Portable, Universal, Single Sign-On for Your Clusters

Miguel Martinez @migmartri
May 21st, Kubecon EU

bitnami

Hi, I am Miguel!

- Spaniard in San Francisco, obsessed with Mexican food
- Full-stack developer at Bitnami
- Core contributor of Kubeapps, Monocular and Helm

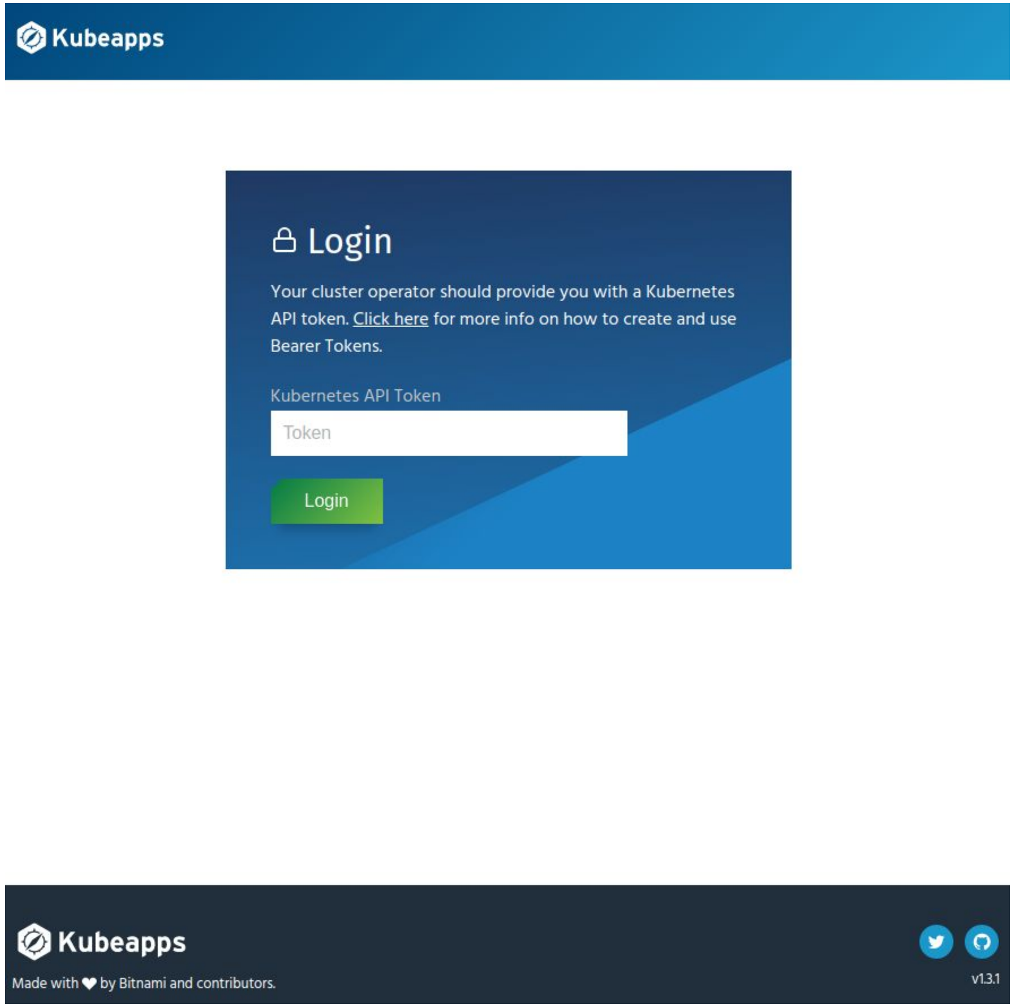
@migmartri

Agenda

- Problem statement
- OIDC intro and key takeaways
- Implementation challenges
- Workarounds and demos

Our problem

Support Single Sign-on in Kubeapps



Your cluster operator should provide you with a Kubernetes API token. Click here for more info on how to create and use Bearer Tokens.

Made with ♥ by Bitnami and contributors.

