Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Technology
Arts Sciences
TH Köln

Granular access control to kube-apiserver using OpenID Connect

Presenter → Dikshita Kalita
Supervisor → Prof. Dr. Martin Leischner
Mentor → Richard Clauß

Date → 20.01.2023

Agenda

1. Motivation - Cryptojacking and attacks against the kube-apiserver
2. Research Question - How can security be improved?
3. Reason why OpenID Connect can improve security of the kube-apiserver
4. OpenID Connect with Keycloak in Practice
5. Summary

1.1 Motivation

Problem / Attack → **CRYPTOJACKING**

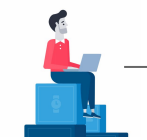
Attacker gains access to our kubernetes cluster and hijacks the compute resources in order to mine cryptocurrency

Primary, but not exclusive point of attack:

kube-apiserver (core of the kubernetes control-plane)

- Attacker compromises it
- Then starts malicious pods, which mine cryptocurrencies

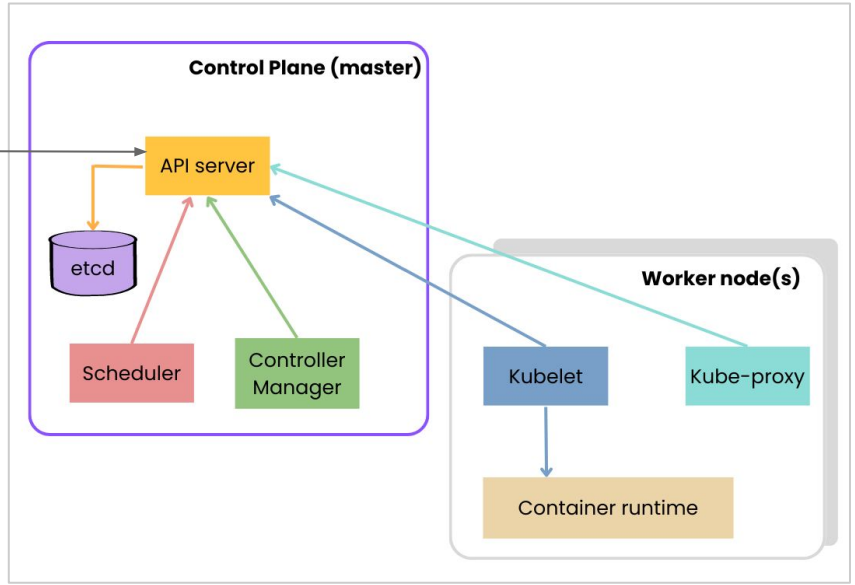
1.2 Compromised Client-Certificate and Client-Key



CLI
[kubectl]

Port: 6443

Client-Certificate
+
Client-Key



One way to compromise kube-apiserver :

→ Get client key and certificate used with kubectl

Fig: Kubernetes architecture

2 Research Question

How can the security of the kube-apiserver be improved against a compromised pair of client-key and -certificate?

3.1 Client-certificates are valid infinitely

Client-Certificate
+
Client-Key

→ Kubeconfig YAML


```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUMvakNDQWVhZ0F3SUJBZ0lCQURBTkJna3Foa2lHOX
cwQkFRc0ZBREFWTVJNd0VRWURWUVFERXdwcmRXSmwkY201bGRHVnpNQjRYRFRJek1ERXhPREV3TUR
Rd05Gb1hEVEI6TURFeE5URXdNRFF3TkZvd0RVJUSUZJQ0FURS0tLS0tCg==
    server: https://10.20.116.209:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURJVENDQWdtZ0F3SUJBZ0lJWE1iamFjTzBXSGt3RFFZS
ktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEplWlhSbGN6QWVGdzB5TXpBeE1
UZ3hNREEwTURSYUZ3MHlOREF4TVRENZemcy95SlhELUVORCBDRVJUSUZJQ0FURS0tLS0tCg==
    client-key-data:
LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFcFFJQkFBS0NBUUVBOG9LajdxVEFKdFJ0QVJm
eHAyVnFaalhuMWZrcjREQVp0bnUzUTh6L3ZwY1NsUFZRCjZKaUMwc1cvb0x0TXZFR0VOY2tJRysvbE1xWn
o4KzFhc3p3VHRHeGZlZHlubXg2MmlKMWejlGa0YU5yUG5mS1BlVFk5VQo=
```


