuff

Verify provenance
Generate provenance and build with a secure build system

When you're making stuff

C	Build from modified source	Verify the provenance
D	Compromise build process	Use a secure build system
E	Compromised dependencies	Verify the provenance
F	Upload a package modified outside of build process	Verify the provenance
G	Compromised package repo	Verify the provenance
H	Use a compromised package	Verify the provenance

@priyawadhwa16

@bobcatwilson

SLSA 1.0

Generate provenance and build with a secure build system

Provenance generation	Exists	L1
	Authentic	L2
	Non-forgeable	L3
Isolation Strength	Build service	L2
	Ephemeral and isolated	L3

- Cloud Native CI/CD platform built on Kubernetes

How Tekton does SLSA: Provenance

- Tekton generates provenance with the optional Tekton Chains service
- Authentic and non-forgeable (or is it?)

Provenance generation	Exists	L1
	Authentic	L2
	Non-forgeable	L3

How Tekton does SLSA: Isolation

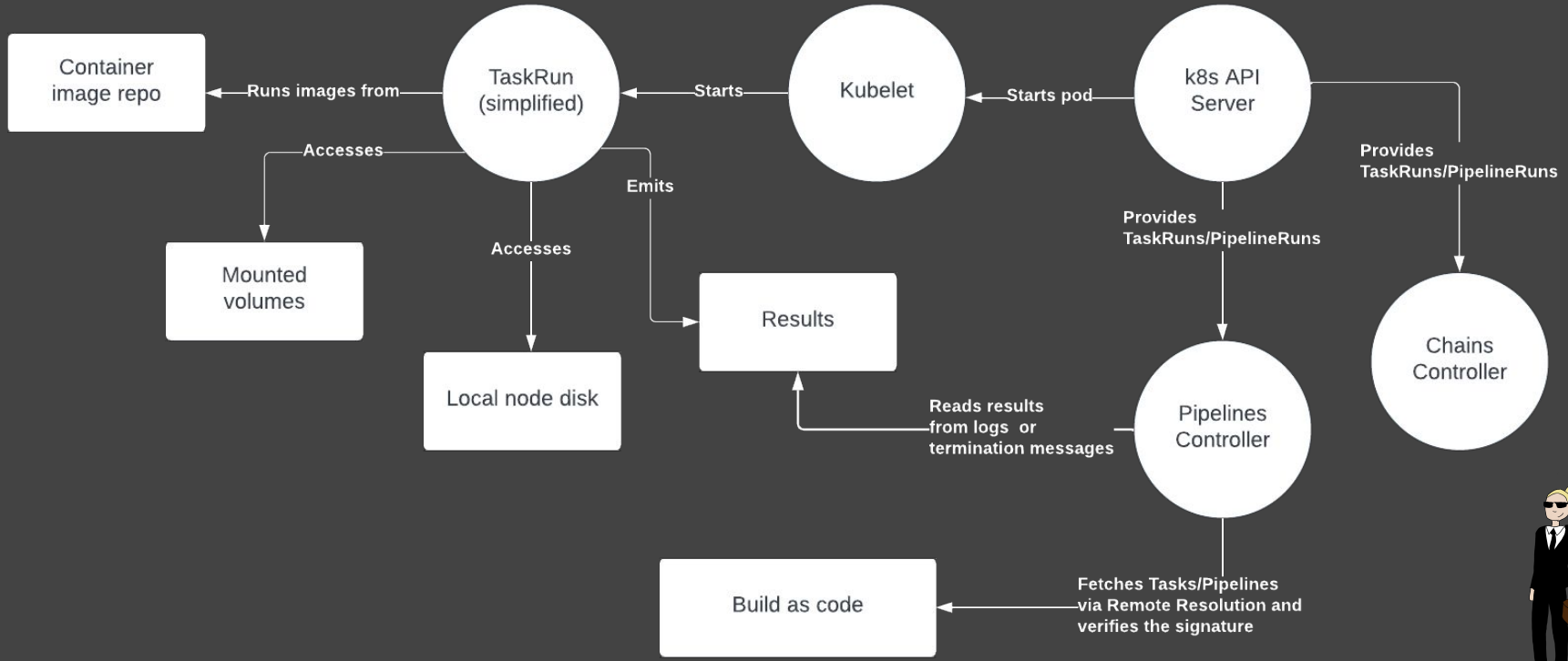
- Ephemeral and isolated execution via Kubernetes? (or is it?)