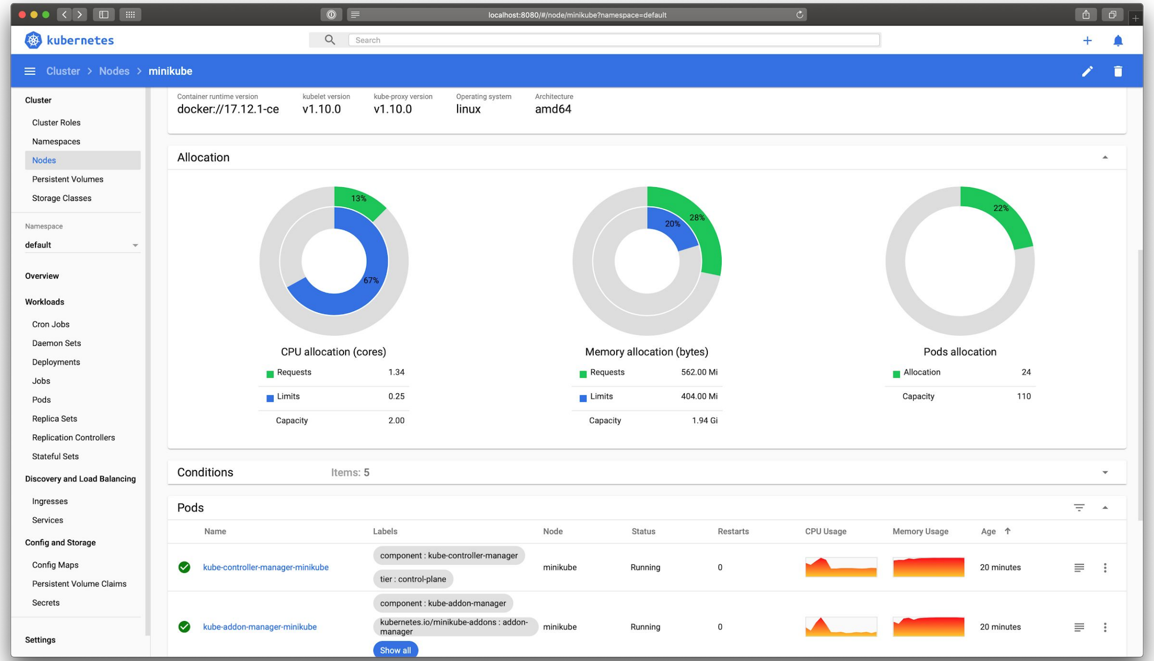
- Only supported k8s service accounts
- Adoption barrier
- Security best practices blocker

Single sign-on, most requested feature

bitnami https://github.com/kubeapps/kubeapps

Our problem

General Statement



Container runtime version	kubelet version	kube-proxy version	Operating system	Architecture
docker://17.12.1-ce	v1.10.0	v1.10.0	linux	amd64

Allocation

CPU allocation (cores)

🟩 Requests	1.34
🟦 Limits	0.25
Capacity	2.00

Memory allocation (bytes)

🟩 Requests	562.00 Mi
🟦 Limits	404.00 Mi
Capacity	1.94 Gi

Pods allocation

🟩 Allocation	24
Capacity	110

Conditions Items: 5

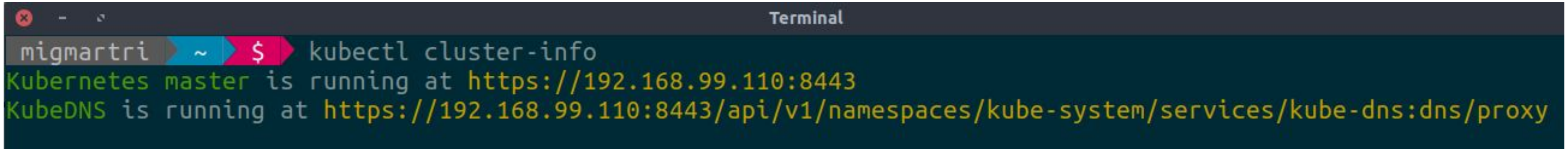
Pods

	Name	Labels	Node	Status	Restarts	CPU Usage	Memory Usage	Age ↑
✓	kube-controller-manager-minikube	component : kube-controller-manager tier : control-plane	minikube	Running	0			20 minutes
✓	kube-addon-manager-minikube	component : kube-addon-manager kubernetes.io/minikube-addons : addon-manager	minikube	Running	0			20 minutes

