
KubeCon
CloudNativeCon
North America 2024

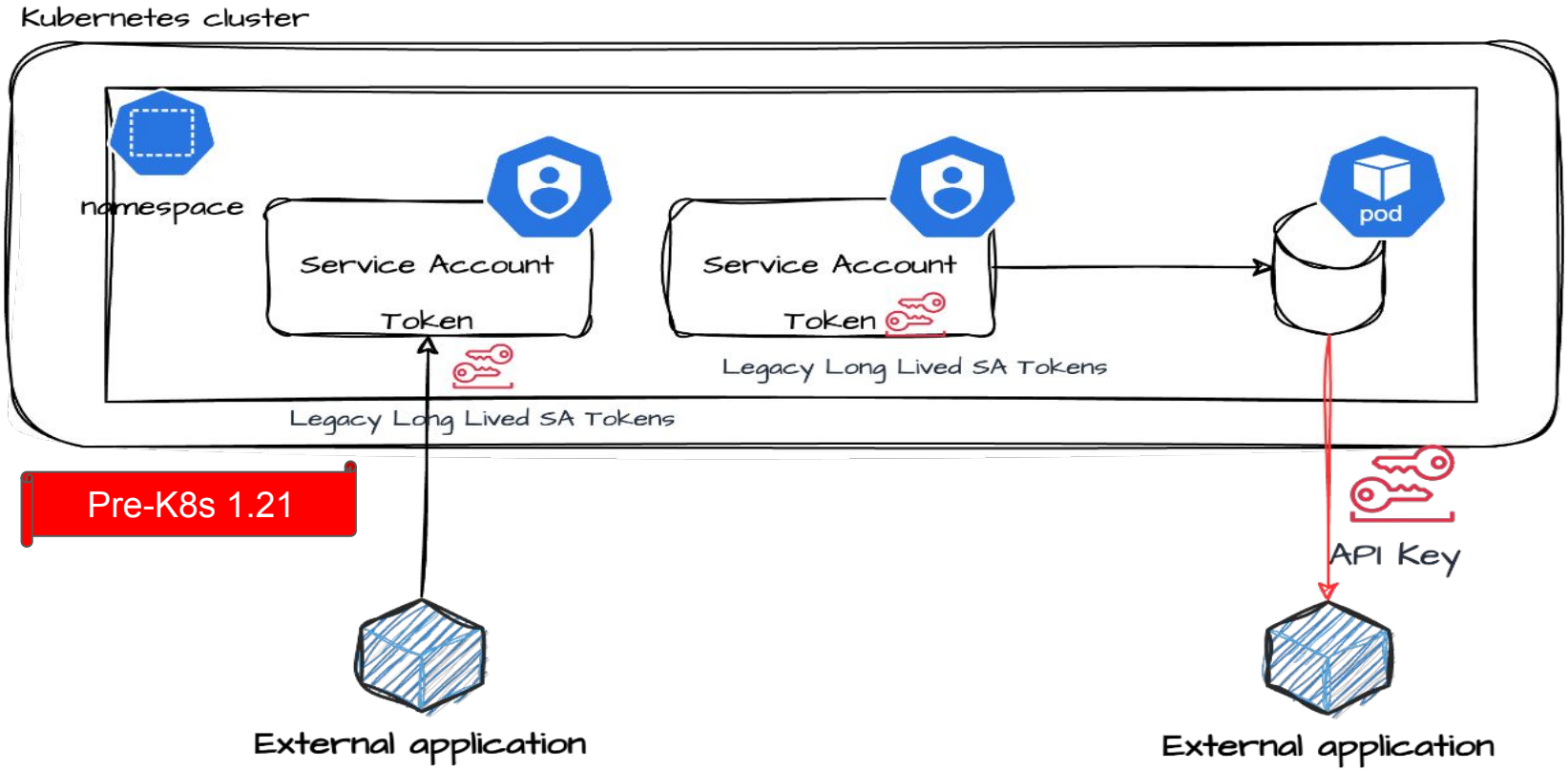
KubeCon | CloudNativeCon

North America 2024

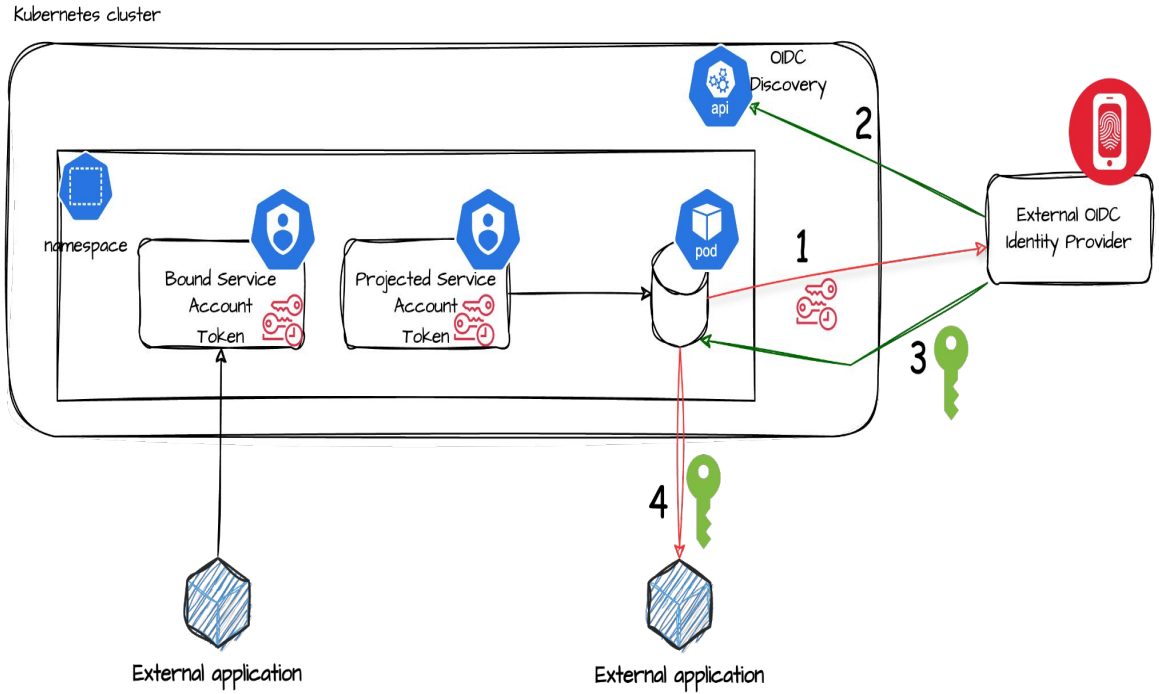
Workload Identity Federation - STOP Using Long-Lived Credentials!

Ben Dronen (Ford Motor Company)
Anjali Telang (Red Hat)

State Of Workload Identity - Pre K8s 1.21



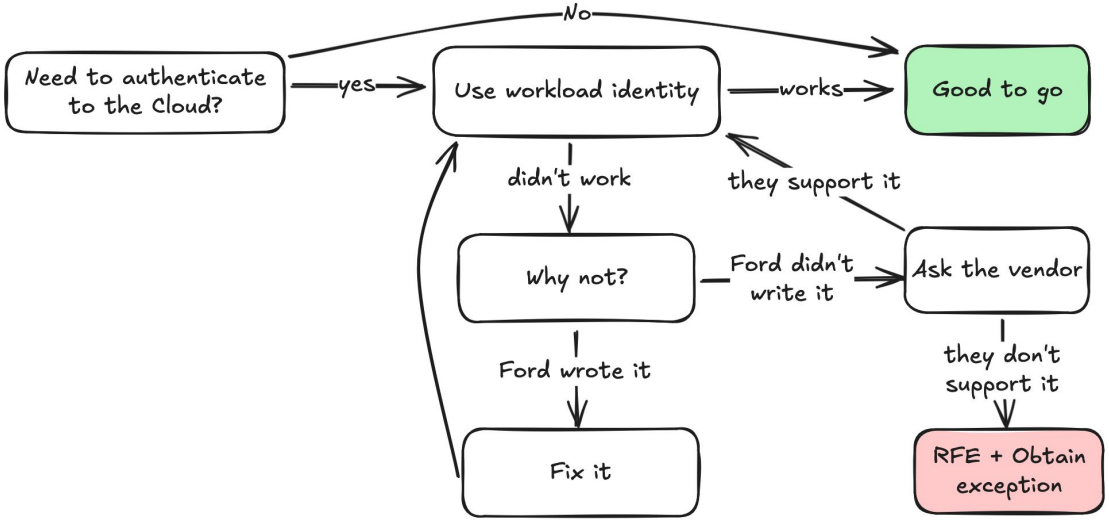
Workload Identity Federation



1. K8s workload authenticating with External App is re-directed to OIDC Identity Provider
2. Identity Provider validates the K8s SA token against configured federation trust relationship
3. Identity Provider provides short-lived, audience-bound OIDC tokens to the K8s Workload
4. Workload uses these to access External App

