Protectors of the Realm: Breaking and Fixing Keycloak Configurations

Max Maaß, Tim Walter

2025-08-28 | KeyConf25 Amsterdam

iteratec

Once Upon a Time...

Who had to audit a Keycloak before?

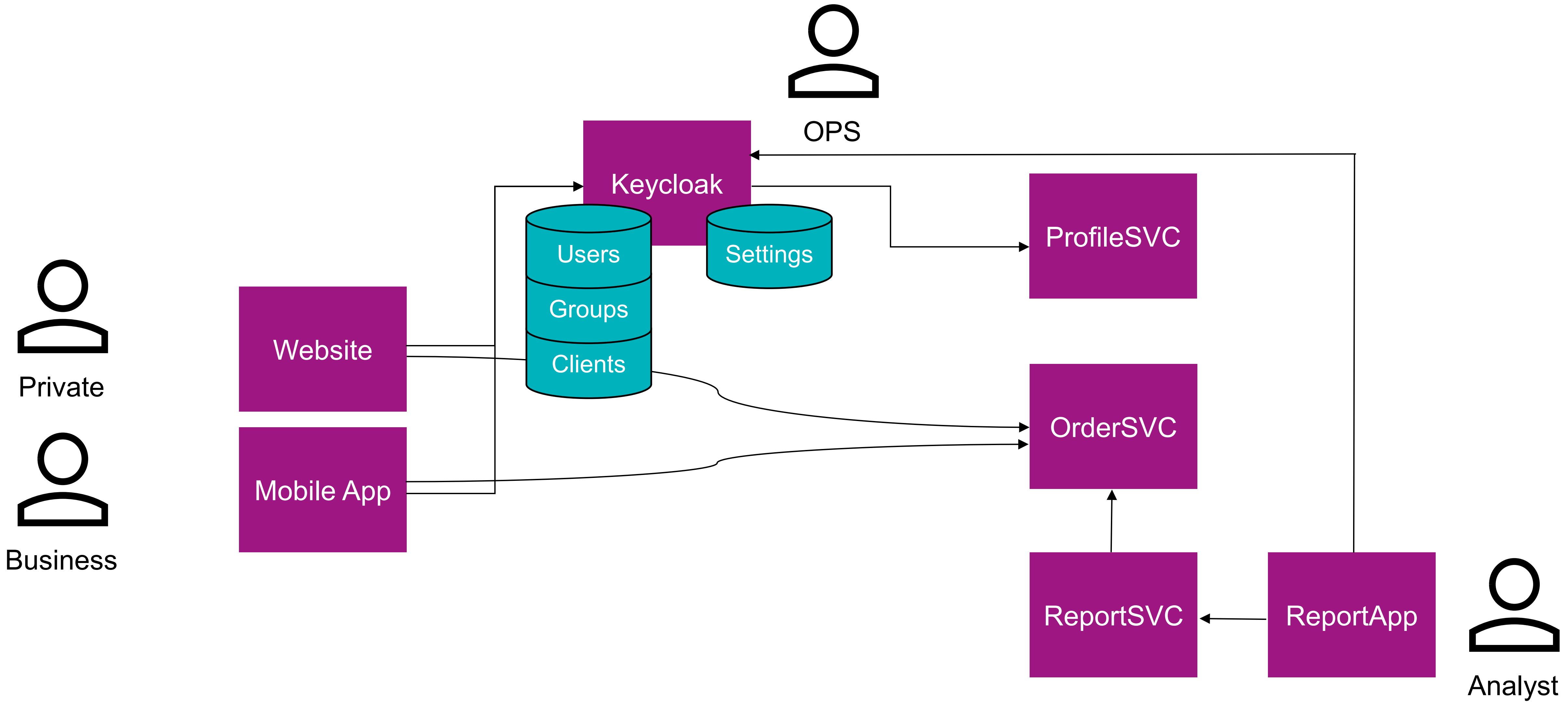
Who felt like they knew what they were doing?

Step 1: The Lay of the Land

- What systems are involved?
- What Keycloak features are they using?
- What groups of users are there? Internals, externals, …?
- What is the business context? Which business processes are involved?
- Who is maintaining and configuring the server?
- Are there any extensions (SPIs) installed? What are they doing?
- ...