Application (YourApp) that

1. AuthN users via Single sign-on
2. **Talks to the k8s API server**

i.e kubectl, Kubernetes Dashboard, Kubeapps



```
migmartri  ~  $  kubectl cluster-info
Kubernetes master is running at https://192.168.99.110:8443
KubeDNS is running at https://192.168.99.110:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

bitnami

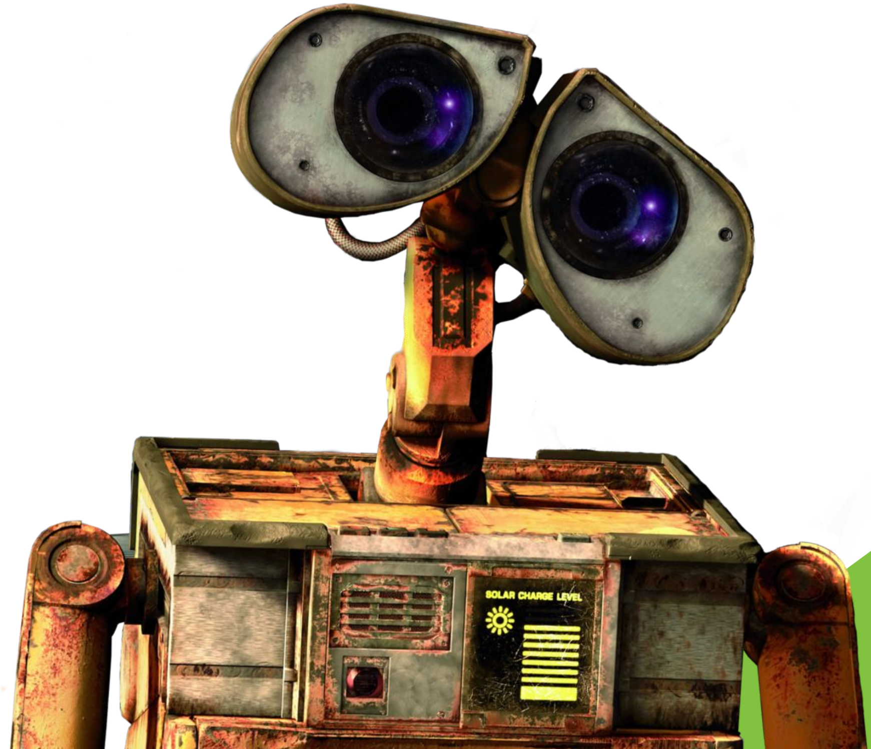
Our problem

Solution Scope and Caveats

Platform ~~dependent~~ vs Independent

AuthN vs ~~AuthZ~~

~~Robots~~ vs Humans



bitnami

AuthN in Kubernetes

User Authentication Overview

	Self-service	Rotation	Revocation	UX
X509 Client Certs	🟥	🟥	🟥	🟨
Token (SA or Static)	🟥	🟨	🟨	🟨
Basic Auth	🟥	🟨	🟨	🟨
Single sign-on - OpenID Connect	🟩	🟩	🟨	🟩

https://kubernetes.io/docs/reference/access-authn-authz/authentication/

SSO in Kubernetes

Why?

For Users

- Familiar AuthN mechanism
- No need to have additional set of credentials
- Self-service

For Cluster Operators

- No manual generation or transfer of credentials
- Built-in rotation and revocation methods
- AuthN delegation
- Support for groups and scopes

SSO in Kubernetes

Kubernetes API understands
OpenID Connect (OIDC)

OAuth2 != OIDC!

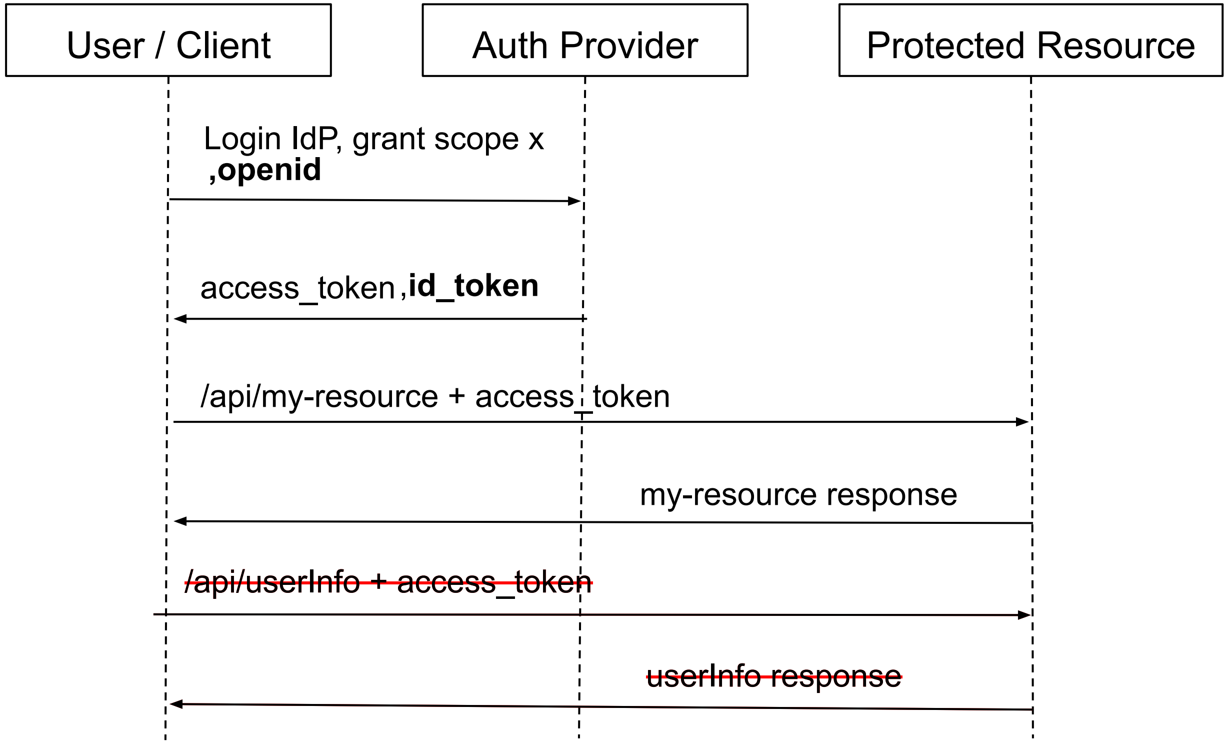
bitnami

SSO in Kubernetes

OpenID Connect (OIDC)

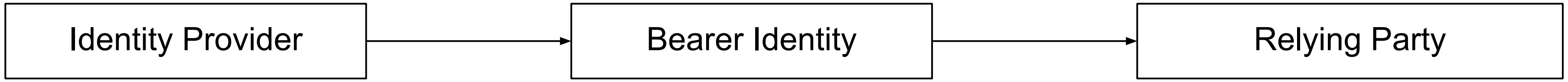
Identity layer **on top of** the OAuth 2.0 AuthZ protocol

Authentication Info **standardized** in a **cryptographically signed JWT token** called id_token

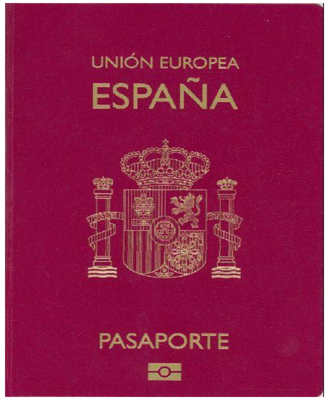

User / Client
Auth Provider
Protected Resource
Login IdP, grant scope x ,**openid**
access_token ,**id_token**
/api/my-resource + access_token
my-resource response
~~/api/userInfo + access_token~~
~~userInfo response~~

SSO in Kubernetes

OpenID Connect - Trust Chain example. Trip to the US



- Verify Identity
- Craft Passport



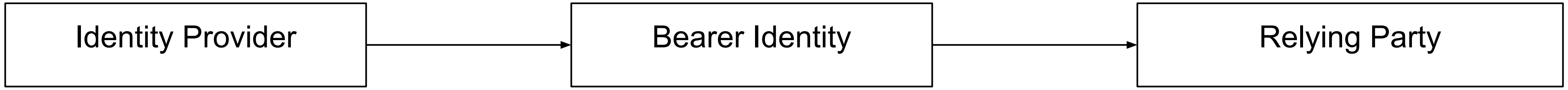
U.S. CUSTOMS AND BORDER PROTECTION
U.S. DEPARTMENT OF HOMELAND SECURITY

- It is from a trusted source
- It has not been tampered with (ePassport)
- It is not expired

The relying party does not contact the identity provider

SSO in Kubernetes

OpenID Connect - Trust Chain in Kubernetes



dex

KEYCLOAK

Microsoft Active Directory

ID Token (JWT)

- Header
- Payload
  - Identity (sub/email)
  - Who provisioned this token (**iss**)
  - Intended client audience (**aud**)
  - Expiration time (**exp**)
  - Claims (name, email, ...)
- **Signature**

K8s API Server

3rd party app

- It has not been tampered with (**signature**)
- It is from a trusted source (**iss**)
- I am the receiver (**aud**)
- It is not expired (**exp**)
- **DOES NOT contact the IdP (except to retrieve the Public Key)**

SSO in Kubernetes

Integration

You need to configure the K8s API server
to trust an OIDC Identity Provider