Ford’s Approach to Workload Identity

- Ford's Cloud journey has ramped up dramatically in past ~4 years
- Prior to this ramp up, services leveraged traditional credentials with long expiry times
- Traditional credentials inevitably expired - causing service interruptions
- Credential management is a pain - especially at scale (over 7,000 Tekton namespaces alone!)
- **Use Workload Identity wherever possible! Whether running in Kubernetes or not!**



Things Ford Uses Workload Identity For

Configuring Workload Identity in Kubernetes

- Public OIDC service account issuer endpoint
- Create Kubernetes service account, know subject claim
  - Takes the format: `system:serviceaccount:<namespace>:<service account>`
- Define the allowed audience for token exchanges
  - Projected service account tokens can only have a single audience!
- Expiry - allowed token validity period varies between Cloud Providers


```
{
  "header": {
    "alg": "RS256",
    "kid": "EH_Eie2RPKNKm50Dz7Q17TFLMhyvGXV68kW0wqHA1CY"
  },
  "payload": {
    "iss":
"https://storage.googleapis.com/dronenb-kubecon-2024-demo",
    "sub": "system:serviceaccount:default:default"
    "aud": [
      "https://storage.googleapis.com/dronenb-kubecon-2024-demo"
    ],
    "exp": 1730710816,
    "iat": 1730707216,
    "jti": "a4635556-9c89-4611-9593-be5cf5d5d41f",
    "kubernetes.io": {
      "namespace": "default",
      "serviceaccount": {
        "name": "default",
        "uid": "1ae8ff9c-5aca-48ae-b1c7-0bfff25ee069"
      }
    },
    "nbf": 1730707216,
  },
  "signature":
"33:e6:cf:73:2f:61:25:07:87:af:30:25:4c:d8:57:46:6b:56:87:21:36:4e:1
a:e1:e7:90:17:e2:ca:65:63:da:f0:f7:f1:c0:6b:16:c5:23:be:81:f8:58:e4:
34:27:fe:8f:d0:d1:d0:2b:76:5a:6d:a6:97:5d:dc:12:84:80:bc:80:60:3a:d0
:4d:34:48:d5:88:2c:48:88:2d:a3:e5:3a:8c:85:72:c2:98:bd:b6:fc:46:0b:9
b:3c:80:a8:75:73:e6:b2:5d:aa:e5:ae:3f:63:0d:35:97:78:53:3f:55:20:ca:
a7:59:10:e3:e1:50:44:49:7d:a3:93:44:59:16:bf:e8:44:a9:c3:25:48:cc:52
:eb:5d:44:d7:62:b6:d8:e4:da:b1:32:1a:e0:bc:f9:98:a8:99:ca:fc:20:2e:8
9:34:49:4e:45:c4:39:25:7c:8b:03:90:e8:c4:9c:98:ce:6a:15:23:c5:9b:5f:
b7:72:8f:6e:68:32:89:08:f3:6d:6d:18:67:cf:a9:98:2f:97:c5:e0:18:a6:16
:a1:42:0a:56:95:e8:6e:44:9f:10:4a:be:fd:94:24:04:49:7d:e7:1f:2c:e2:1
f:f9:bb:71:52:ce:4b:1e:e4:95:6e:e6:4c:d6:d8:ee:11:e0:1d:b4:e1:0e:38:
55:11:e8:3e:07:43:2b"
}
```


Example service account token created with `kubectl create token --namespace default default | jc --jwt -p`