Step 1: The Lay of the Land

iteratec



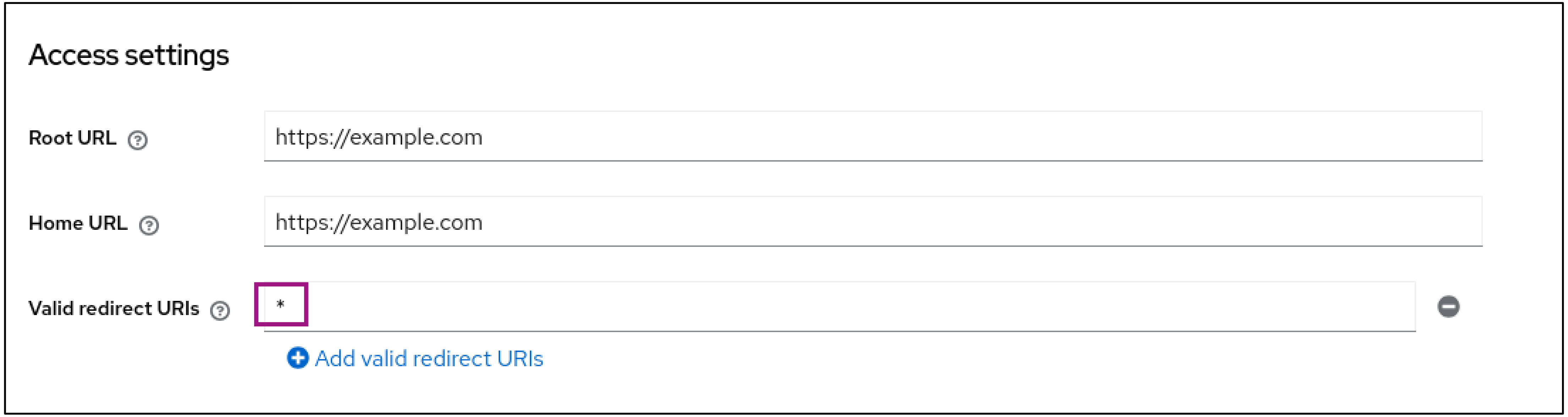
Step 2: Slay the Dragon Before the Wolves

- Check the Keycloak version

- Configuration issues
  - Many features and configuration possibilities
  - Insecure defaults
  - Configuration is mostly changed by hand

Example #1: Valid Redirect URIs

iteratec

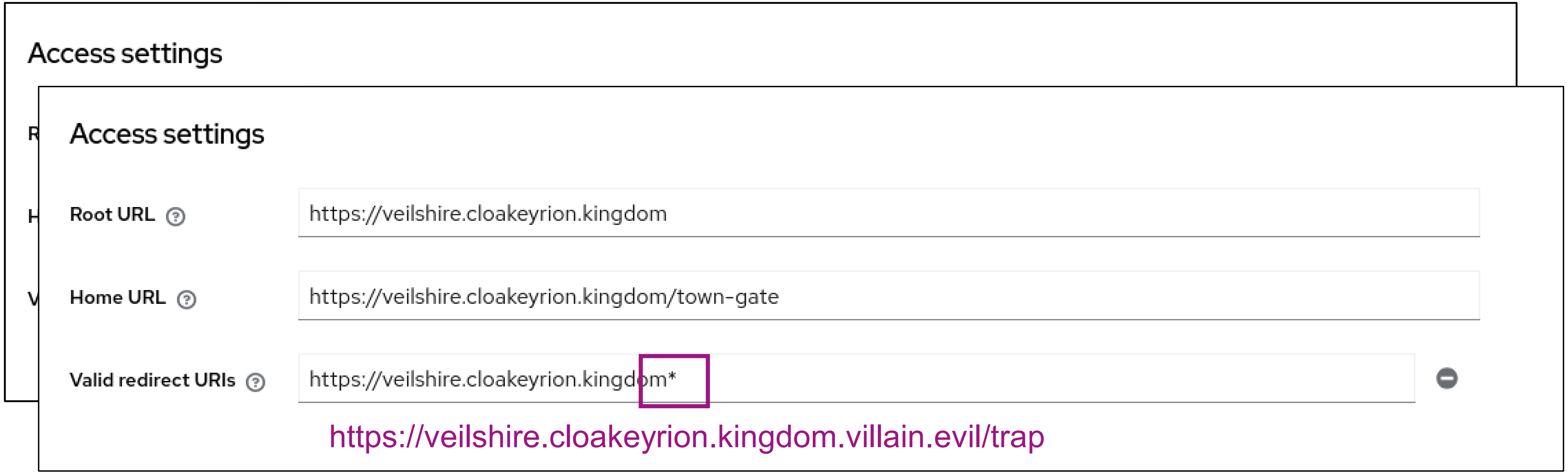


Access settings

Root URL	https://example.com
Home URL	https://example.com
Valid redirect URIs	*

Example #1: Valid Redirect URIs

iteratec


Access settings

Root URL	https://veilshire.cloakeyrion.kingdom
Home URL	https://veilshire.cloakeyrion.kingdom/town-gate
Valid redirect URIs	https://veilshire.cloakeyrion.kingdom*