Fig : kubeconfig file with client certificate and key

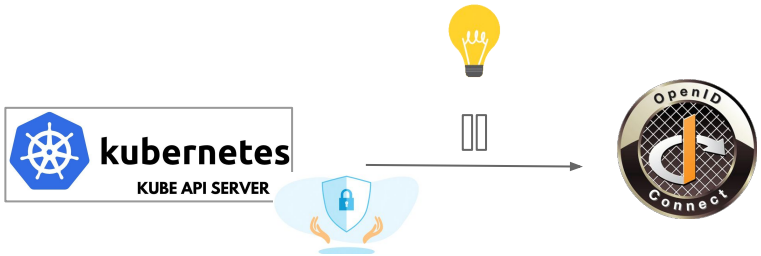
Ways to obtain key and certificate

- Fired ex-employees turned into malicious actors to hijack clusters
- Malware
- Compromised backups

Problem

- Keys and Certificates are valid for eternity
- Kubernetes does not implement any revocation mechanism

3.2 Solution OpenID Connect



OpenID Connect solves the mentioned problem

REASON

- Uses signed tokens for authentication and authorization
- Tokens are short-lived and expire after a configurable duration
- If necessary, we can disallow any requests for new tokens
- These tokens are derived from OpenID provider like Keycloak

4.1 Authorization Code Flow with kubelogin

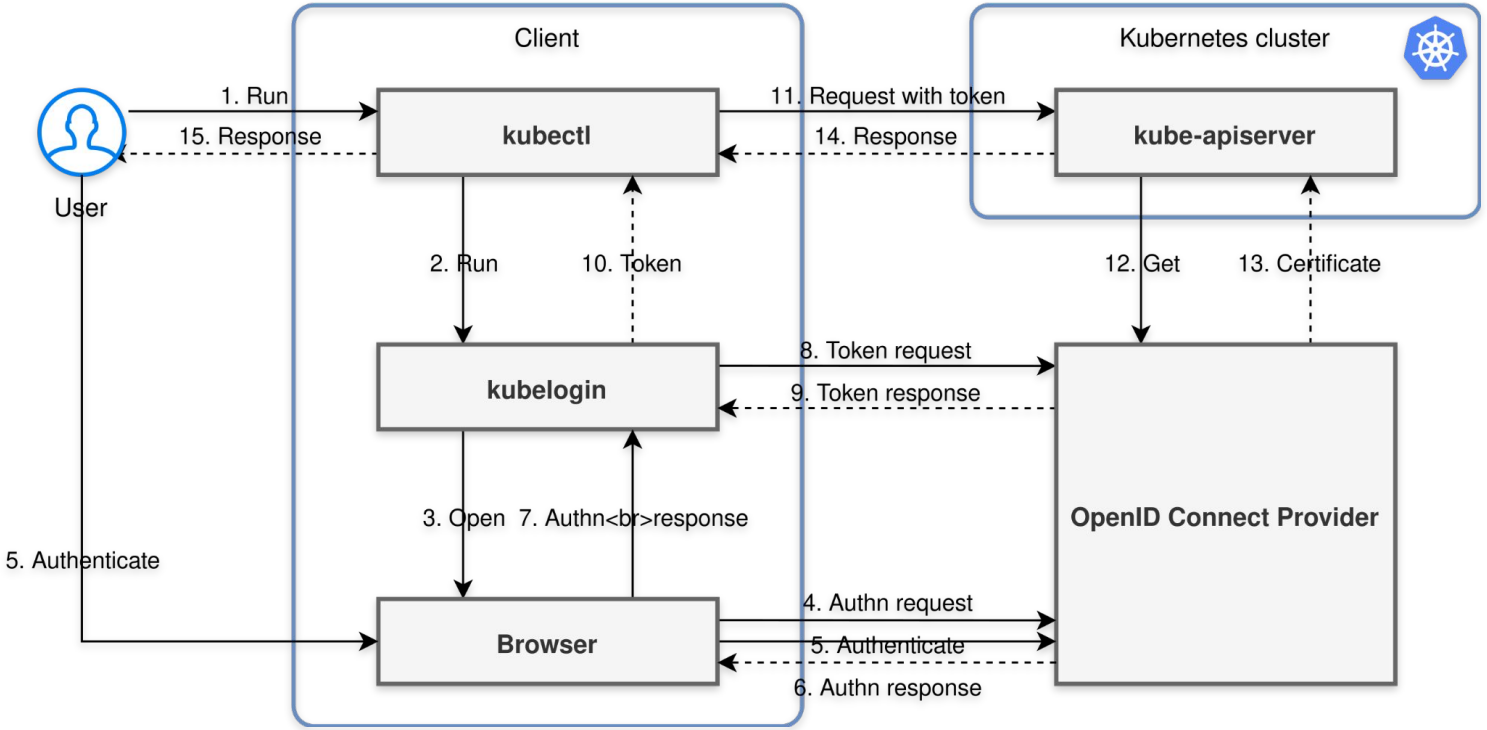


Fig: Authorization flow with kubelogin, Source: [7]

