Northern Tech on the Rock
Jan 10, 2026

TypeScript for Confident Deploys and Safe Secrets

Thada Wangthammang

Platform Team Lead (DevOps)
T. T. Software Solution / Itron

Thada Wangthammang, (Mild)
Platform Team Lead (DevOps)

T.T. Software Solution / Itron

T.T.Software
Web Application Expert

Microsoft MVP, Microsoft Azure, Cloud Native
Admin of ThaiType

MVP Microsoft® Most Valuable Professional

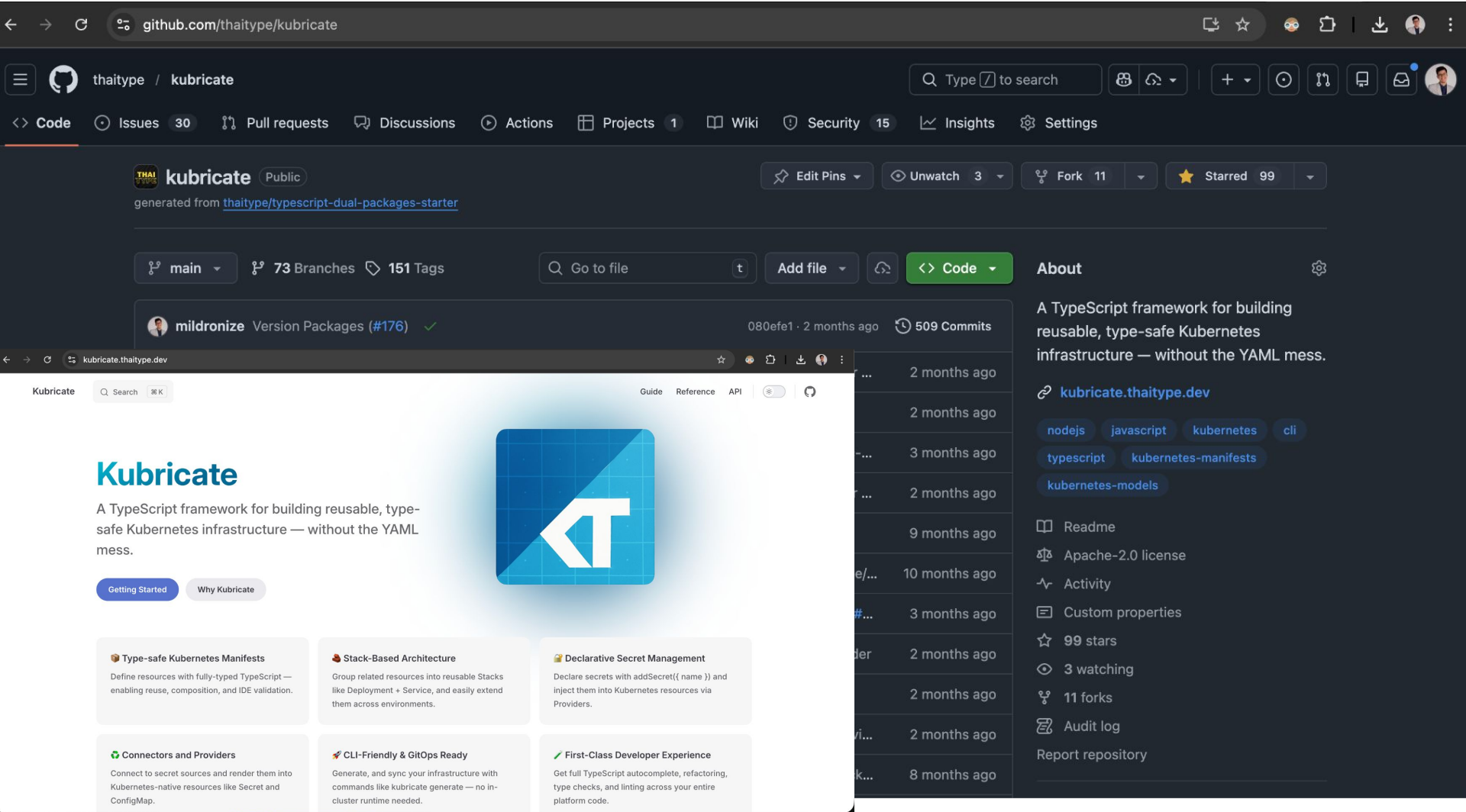
thai type

ไทยไทป์

8.8K followers · 231 following

Professional dashboard | Edit | Advertise

Posts | About | Mentions | Reels | Photos | Live | More


github.com/thaitype/kubricate
thaitype / kubricate
Type / to search
Code
Issues 30
Pull requests
Discussions
Actions
Projects 1
Wiki
Security 15
Insights
Settings
kubricate
Public
generated from thaitype/typescript-dual-packages-starter
Edit Pins
Unwatch 3
Fork 11
Starred 99
main
73 Branches
151 Tags
Go to file
t
Add file
Code
About
A TypeScript framework for building reusable, type-safe Kubernetes infrastructure — without the YAML mess.
kubricate.thaitype.dev
nodejs
javascript
kubernetes
cli
typescript
kubernetes-manifests
kubernetes-models
Readme
Apache-2.0 license
Activity
Custom properties
99 stars
3 watching
11 forks
Audit log
Report repository
mildronize Version Packages (#176)
080efe1 · 2 months ago
509 Commits
2 months ago
2 months ago
3 months ago
2 months ago
9 months ago
10 months ago
3 months ago
2 months ago
2 months ago
2 months ago
8 months ago
kubricate.thaitype.dev
Kubricate
Search ⌘K
Guide
Reference
API
Kubricate
A TypeScript framework for building reusable, type-safe Kubernetes infrastructure — without the YAML mess.
Getting Started
Why Kubricate
Type-safe Kubernetes Manifests
Define resources with fully-typed TypeScript — enabling reuse, composition, and IDE validation.
Stack-Based Architecture
Group related resources into reusable Stacks like Deployment + Service, and easily extend them across environments.
Declarative Secret Management
Declare secrets with addSecret({ name }) and inject them into Kubernetes resources via Providers.