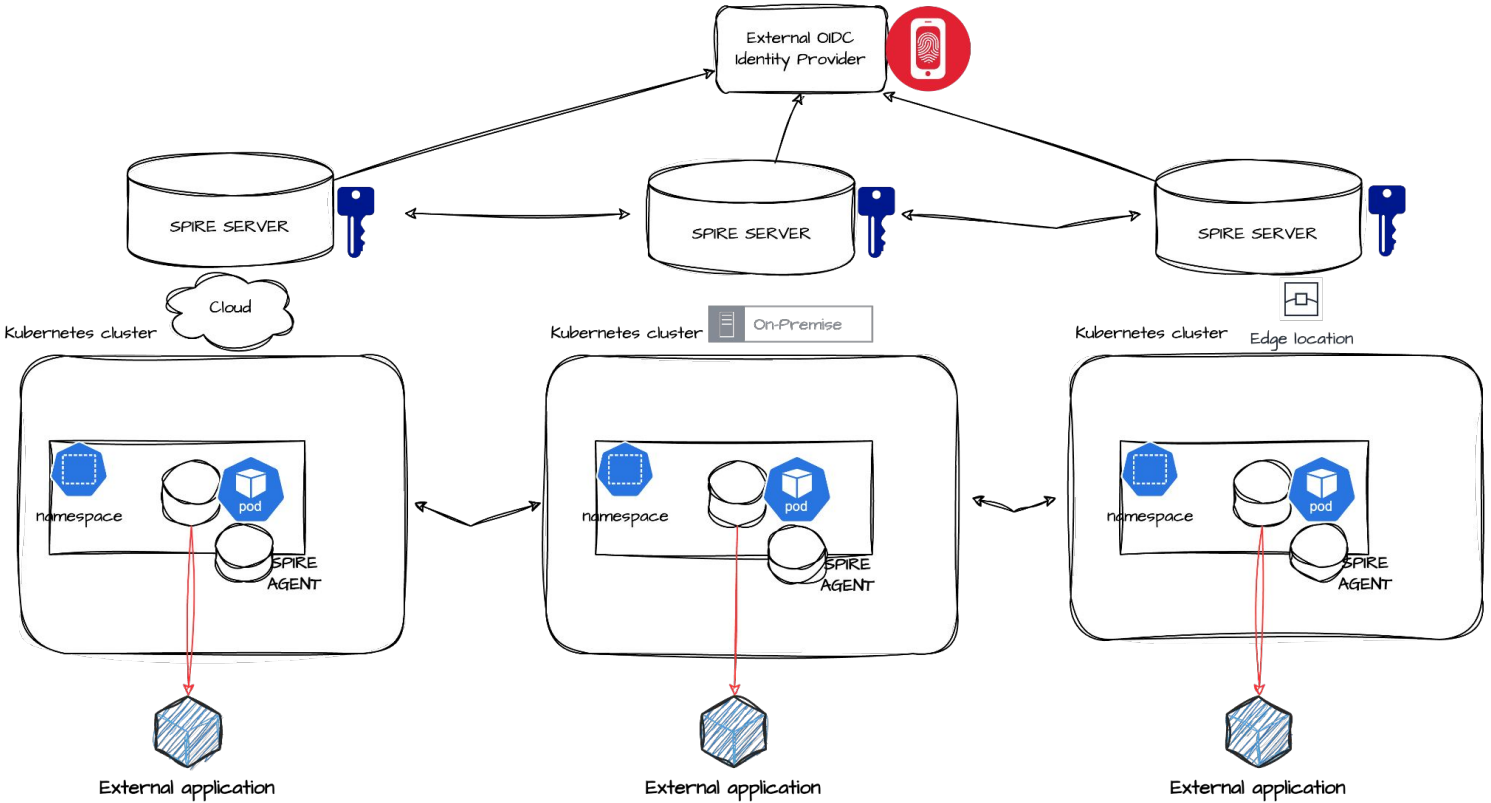
Demo

github.com/dronenb/kubecon-2024-wif-demo

Challenges Implementing Workload Identity at Scale

- Works great for a single cluster - but what about when running hundreds of clusters?
- Trying to get developers to use workload identity is hard enough - when running at scale, communicating OIDC issuers for each cluster to developers to configure their IAM mappings is not fun
- Moving workloads between clusters requires reconfiguring all IAM
- Would be nice if it was possible to federate with a single issuer

Workload Identity Federation with SPIRE



Questions?

Helpful Links

- https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_create_oidc.html
- https://docs.aws.amazon.com/rolesanywhere/latest/userguide/workload-identities.html
- https://docs.aws.amazon.com/STS/latest/APIReference/API_AssumeRoleWithWebIdentity.html
- https://learn.microsoft.com/en-us/entra/workload-id/workload-identity-federation
- https://cloud.google.com/iam/docs/workload-identity-federation-with-kubernetes
- https://docs.openshift.com/container-platform/4.17/authentication/managing_cloud_provider_credentials/about-cloud-credential-operator.html
- https://spiffe.io/docs/latest/spiffe-about/spiffe-concepts/

Feedback