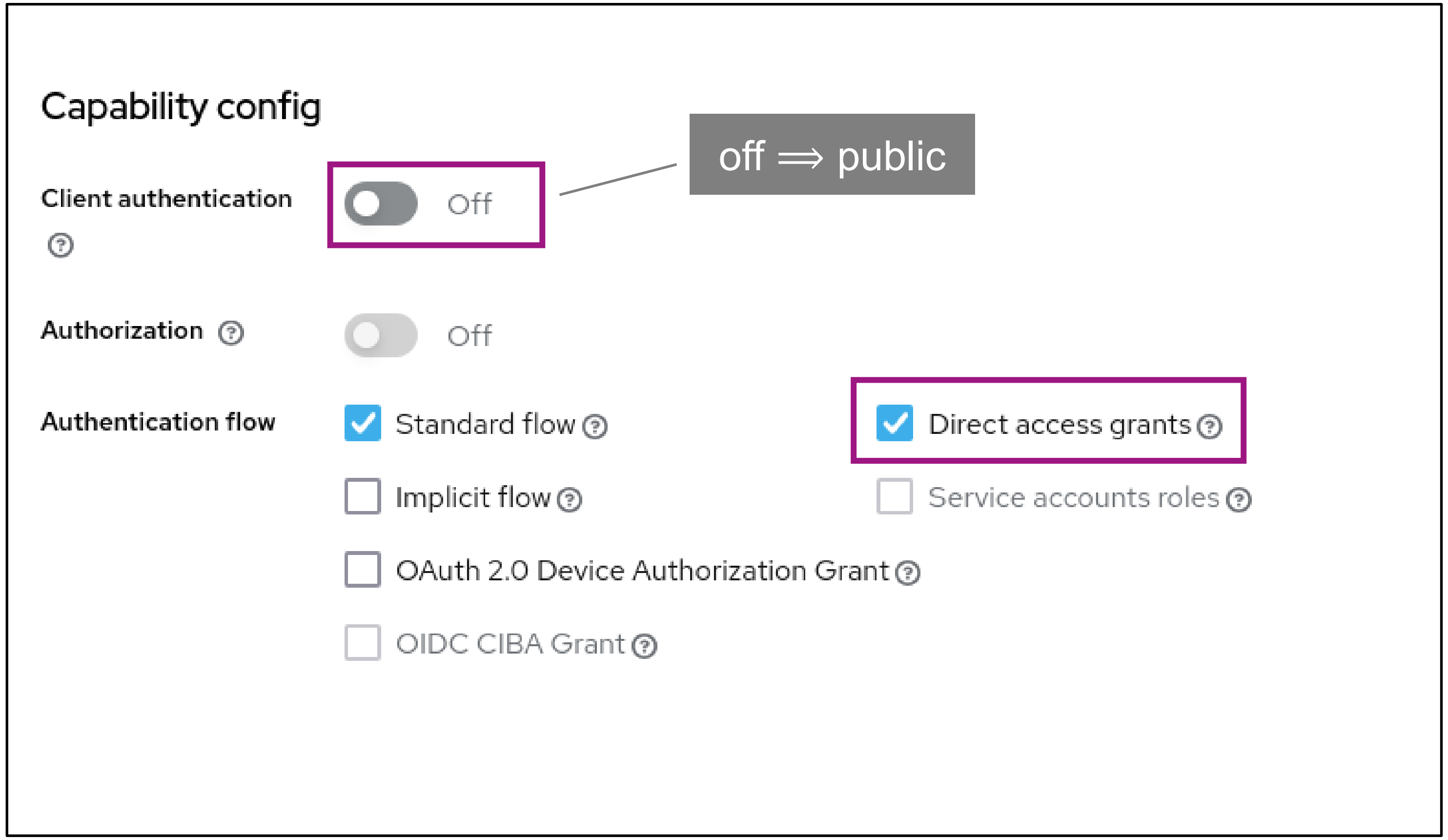
https://veilshire.cloakeyrion.kingdom.villain.evil/trap

Example #2: Direct Access Grants

iteratec

- *Legacy OAuth 2.0 Password Grant Type*
- Directly send username and password to Keycloak to get an access token in exchange
- Bypass security mechanisms like redirect URI check
- Easy brute-forcing or phishing

Should be turned off at least for public clients!