Isolation Strength	Build service	L2
	Ephemeral and isolated	L3

Zooming in on the build process

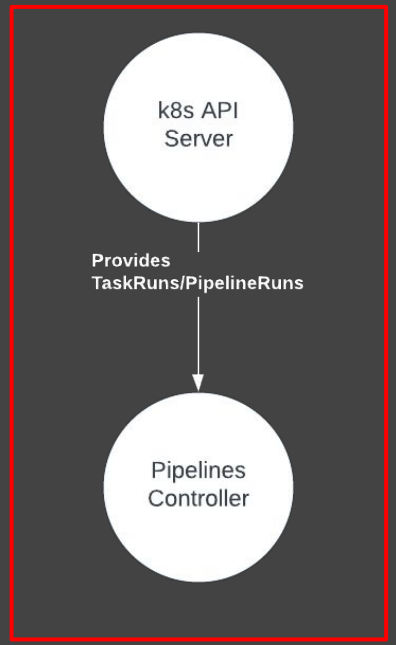
SLSA Threat Model: The build process

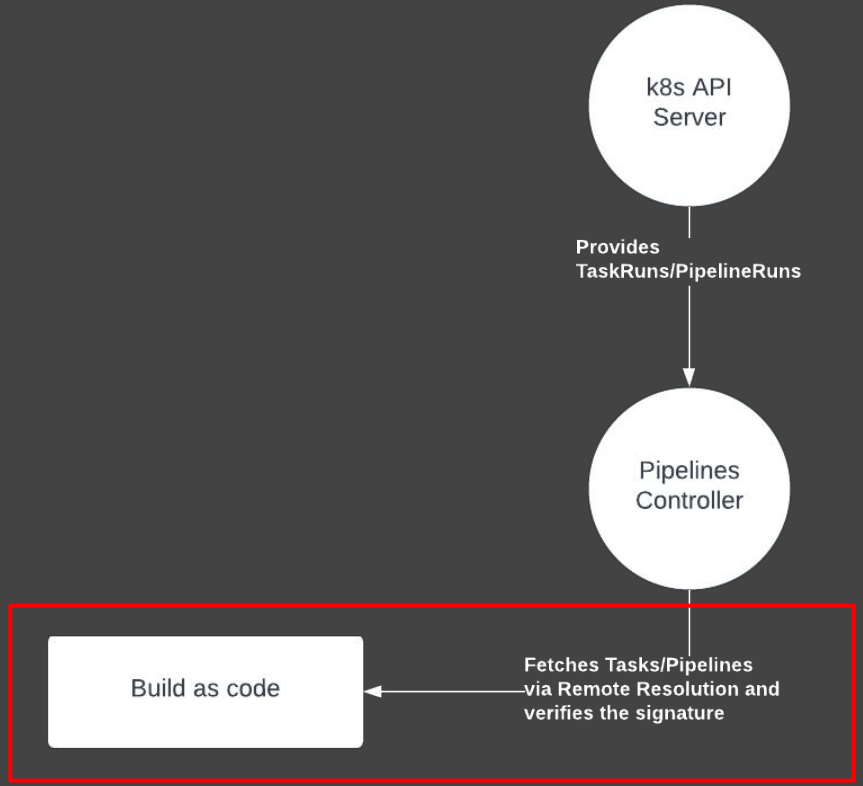
Overview of Tekton execution + provenance generation