4.1.1 Structure of Json Web Tokens

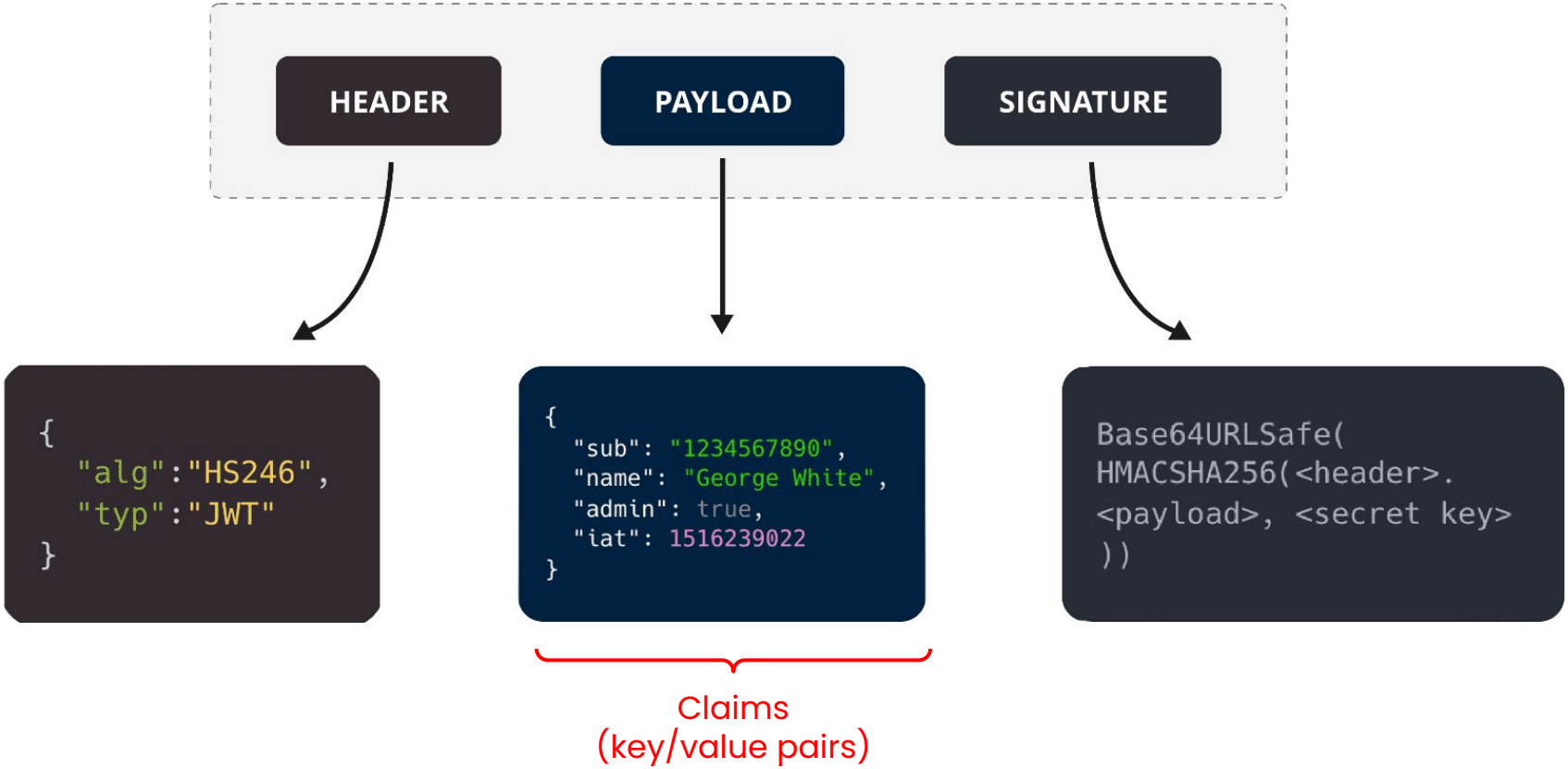


Fig: Parts of JSON Web Token

4.2 Mapping kubernetes groups to claims

Example for a JWT payload:

```
1  {
2    "preferred_username": "testuser",
3    [...]
4    "groups": [
5      "kubernetes_role",
6      "manage-account",
7      "manage-account-links",
8      "view-profile"
9    ],
10   [...]
11 }
```

Identity Provider URL

```
kube-apiserver \
--oidc-issuer-url=https://10.20.116.209.nip.io/realms/master \
--oidc-username-claim=preferred_username \
--oidc-groups-claim=groups
[ ... ]
```

Groups are then mapped to Roles with permissions in Kubernetes

4.3 Configuration of KeyCloak | Login to keycloak dashboard

Challenges:

- TLS required for kube-apiserver
- Domain for keycloak resolvable in- and outside of the cluster
- OAuth, OIDC and Keycloak are complex