Capability config

Client authentication: Off

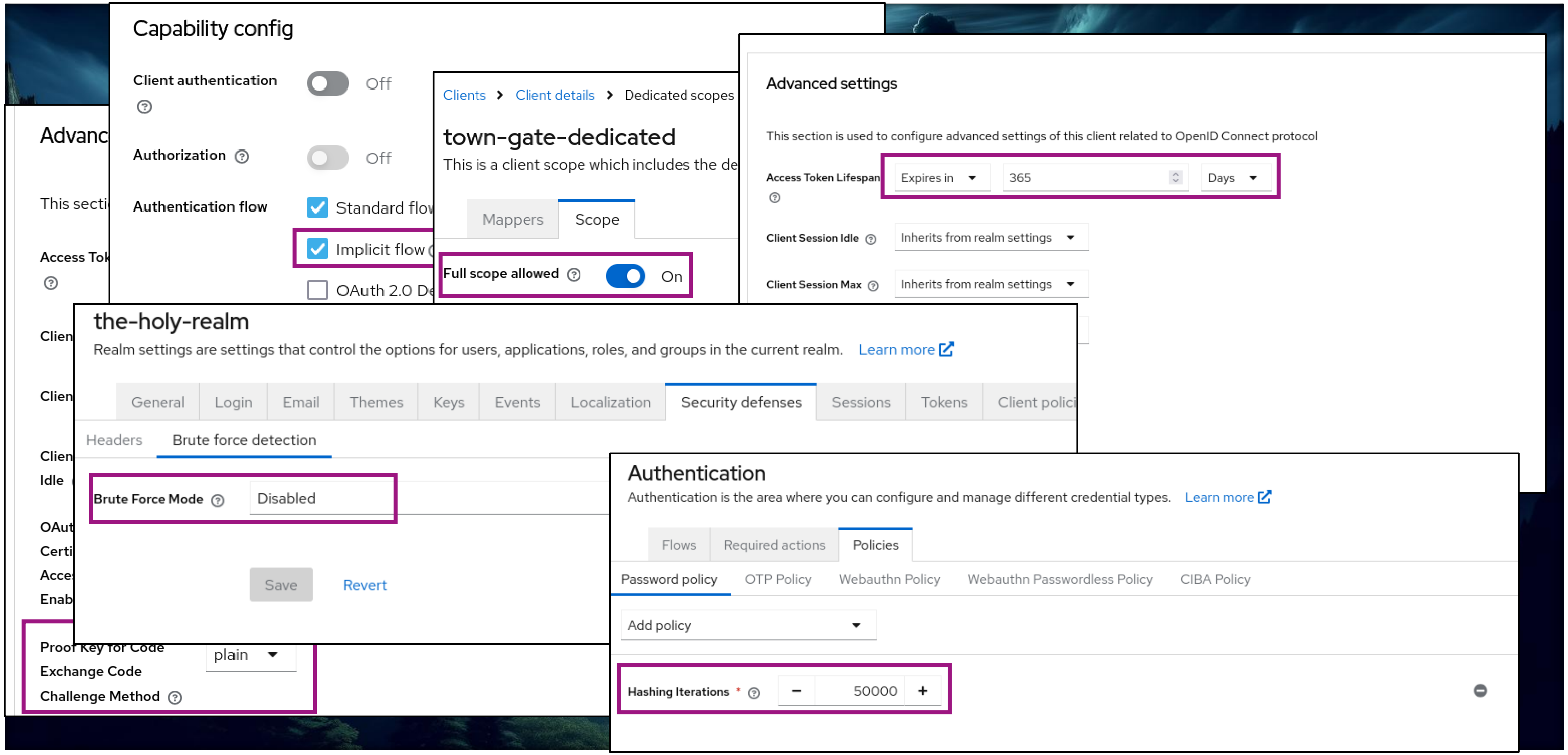
off ⟹ public

Authorization: Off

Authentication flow:
- [✓] Standard flow
- [✓] Direct access grants
- [ ] Implicit flow
- [ ] Service accounts roles
- [ ] OAuth 2.0 Device Authorization Grant
- [ ] OIDC CIBA Grant

More Hidden Shadows Lurking in the Depths of the Realm

iteratec



Advanc

This secti

Access Tok

Clien

Clien

Clien
Idle

OAut
Certi
Acce
Enab

Proof Key for Code Exchange Code Challenge Method: plain

Capability config

Client authentication: Off

Authorization: Off

Authentication flow: ✓ Standard flow, ✓ Implicit flow, ☐ OAuth 2.0 De

Clients › Client details › Dedicated scopes

town-gate-dedicated

This is a client scope which includes the de

Mappers | Scope

Full scope allowed: On

Advanced settings

This section is used to configure advanced settings of this client related to OpenID Connect protocol

Access Token Lifespan: Expires in 365 Days

Client Session Idle: Inherits from realm settings

Client Session Max: Inherits from realm settings

the-holy-realm

Realm settings are settings that control the options for users, applications, roles, and groups in the current realm. Learn more

General | Login | Email | Themes | Keys | Events | Localization | Security defenses | Sessions | Tokens | Client polici

Headers | Brute force detection

Brute Force Mode: Disabled

Save | Revert

Authentication

Authentication is the area where you can configure and manage different credential types. Learn more