```
# API server flags

--oidc-issuer-url https://my-oidc-idP.com # .../.well-known/openid-configuration
--oidc-client-id my-client-id
--oidc-username-claim email
--oidc-groups-claim groups

# oidc-issuer.match(id_token.iss) && oidc-client-id.match(id_token.aud)
```

```
$ curl https://api-server -H "Authorization: Bearer ${id_token}"

$ kubectl --token ${id_token}
```

https://kubernetes.io/docs/reference/access-authn-authz/authentication/

Problem Statement

Summary

Application (YourApp) that:

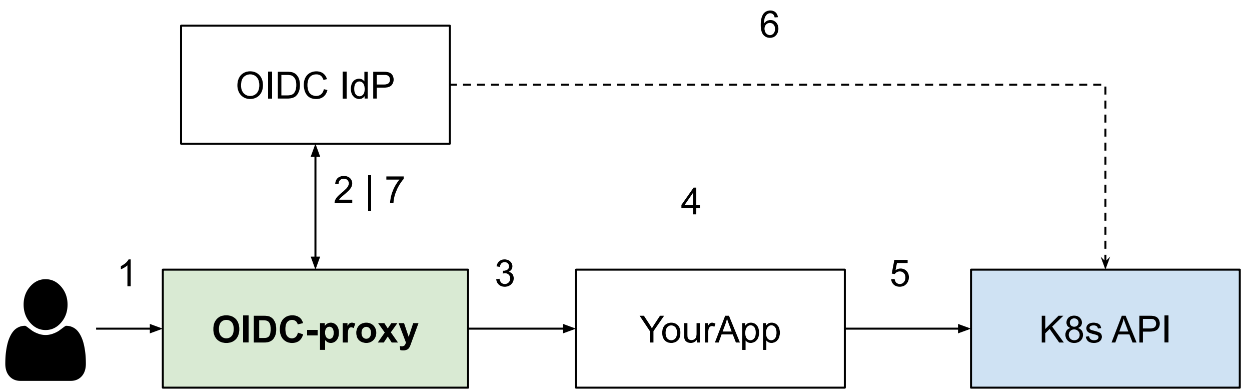
1. AuthN users via OIDC single sign-on
2. Talks directly to the k8s API server

i.e kubectl, Kubernetes Dashboard, Kubeapps

Problem Statement

Solution

Proxy configured with the **same OIDC IdP than the k8s API server ***

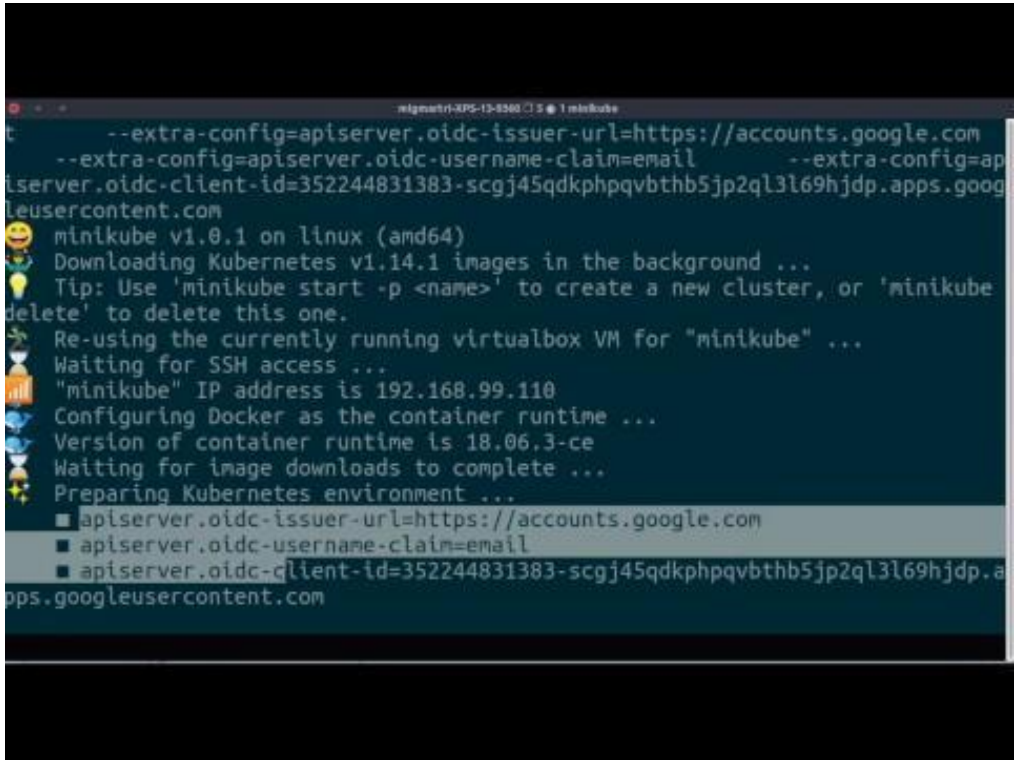


- Enforce AuthN with an external IdP
- Takes care of the OAuth2 dance, token exchange and refresh
- Inject ID Headers and forward them upstream

Problem Statement

Demo, Exclamation Mark

SSO-enabled Kubernetes Dashboard
+ Google's IdP on Minikube