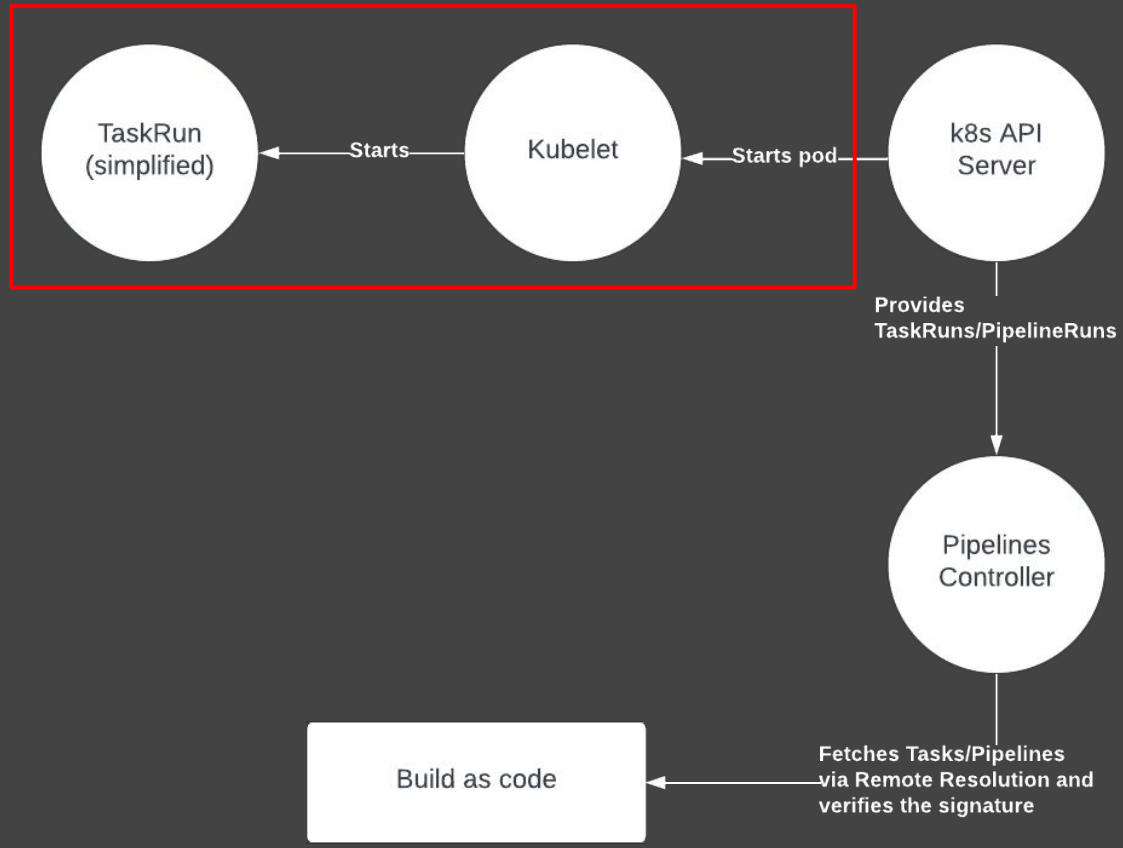


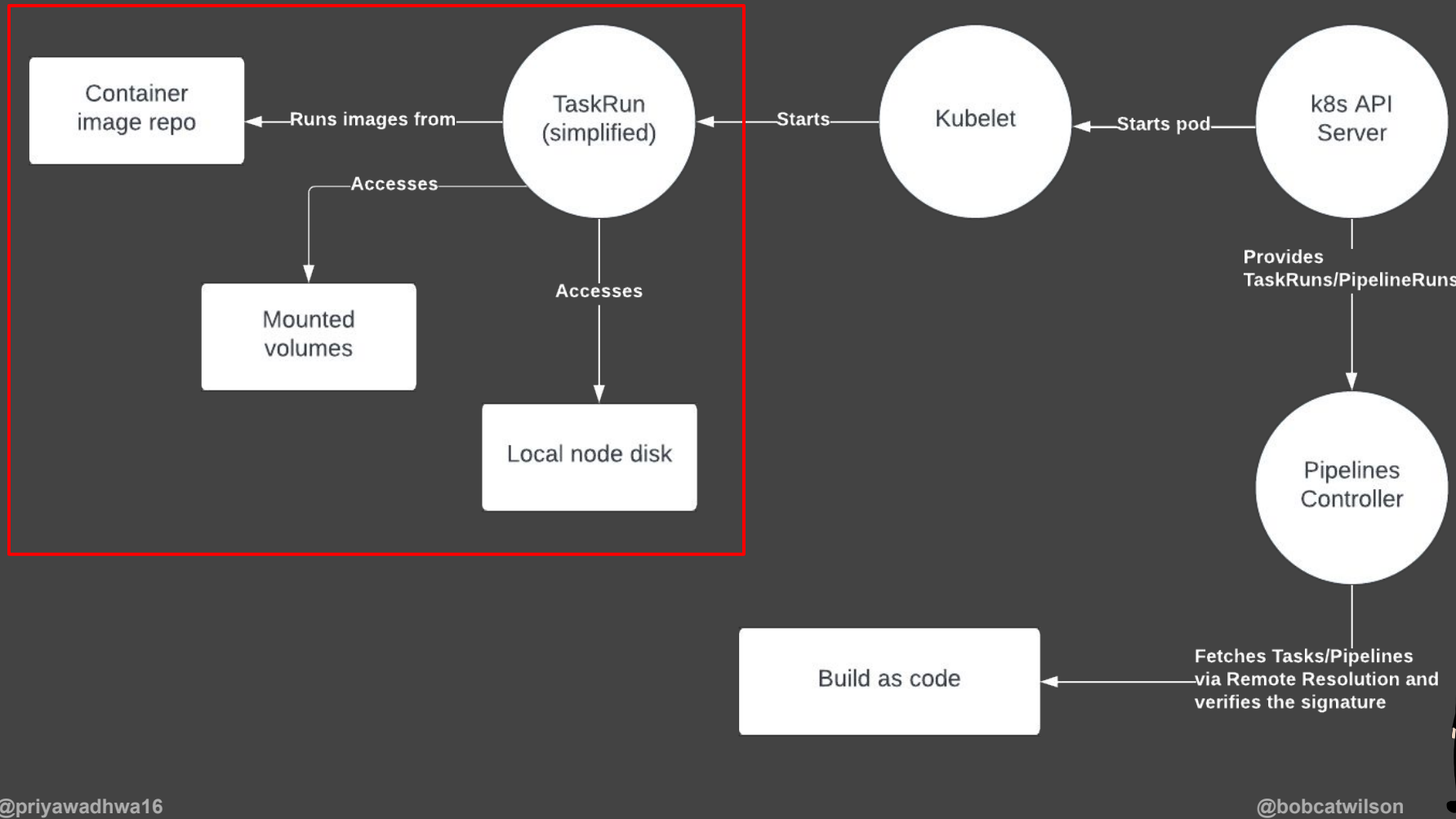
@priyawadhwa16

@bobcatwilson


k8s API Server
Provides TaskRuns/PipelineRuns
Pipelines Controller

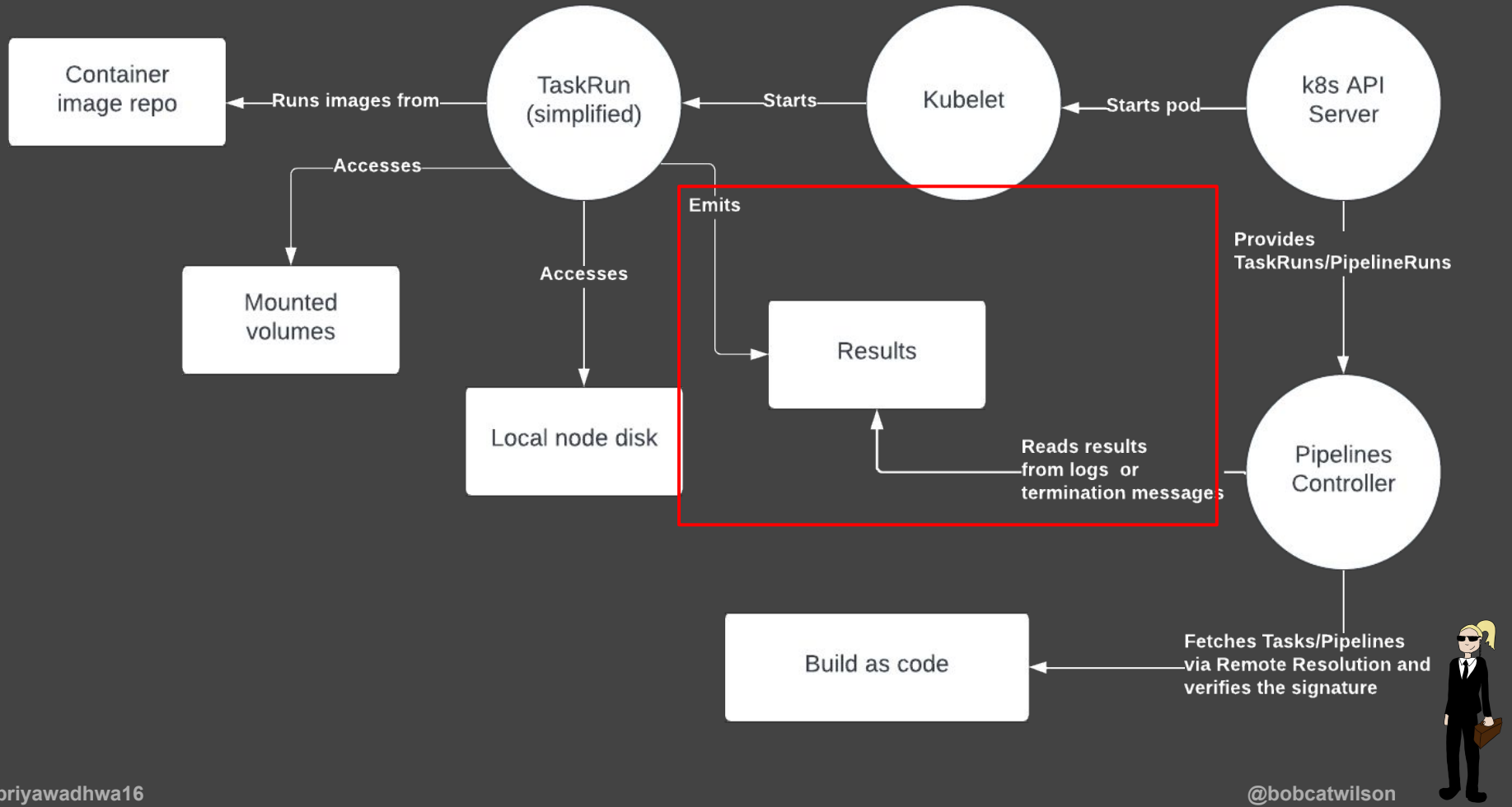

k8s API Server
Provides TaskRuns/PipelineRuns
Pipelines Controller
Build as code
Fetches Tasks/Pipelines via Remote Resolution and verifies the signature


TaskRun
(simplified)
Starts
Kubelet
Starts pod
k8s API
Server
Provides
TaskRuns/PipelineRuns
Pipelines
Controller
Fetches Tasks/Pipelines
via Remote Resolution and
verifies the signature
Build as code