Flows | Required actions | Policies

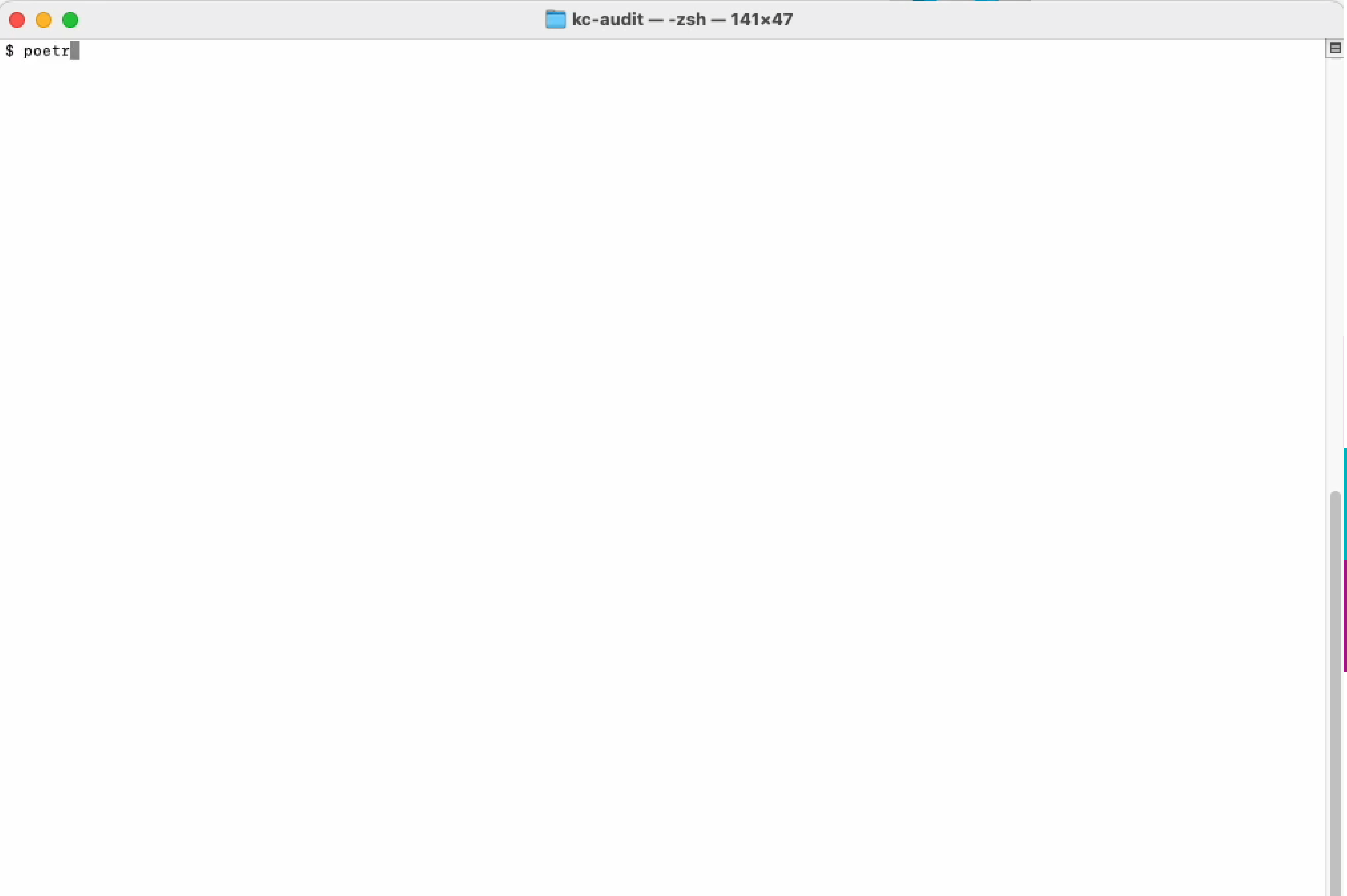
Password policy | OTP Policy | Webauthn Policy | Webauthn Passwordless Policy | CIBA Policy

Add policy

Hashing Iterations *: 50000

Who wants to check this by hand?

For each client, group, scope, …?