Connectors and Providers
Connect to secret sources and render them into Kubernetes-native resources like Secret and ConfigMap.
CLI-Friendly & GitOps Ready
Generate, and sync your infrastructure with commands like kubricate generate — no in-cluster runtime needed.
First-Class Developer Experience
Get full TypeScript autocomplete, refactoring, type checks, and linting across your entire platform code.

Prerequisite Installation

- Bun/node.js
- Az cli
- Github account
- Microsoft Authenticator (Your mobile)

Checkout Slide

https://docs.google.com/presentation/d/1BWcvEZYCFBLRNOGvDJ9YNefIBRW2DncbdS4cXvOaaBQ/edit?usp=sharing

How to Participate with this Workshop

Submit your email, i'll invite you into Azure Cloud
(Scan QR code to Submit to google Form)
https://forms.gle/9gE3BdECgrWUFc6M7

1. You'll be invited in into my Azure Tenant (No paid needed)
2. You will be in Azure or Get Deployment Secret

Register Workshop

Talk Outline

- Why we need to care about secret?
- Being Problem Solving
- Big Picture Concept
- Mini Project Introduction
- Deployment Side
- Secrets Side
- Wired With CI/CD
- What’s Next?

What’s include in this Workshop?

- Programming Language: **TypeScript**
- CI & CD: **Github Actions**
- App Deployment: **Azure Container App**
- Secret Store: **Azure KeyVault**

Setting the Expectation

- This workshop is a bridge between what I want to communicate and how you understand it
- You may see tools or concepts you are not familiar with
- That is completely fine
- The goal is not tool mastery, but understanding the idea behind them
- Think of this as a starting point, not a complete solution

Secret Challenge!

- How to **store** the secret?
- How to **revoke** the secret?
- How to **rotate** the secret?
- How to **control access control** to the secret?

https://learn.microsoft.com/en-us/azure/well-architected/security/application-secrets
https://learn.microsoft.com/en-us/azure/key-vault/general/best-practices
https://learn.microsoft.com/en-us/azure/key-vault/secrets/secrets-best-practices