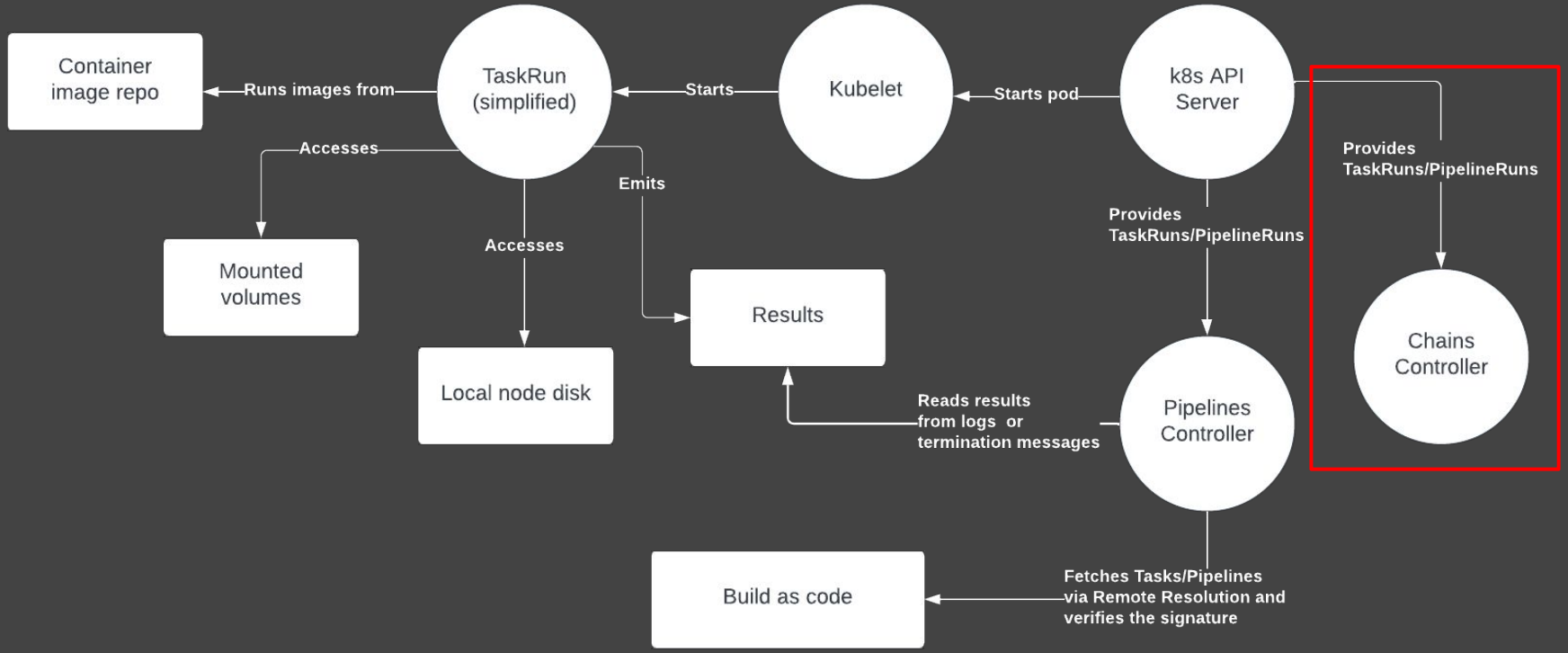
@priyawadhwa16

@bobcatwilson

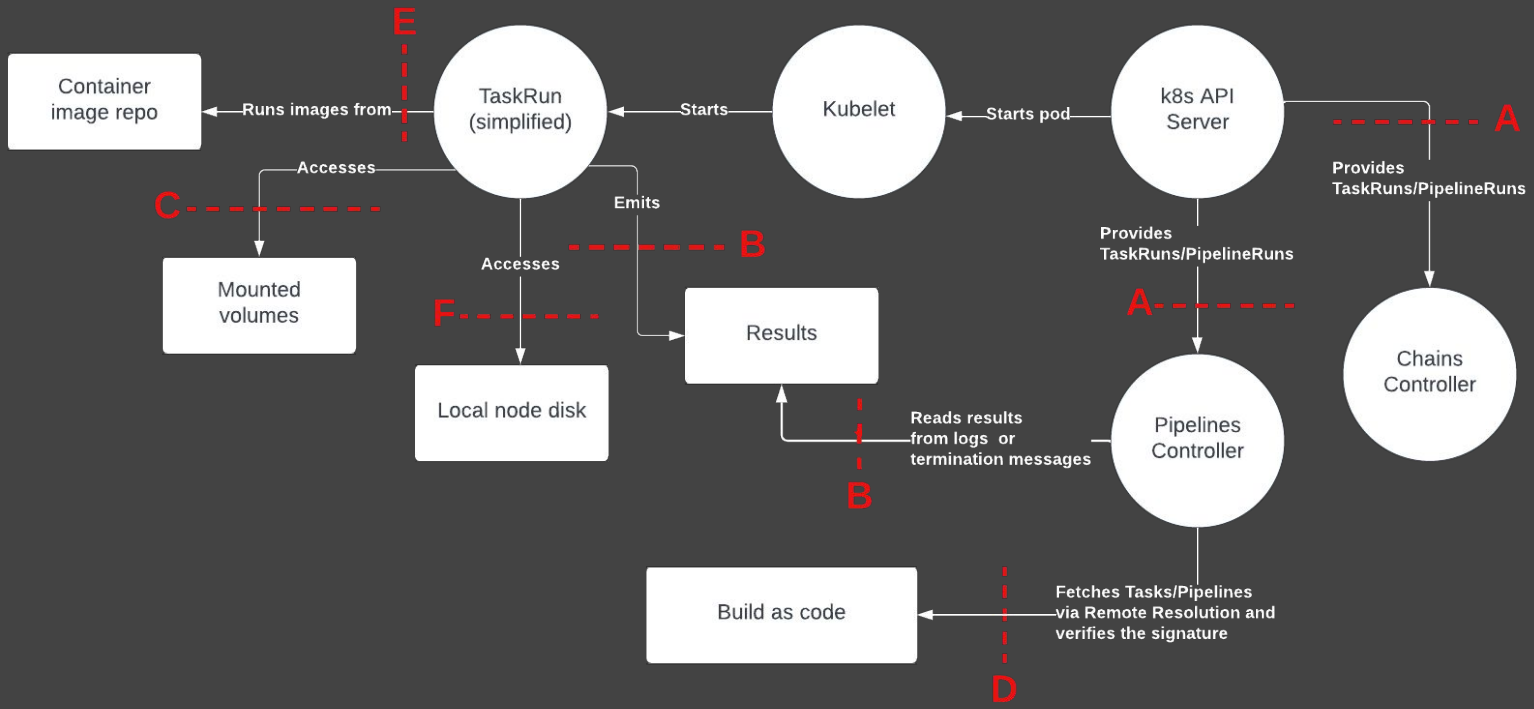


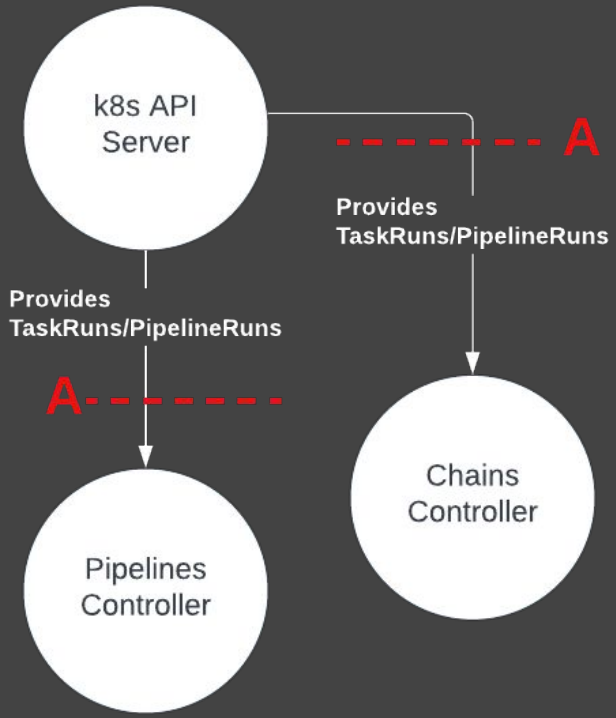
@priyawadhwa16

@bobcatwilson

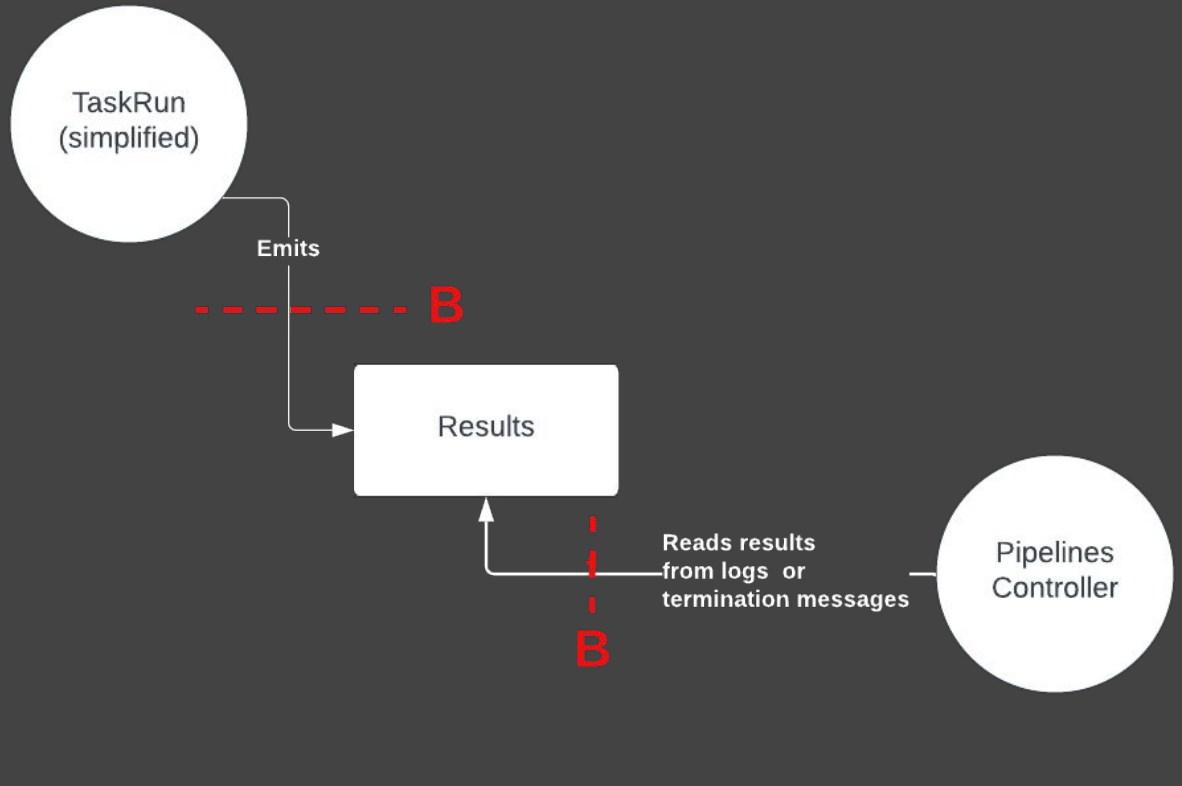

Container image repo
Runs images from
TaskRun (simplified)
Starts
Kubelet
Starts pod
k8s API Server
Accesses
Emits
Provides TaskRuns/PipelineRuns
Provides TaskRuns/PipelineRuns
Mounted volumes
Accesses
Results
Chains Controller
Local node disk