```
t        --extra-config=apiserver.oidc-issuer-url=https://accounts.google.com
    --extra-config=apiserver.oidc-username-claim=email        --extra-config=ap
iserver.oidc-client-id=352244831383-scgj45qdkphpqvbthb5jp2ql3l69hjdp.apps.goog
leusercontent.com
    minikube v1.0.1 on linux (amd64)
    Downloading Kubernetes v1.14.1 images in the background ...
    Tip: Use 'minikube start -p <name>' to create a new cluster, or 'minikube
delete' to delete this one.
    Re-using the currently running virtualbox VM for "minikube" ...
    Waiting for SSH access ...
    "minikube" IP address is 192.168.99.110
    Configuring Docker as the container runtime ...
    Version of container runtime is 18.06.3-ce
    Waiting for image downloads to complete ...
    Preparing Kubernetes environment ...
    ■ apiserver.oidc-issuer-url=https://accounts.google.com
    ■ apiserver.oidc-username-claim=email
    ■ apiserver.oidc-client-id=352244831383-scgj45qdkphpqvbthb5jp2ql3l69hjdp.a
pps.googleusercontent.com
```

Problem Statement

We Are Not Done Yet

The solution does not work in all platforms

- K8s provider API server lockdown
- Ops do not want OIDC enabled in k8s API
- IdP or authN requirements mismatch (LDAP)
- IdP groups/user claim support

Problem Statement

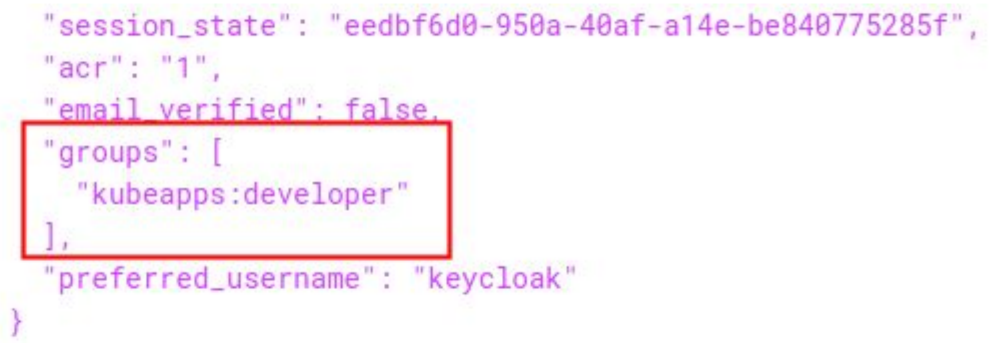
Challenge 1: K8s API Server OIDC Customisation

Kubernetes Distro	API Server OIDC Customization
GKE (Google)	No
EKS (AWS)	No
AKS (Azure)	Active Directory
OKE (Oracle)	Oracle Identity Cloud Service
Minikube	Any
Kops	Any
kubeadm	Any

Problem Statement

Challenge 2: Identity Providers and Group Claims

```
subjects:
- kind: Group
  name: "kubeapps:developer"
  apiGroup: rbac.authorization.k8s.io
```



```
  "session_state": "eedbf6d0-950a-40af-a14e-be840775285f",
  "acr": "1",
  "email_verified": false,
  "groups": [
    "kubeapps:developer"
  ],
  "preferred_username": "keycloak"
}
```

OIDC Identity Provider	Group Claims Support
Okta	🟩
Dex	Depends on Upstream
Keycloak	🟩
Active Directory	🟩
Google Accounts	🟥
...	...