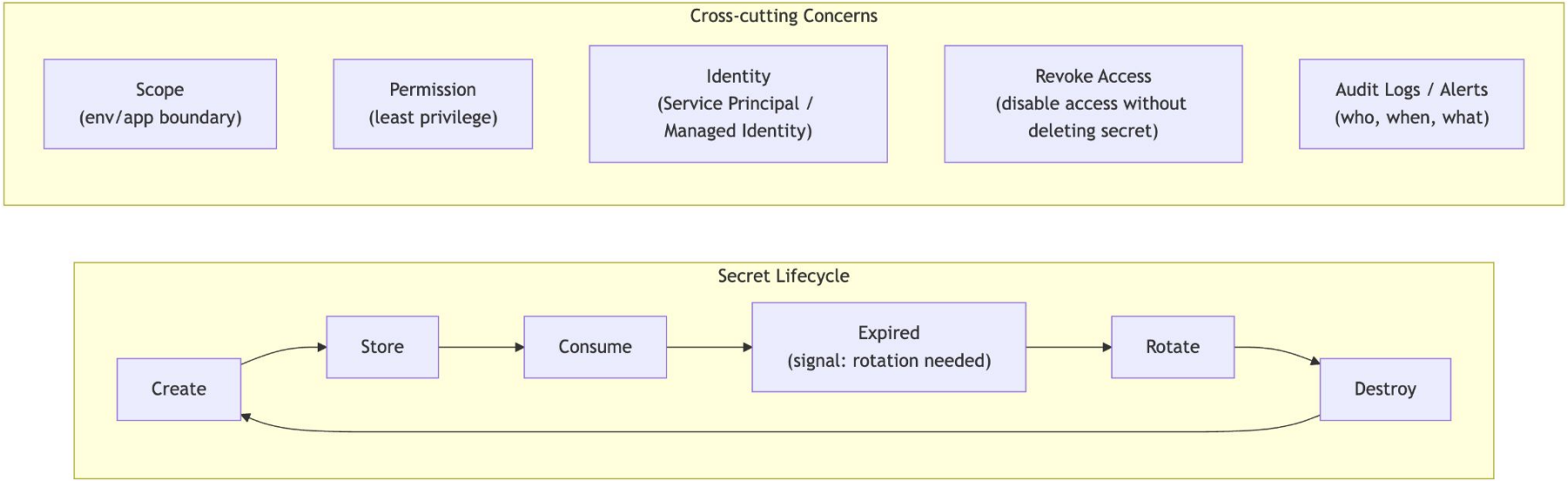
Secret Lifecycle (The Time Dimension)

- **Secrets** are not static values, they have a lifecycle
- A secret is **created**, **stored** securely, and **consumed** by workloads
- Secrets **expire** as a signal, not a failure
- **Rotation** replaces secrets before they become a risk
- **Destroy** completes the lifecycle and enables clean recreation

Cross-Cutting Security Controls

- **Scope** defines *where* a secret can be used
- **Permission** defines *who* can access or modify it
- **Identity** executes lifecycle actions, not humans
- **Revoke** disables access without deleting the secret
- **Audit logs** and alerts provide traceability and proof

Secret Life Cycle



Summary — Why This Matters

- Secret management is a system, not a manual task
- Lifecycle and access controls must work together
- Security actions should be automated, not ad-hoc
- Automation makes this process predictable and repeatable
- This is the foundation for secure, scalable deployments

When Scale Becomes the Problem

- One or two secrets rarely cause issues
- Problems appear when the number quietly grows
- To reduce complexity, we often widen permission or scope
- Fewer secrets feel easier, but security becomes weaker
- The real solution is not fewer secrets, but better control

Control the System, Not the Count

- Reducing secrets by widening access trades security for convenience
- Manual control does not scale with growing systems
- Code allows us to manage many secrets safely
- Rules are enforced consistently, not selectively
- Automation lets security scale without compromise

Control the System, Not the Count

- Reducing secrets by widening access trades security for convenience
- Manual control does not scale with growing systems
- Code allows us to manage many secrets safely
- **Rules** are enforced consistently, not selectively
- Automation lets security scale without compromise

“The problem is not having too many secrets.

The problem is having no system to manage them.”