Reads results from logs or termination messages
Pipelines Controller
Build as code
Fetches Tasks/Pipelines via Remote Resolution and verifies the signature

Threat model for Tekton on k8s

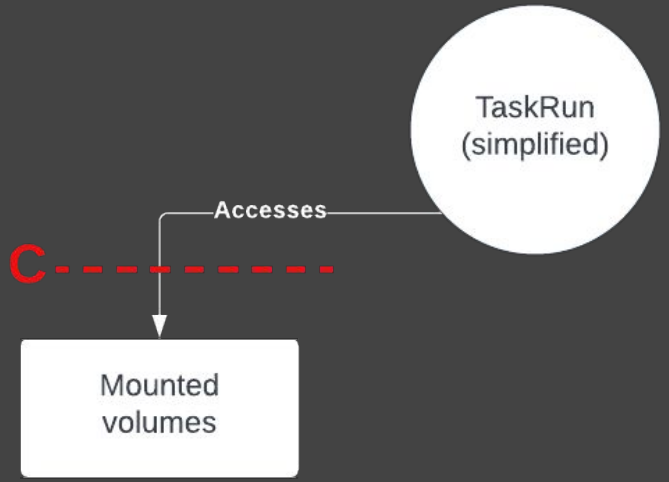




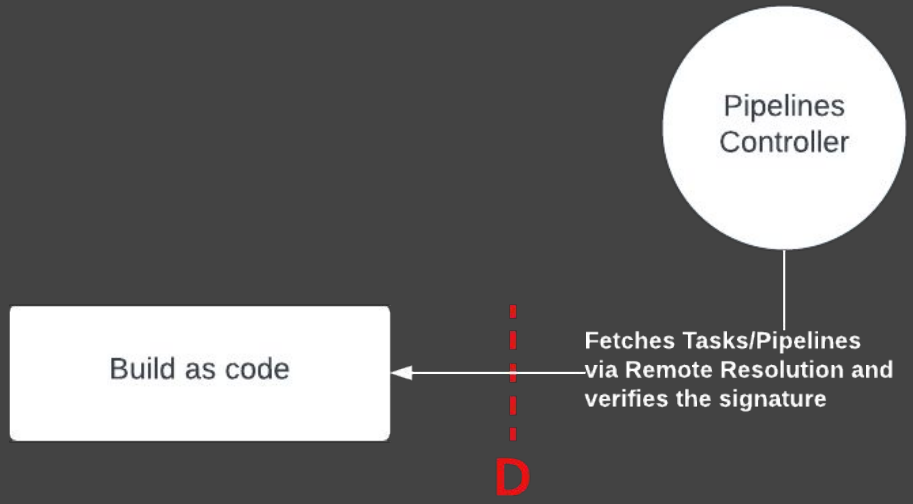
A	Mutated CRDs (TaskRun, PipelineRun, ResolutionRequest)	• CRD status should only be updated by Tekton controllers • Can change params and results that were used, success / failure, what Tasks were run, etc.