```
--oidc-groups-claim "groups" # API flag
```

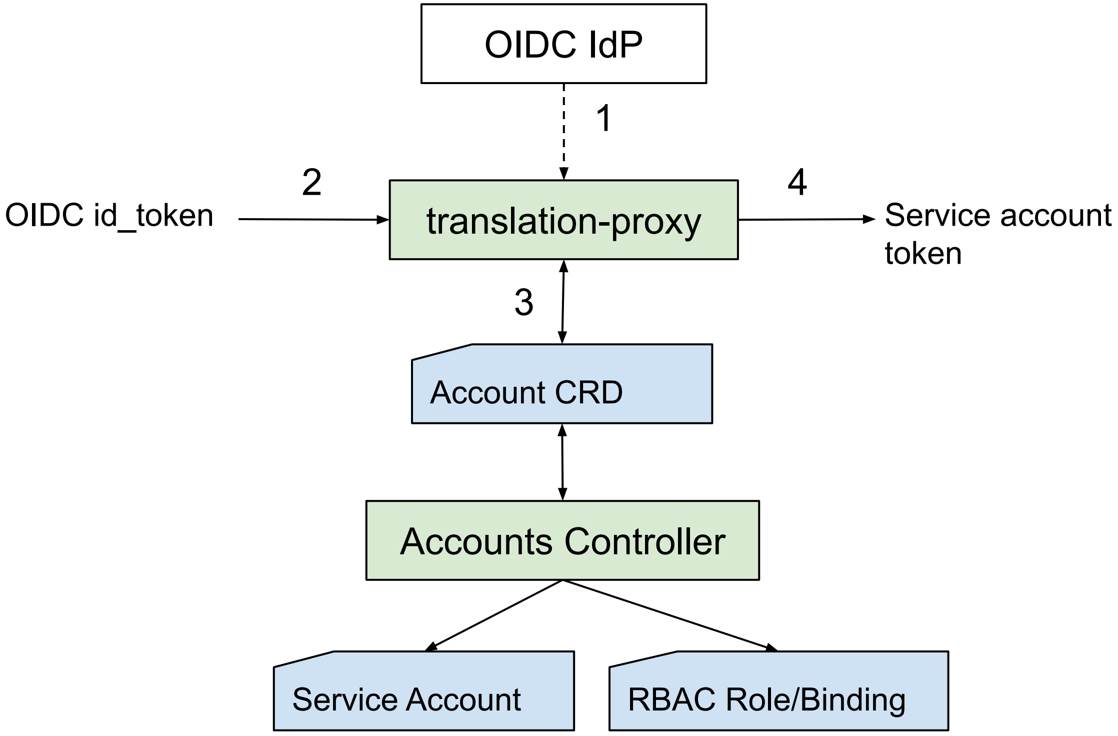
> “Easy things should be easy, and hard things should be possible.”

Larry Wall

Workaround 1

Translation to Service Accounts

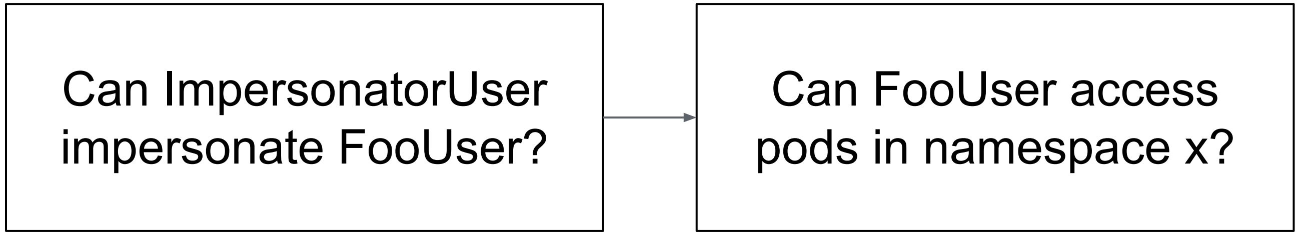
Translate OIDC **id_tokens into service accounts** via a translation proxy and a custom controller



Workaround 2

Kubernetes Impersonation

```
$ kubectl get pods --as FooUser --as-group kubeapps-user
```



Can ImpersonatorUser impersonate FooUser?

Can FooUser access pods in namespace x?