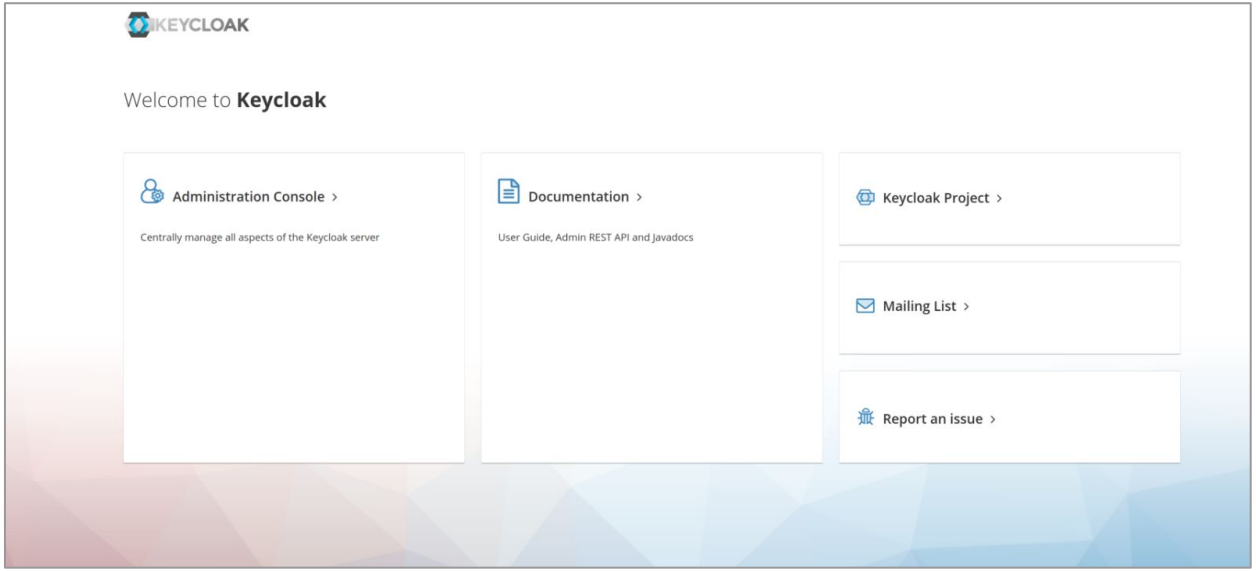