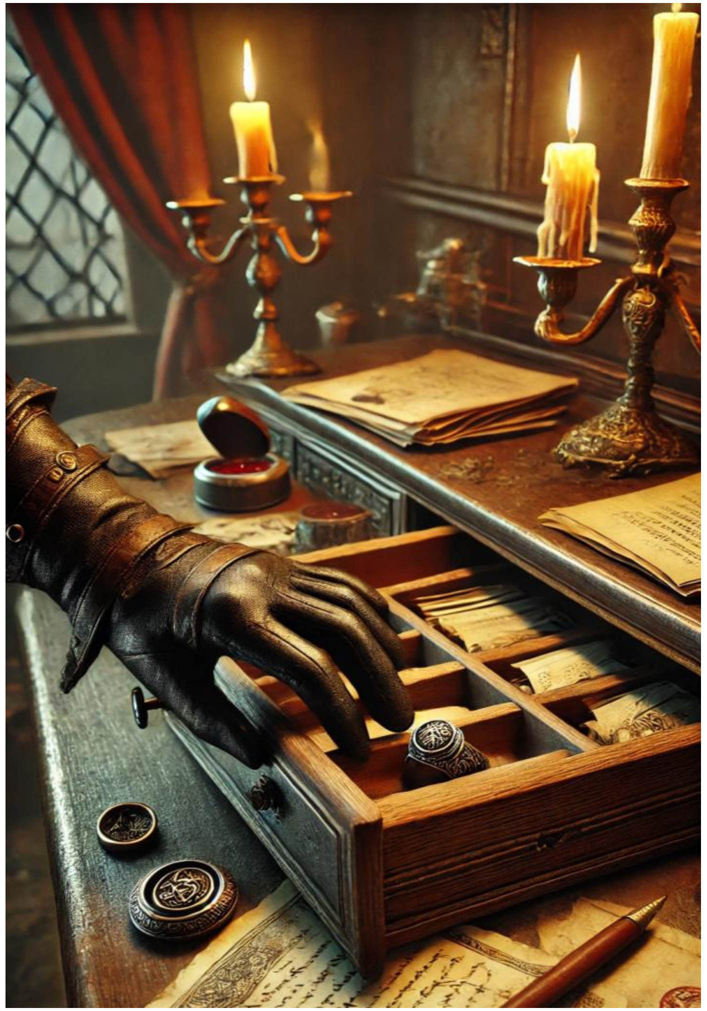


```
$ poetr
```

Introducing kcwarden

- Open Source
- Automatically scan the configuration for errors
- Configurable – mute specific findings you aren't interested in

Step 3: What are we afraid of?



Example #1: M2M – Tech User vs Service Account

iteratec

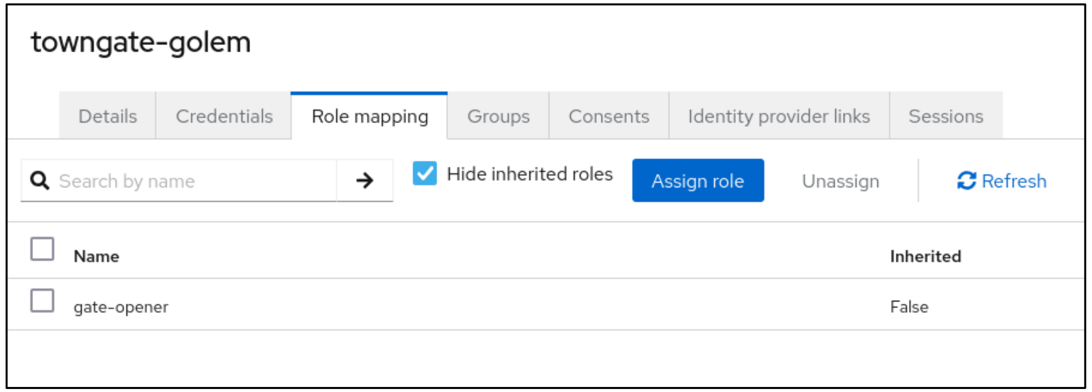
Technical User

- Regular user
- Can be locked by entering wrong passwords
- Requires direct access grants

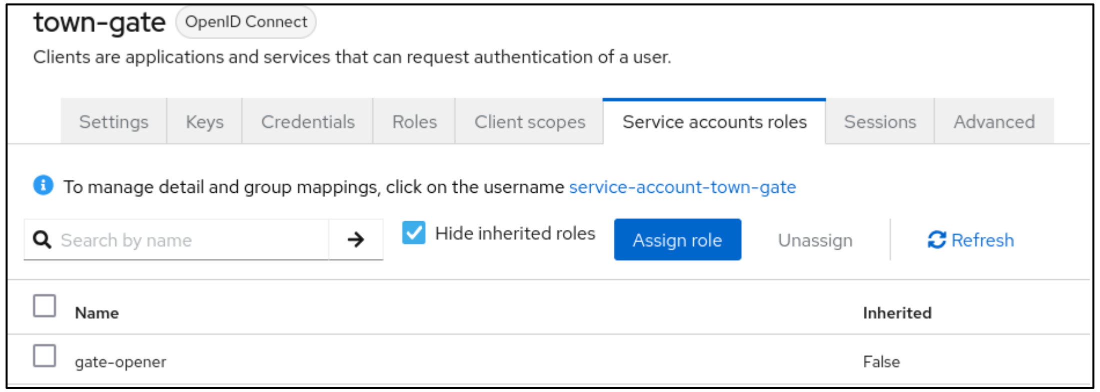
Availability is at risk

Service Account

- Special user bound to a client
- Requires a confidential client
- Access token can be obtained via client authentication


towngate-golem

Name	Inherited
gate-opener	False


town-gate OpenID Connect

Clients are applications and services that can request authentication of a user.