```
- apiGroups: [""]
  resources: ["groups"]
  verbs: ["impersonate"]
  resourceNames:
["developers","kubeapps-user"]
```

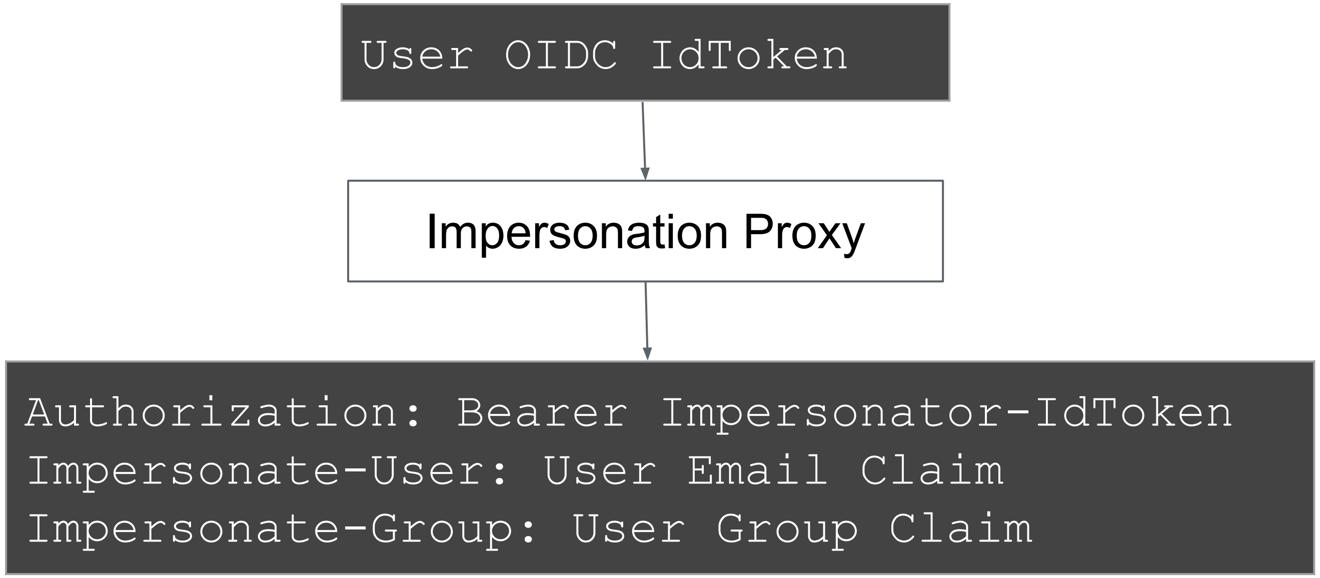
```
$ curl https://api-server/api/v1/pods \
  -H "Authorization: Bearer ${impersonator token}" \
  -H "Impersonate-User: FooUser"
  -H "Impersonate-Group: kubeapps-user"
```

https://kubernetes.io/docs/reference/access-authn-authz/authentication/#user-impersonation

Workaround 2

Kubernetes Impersonation

Proxy in charge of impersonating users and groups based on OIDC id_token claims



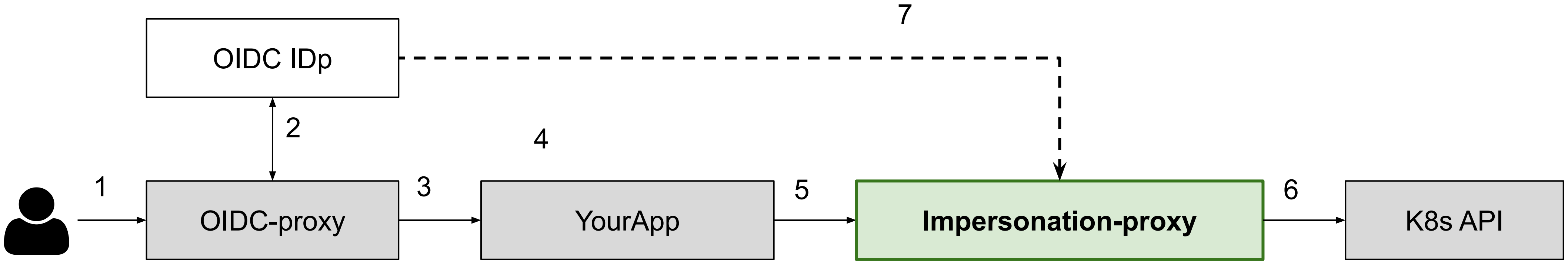
```
User OIDC IdToken
```

Impersonation Proxy