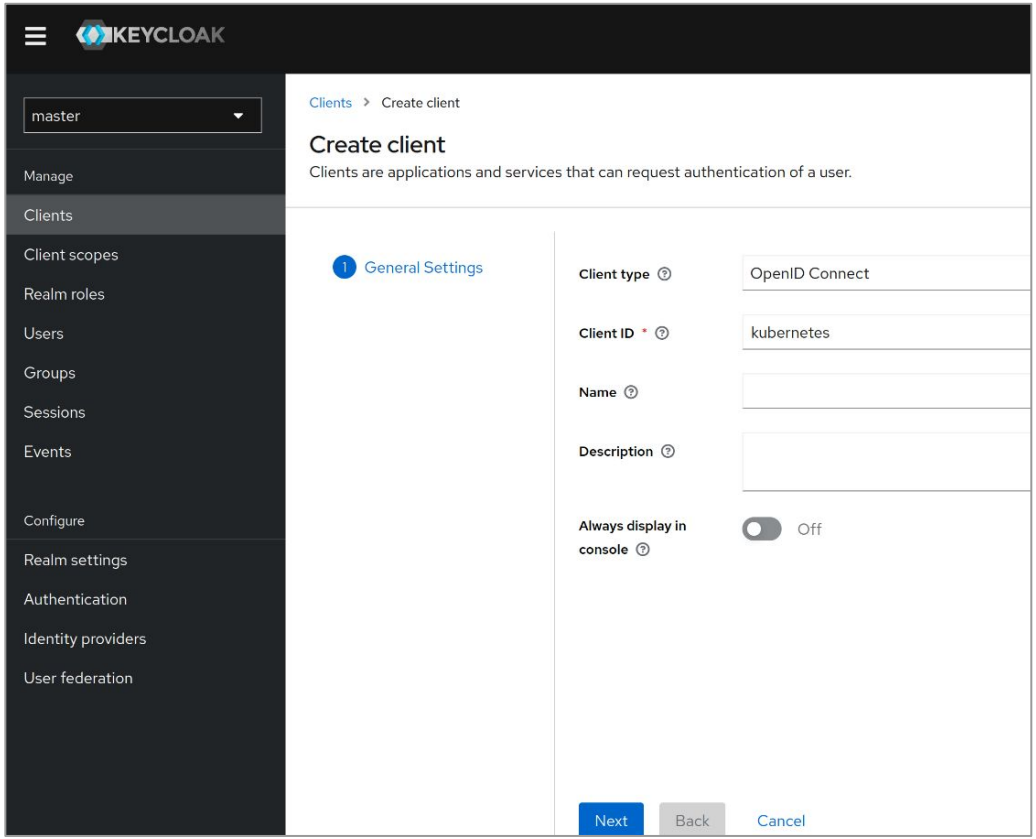