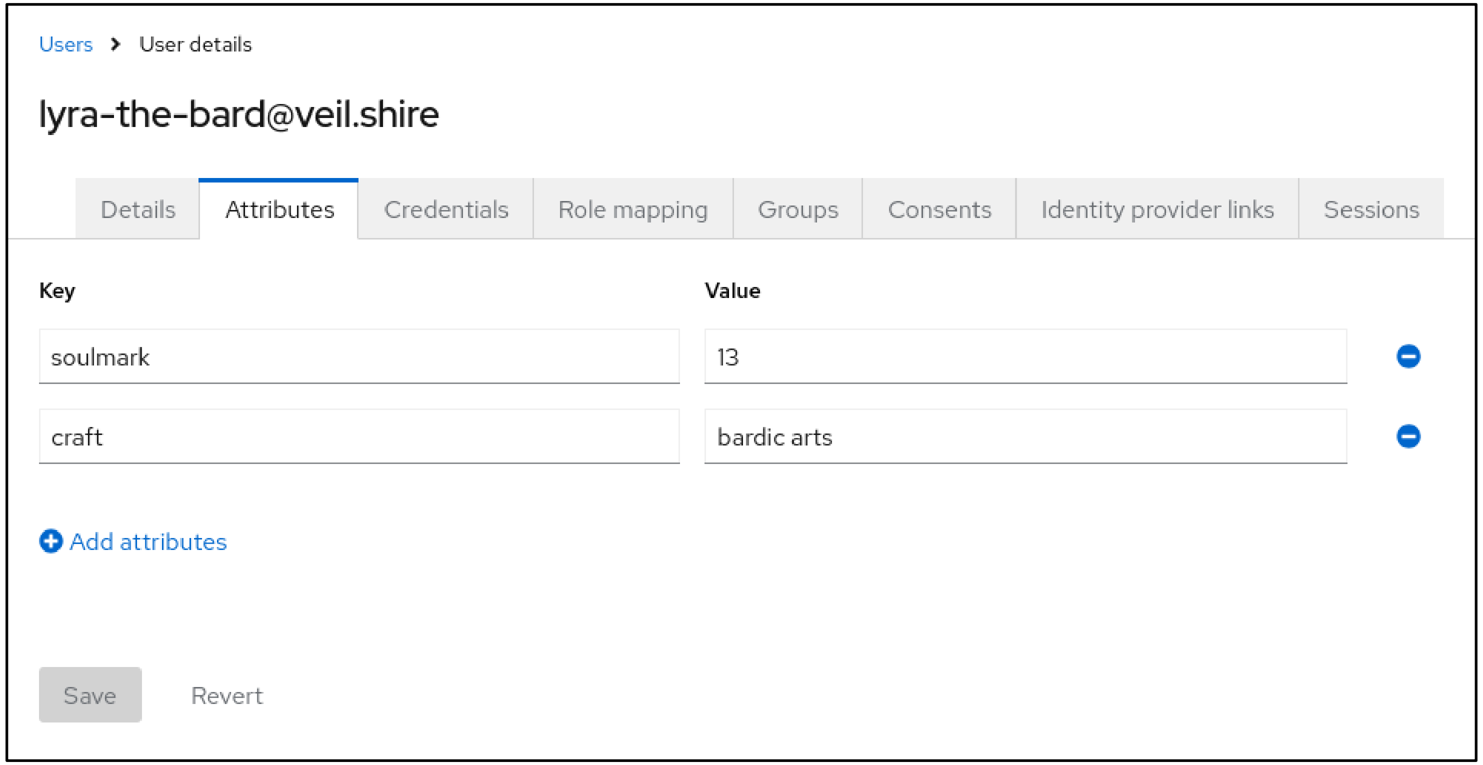
To manage detail and group mappings, click on the username service-account-town-gate

Name	Inherited
gate-opener	False

Example #2: User Attributes

- Additional user account data
- E.g., used to store IDs for other systems
- Depending on configuration, it is probably editable by the user

Account takeover possible



lyra-the-bard@veil.shire

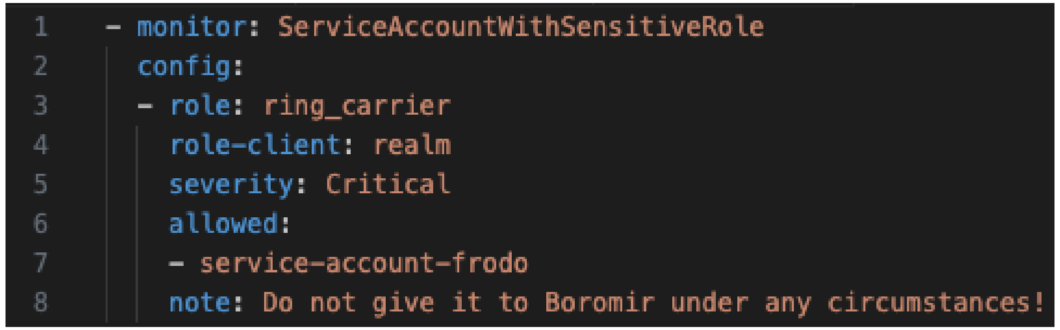
Details | Attributes | Credentials | Role mapping | Groups | Consents | Identity provider links | Sessions