B	Mutated results	• Should only be reported by the pod (TaskRun) that ran the Task emitting the results • Can change reported source that was fetched, URIs and digests of built artifacts

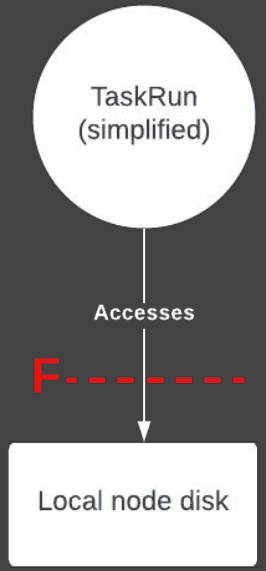


C	Mutated workspaces	• Should only be written to by the intended Tasks



D	Compromised Task or Pipeline specs	• Version control (or Tekton Bundle in OCI registry) should be the source of truth

E	Compromised step images	• Fetched image may not be what the Task author intended



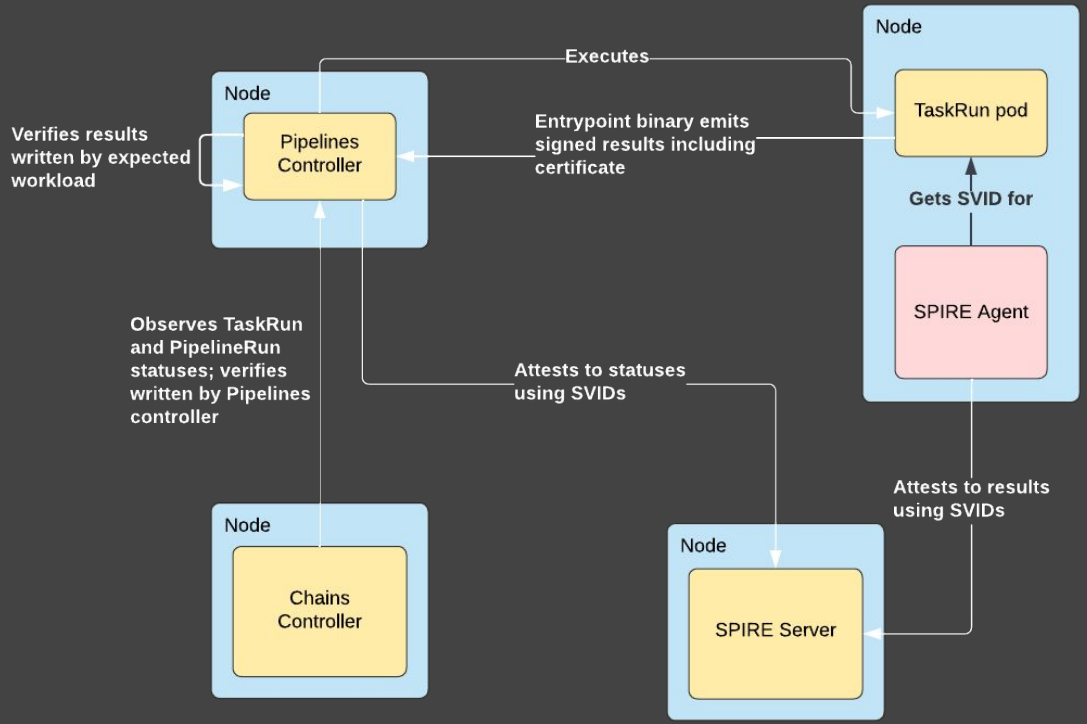
F	Kubernetes pod breakout	• Task shouldn't have unintended side effects on the underlying node or on the cluster

Filling the gaps

Threat: Mutated CRDs
Threat: Mutated Results

- CRD status should only be updated by Tekton controllers
  - Can change params and results that were used, success / failure, what Tasks were run, etc.
- Should only be reported by the pod (TaskRun) that ran the Task emitting the results
  - Can change reported source that was fetched, URIs and digests of built artifacts

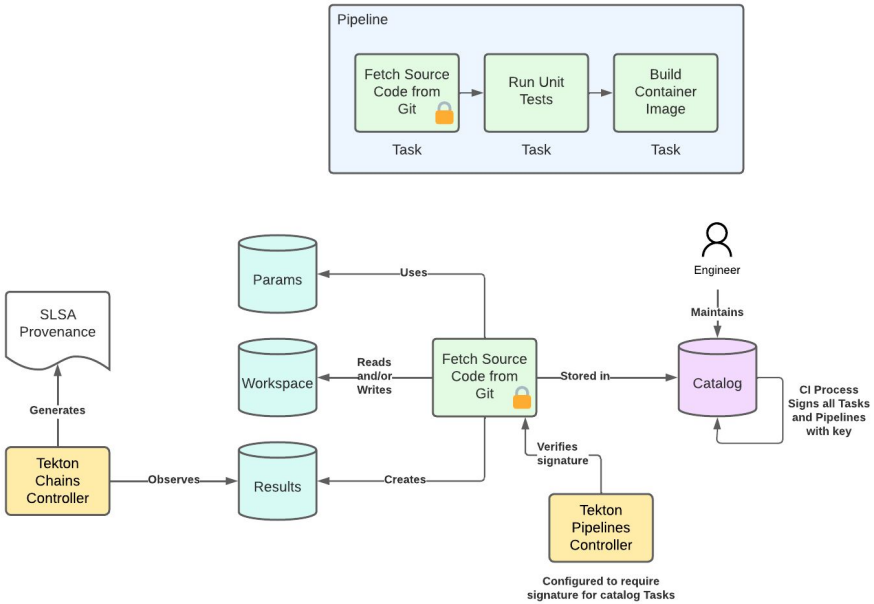
Threat: Mutated CRDs
Threat: Mutated Results
Solution: SPIRE



Threat: Compromised Task or Pipeline specs
Threat: Compromised step images

- Version control (or OCI registry) should be the source of truth
- Fetched image may not be what the Task author intended

Threat: Compromised Task or Pipeline specs
Threat: Compromised step images
Solution: Trusted Resources