Fig: Accessing Keycloak dashboard

4.4 Configuration of KeyCloak | Clients



Client = Kubectl

Fig: Creating Client “kubernetes”

4.5 Configuration of KeyCloak | Clients

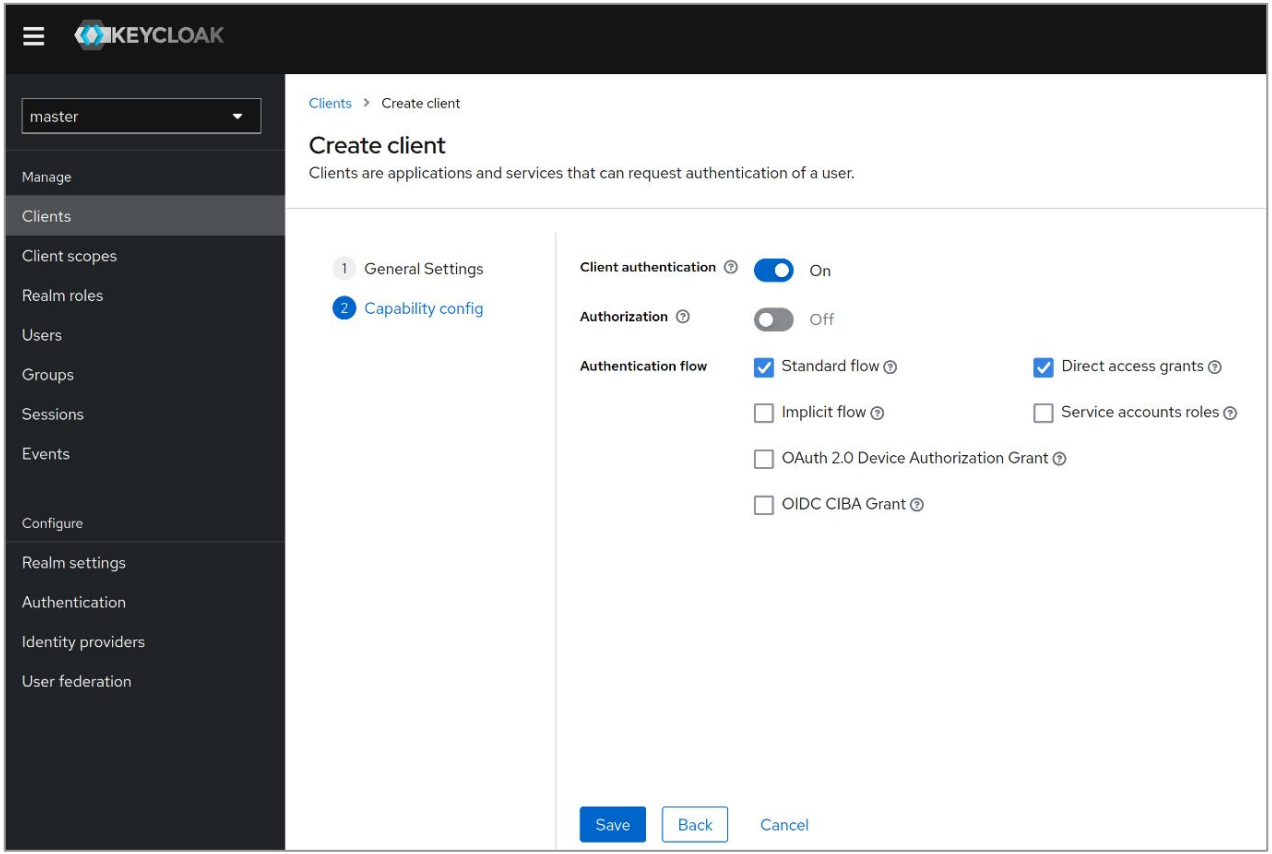


Fig: Choosing authentication flow for created client “kubernetes”