```
Authorization: Bearer Impersonator-IdToken
Impersonate-User: User Email Claim
Impersonate-Group: User Group Claim
```

Workaround 2

Kubernetes Impersonation

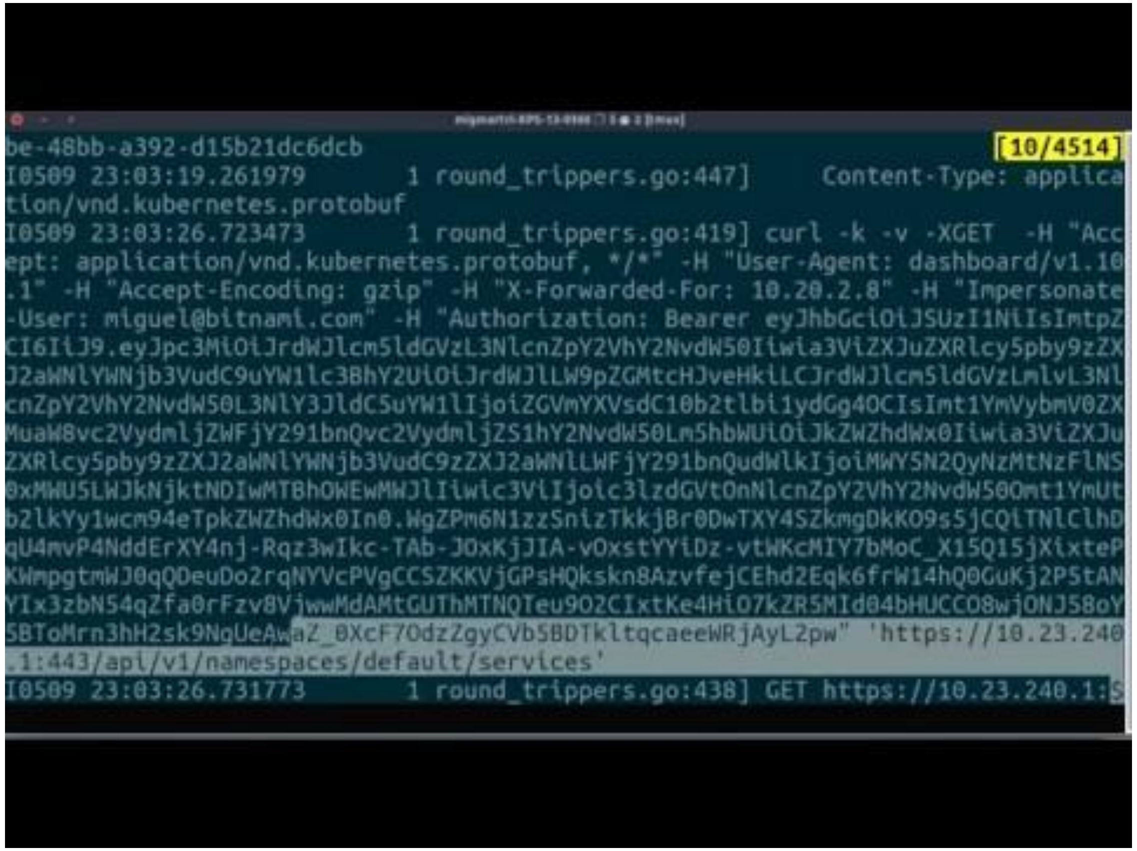


- Extracts OIDC verification logic from API server
- Prevent stale credentials
- Fewer moving pieces

Workarounds

Kubernetes Impersonation

SSO-enabled Kubernetes Dashboard + Google's IdP on **GKE*** using impersonation



```
be-48bb-a392-d15b21dc6dcb                                          [10/4514]
I0509 23:03:19.261979       1 round_trippers.go:447]     Content-Type: applica
tion/vnd.kubernetes.protobuf
I0509 23:03:26.723473       1 round_trippers.go:419] curl -k -v -XGET  -H "Acc
ept: application/vnd.kubernetes.protobuf, */*" -H "User-Agent: dashboard/v1.10
.1" -H "Accept-Encoding: gzip" -H "X-Forwarded-For: 10.20.2.8" -H "Impersonate
-User: miguel@bitnami.com" -H "Authorization: Bearer eyJhbGciOiJSUzI1NiIsImtpZ
CI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZX
J2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLW9pZGMtcHJveHkiLCJrdWJlcm5ldGVzLmlvL3Nl
cnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoiZGVmYXVsdC10b2tlbi1ydGg4OCIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50Lm5hbWUiOiJkZWZhdWx0Iiwia3ViZXJu
ZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiMWY5N2QyNzMtNzFlNS
0xMWU5LWJkNjktNDIwMTBhOWEwMWJlIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50Omt1YmUt
b2lkYy1wcm94eTpkZWZhdWx0In0.WgZPm6N1zzSnizTkkjBr0DwTXY4SZkmgDkKO9s5jCQiTNlClhD
qU4mvP4NddErXY4nj-Rqz3wIkc-TAb-JOxKjJIA-vOxstYYiDz-vtWKcMIY7bMoC_X15Q15jXixteP
KWmpgtmWJ0qQDeuDo2rqNYVcPVgCCSZKKVjGPsHQkskn8AzvfejCEhd2Eqk6frW14hQ0GuKj2P5tAN
YIx3zbN54qZfa0rFzv8VjwwMdAMtGUThMTNQTeu9O2CIxtKe4HiO7kZR5MId04bHUCCO8wjONJ58oY
5BToMrn3hH2sk9NgUeAwaZ_0XcF7OdzZgyCVb5BDTkltqcaeeWRjAyL2pw" 'https://10.23.240
.1:443/api/v1/namespaces/default/services'
I0509 23:03:26.731773       1 round_trippers.go:438] GET https://10.23.240.1:S
```

*** API server not configurable**

https://github.com/jetstack/kube-oidc-proxy

SSO in Kubernetes can be **vendor locked** but we can **workaround** it and offer a Universal, **Cross-Platform SSO experience**

Resources

- Kubeapps SSO in-depth document - https://bit.ly/30bi1zF
- SSO for Kubernetes talk - @JoelASpeed - https://bit.ly/2Hh6kQN
- kube-oidc-proxy - @jetstack - https://bit.ly/2Vip6uw
- Demo files repository - @prydonius - https://bit.ly/2HfV9GI
- This slide deck - https://bit.ly/2WD8YoT

@migmartri

Thank You