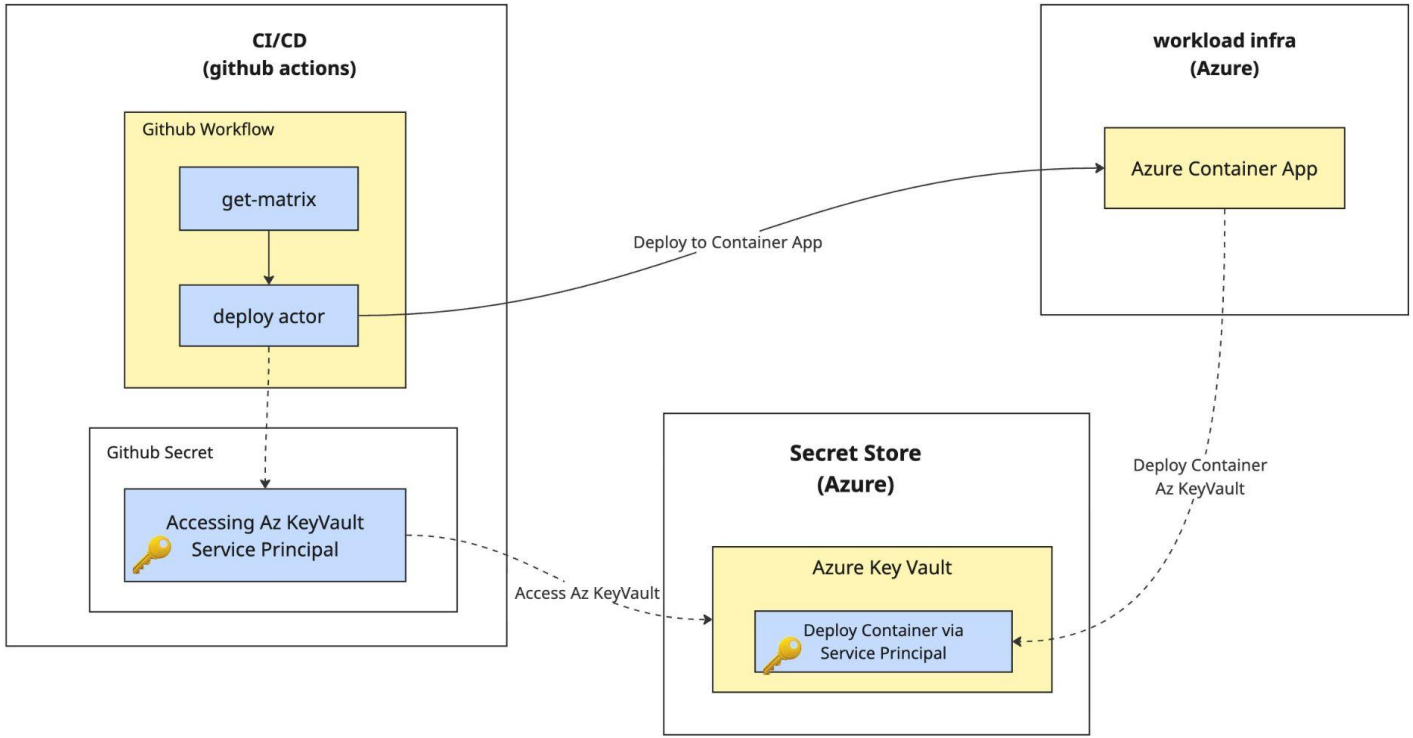
Rules Should Live Where Change Is Safe

- Rules change over time
- They must be visible and reviewable
- They must fail early, not in production
- They should be easy to refactor
- This rules out documents and ad-hoc scripts

Let’s Start Workshop

- Checkout Template Repo:
  https://github.com/mildronize/ts-confident-deploy-and-secret

How Secret Work in the workshop?



How to Get Secret for Github Actions (workshop)

- Go to Azure Key Vault: **kv-ntotr-shared**
- Go to secret name: **github-actions-secret-link**

{

"clientId": "b32711c0-bd14-4e28-9238-291211eb1e87",

"clientSecret": "l418Q~bDVTb652VQHEMbIiNpGrQFz1xrzFBQPdeC",

"subscriptionId": "0c249ac1-38ac-4cb4-a429-8b1448de6d8e",

"tenantId": "859c5c45-c82c-4178-b030-23cf68c69b88",

When Applying Secret System

- Organizations apply secret management differently
- The difference is not tooling, but how rules are enforced
- This usually falls into two approaches
- **Soft constraints** and **Hard constraints**
- Choosing between them is a business decision

Soft Constraints

- Rules enforced through policy or verbal agreement
- Depends on people following guidelines
- Pros
  - Fast to introduce
  - Low initial cost
- Cons
  - Relies on human discipline
  - Failure is possible and often invisible

Hard Constraints

- Rules enforced by systems and automation
- Behavior is restricted by design
- Pros
  - Strong security guarantees
  - Scales reliably with system growth
- Cons
  - Higher upfront investment
  - Requires system-level thinking

Why this matters?

- We started with secrets because failures there are **high-impact and visible**
- But the same pattern exists across the **entire organization**
- Rules, handoffs, and approvals live everywhere—not just in security
- DevOps is a culture that **moves rules from people into systems**
- Faster, safer processes are **directly tied to business outcomes**

Closing — Final Thought

- Tools does not scale organizations, systems do
- Manual processes turn into risk as the business grows
- Automation converts intent into reliable execution
- DevOps connects engineering decisions to business outcomes
- This is how speed and safety coexist

Real World System in Practice?

สร้าง Pipeline อย่างปลอดภัยขึ้นด้วย Azure Container App, GitHub...

173 views • 1 year ago

https://youtu.be/bL7FGkqTbbE?si=ENtHLsOiRrsr1W2Z

Resources

- Template Repo: https://github.com/mildronize/ts-confident-deploy-and-secret
- Speaker Repo:

https://github.com/mildronize/ts-confident-deploy-and-secret-speaker

- Download Slide: ไทยไทบ Facebook Page

Feedback me?

https://forms.gle/5B9xW84jpxeugWxE8