4.6 Configuration of KeyCloak | User

KEYCLOAK

admin

master

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

Events

Configure

Realm settings

Authentication

Identity providers

User federation

Users > Create user

Create user

Enabled

Action

Username *

testuser

Email

testuser@testuser.de

Email verified

On

First name

firstname

Last name

lastname

Required user actions

Select action

Groups

Join Groups

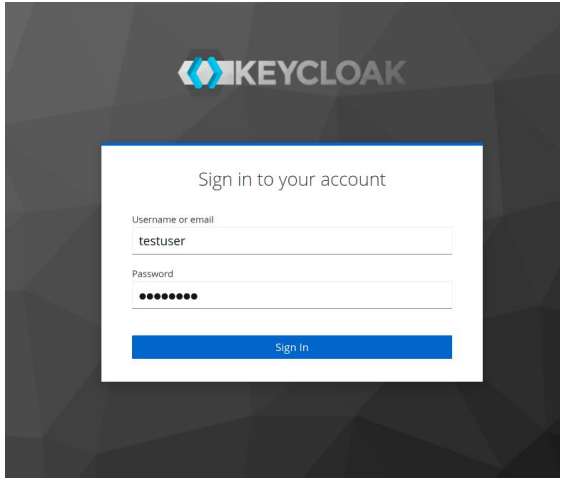
Create

Cancel

Fig: Creating a user

4.7 Final received results

```
[master@master ~]$ k get pods
```




```
[master@master ~]$ k get pods
NAME                                READY   STATUS    RESTARTS        AGE
my-release-nginx-ingress-m6hm2      1/1     Running   2 (146m ago)    7h43m
my-release-nginx-ingress-vwc64      1/1     Running   2 (146m ago)    7h43m
```

5 Summary

- Benefits of OIDC : 

  - ✓ Short-lived tokens which can be easily revoked
  - ✓ Fine granular authentication and authorization management in keycloak and kubernetes

- Results: 

  - ✓ We found a way to recover from compromised client-keys and -certificates.
    - → And this way mitigated the impact of cryptojacking attacks.
  - ✓ Further we implicitly unlocked features in keycloak like Two-Factor Authentication, password recovery and more

Thank you

Literatures

[1] “Authenticating,” *Kubernetes*. [Online]. Available: https://kubernetes.io/docs/reference/access-authn-authz/authentication/. [Accessed: Jan. 18, 2023]

[2] “The Kubernetes API Server: Exploring its security impact and how to lock it down.” [Online]. Available: https://cybersecurity.att.com/blogs/security-essentials/the-kubernetes-api-server-exploring-its-security-impact-and-how-to-lock-it-down. [Accessed: Jan. 18, 2023]

[3] “Kubernetes – Don’t Use Certificates for Authentication.” [Online]. Available: https://www.tremolosecurity.com/post/kubernetes-dont-use-certificates-for-authentication. [Accessed: Jan. 18, 2023]

[4] “What is OAuth 2.0 and what does it do for you?,” *Auth0*. [Online]. Available: https://auth0.com/intro-to-iam/what-is-oauth-2. [Accessed: Jan. 18, 2023]

[5] auth0.com, “JWT.IO.” [Online]. Available: http://jwt.io/. [Accessed: Jan. 18, 2023]

[6] “How to Secure Your Kubernetes Cluster with OpenID Connect and RBAC,” *Okta Developer*. [Online]. Available: https://developer.okta.com/blog/2021/11/08/k8s-api-server-oidc. [Accessed: Jan. 19, 2023]

[7] “Kubelogin Github.” [Online]. Available: https://github.com/int128/kubelogin. [Accessed: Jan. 19, 2023]

6 Appendix: Mapping kubernetes groups to claims

RBAC.yaml :