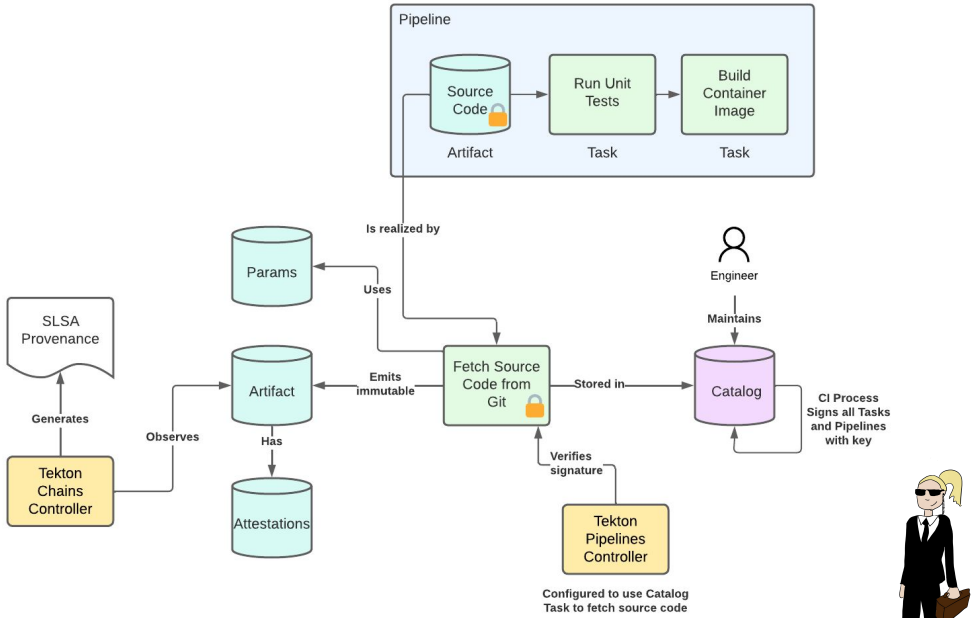


Threat: Mutated Workspaces

- Should only be written to by the intended Tasks

Threat: Mutated Workspaces
Solution: Ephemeral volumes, Tekton Artifacts

- Volumes created and destroyed for each PipelineRun
- Tekton Artifacts (WIP): immutable abstraction



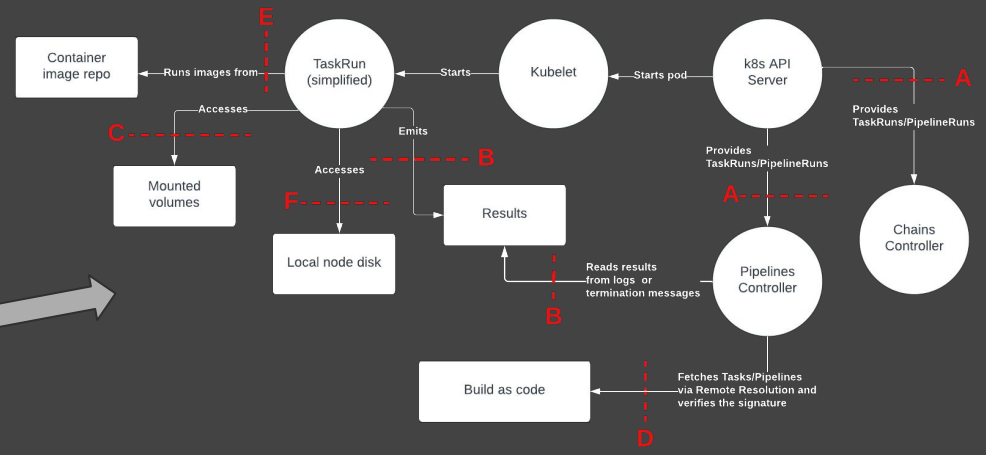
Threat: Kubernetes pod breakout

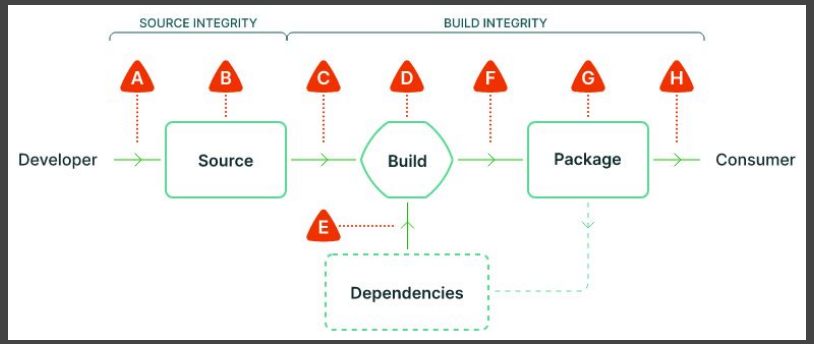
- Task shouldn't have unintended side effects on the underlying node or on the cluster

Threat: Kubernetes pod breakout
Solution: Sandboxed execution

- Do not allow privileged execution; use alternative tool for building Docker images such as Kaniko
- GKE Sandbox, kata containers, VMs instead of pods

C+E+ F+G+ H	Verify provenance
D	Generate provenance and build with a secure build system





A+B	Mutated CRDs + results	Sign with SPIRE
C	Mutated workspaces	Ephemeral pipeline specific volumes Tekton Artifacts (immutable)
D+E	Compromised Task or Pipeline specs, or step images	Trusted Resources Specify images by digest
F	Kubernetes pod breakout	Sandboxed execution, kata containers, VM

What's not addressed

- Verifying image provenance as part of TaskRun execution
- How to meet isolation requirements with volumes, but also support caching
- Integrity of the cluster itself (including kubelet, SPIRE agent, API server etc)
  - How it is built
  - How it is run (attesting on startup)

Fill the gaps

- SLSA helps
- Threat modelling helps

Tekton fills a lot of gaps

- Looking for a build system that was created with SLSA in mind? Consider Tekton

@priyawadhwa16

@bobcatwilson

Resources and credits

- Tekton Supply Chain Security Working group: https://github.com/tektoncd/community/blob/main/working-groups.md#software-supply-chain-security-s3c
- Chitrang Patel @ Google for threat modeling
- Christie wrote a book! Grokking Continuous Delivery
  - 48% off with code KUBECONAMS (http://mng.bz/pd4w)

KubeCon | CloudNativeCon

Europe 2023