Key	Value
soulmark	13
craft	bardic arts

Add attributes
Save
Revert

Step 4: Sending out Patrols

- Configurations are rarely static in an active project
- Need to establish guardrails to protect against unwanted assignments of roles
- kcwarden supports this:



```
  - monitor: ServiceAccountWithSensitiveRole
    config:
    - role: ring_carrier
      role-client: realm
      severity: Critical
      allowed:
      - service-account-frodo
      note: Do not give it to Boromir under any circumstances!
```

- More information in the docs :)

Example: Continuous Monitoring

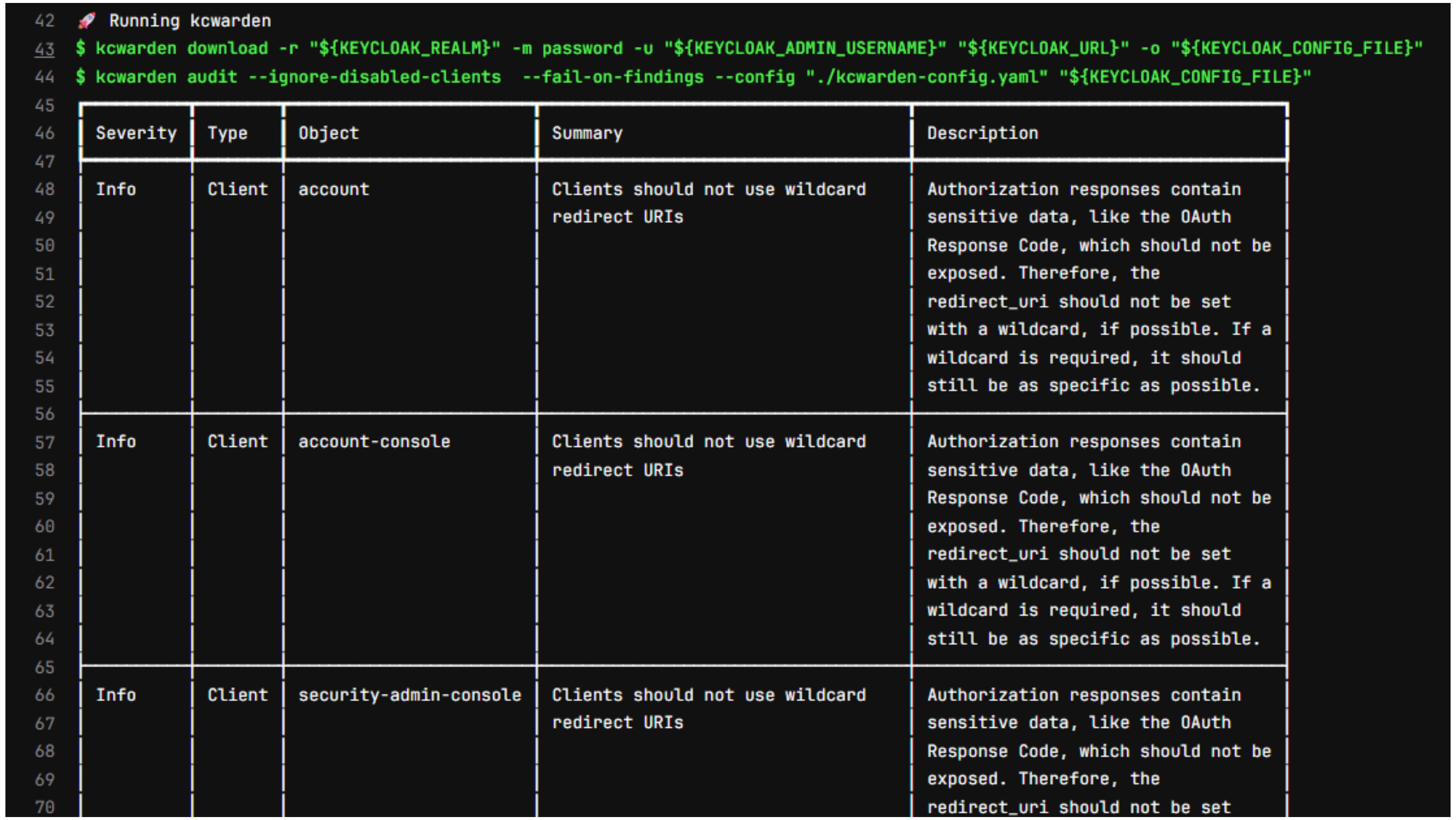
- Establish a config for your project
- Regular runs for alerting
- E.g., using a scheduled GitLab CI/CD pipeline

```yaml
kcwarden:
  stage: audit
  image: ghcr.io/iteratec/kcwarden:latest

  variables:
    KCWARDEN_KEYCLOAK_PASSWORD: ${KEYCLOAK_ADMIN_PASSWORD}

  script:
    - echo "🚀 Running kcwarden"
    - >-
      kcwarden download -r "${KEYCLOAK_REALM}" -m password
      -u "${KEYCLOAK_ADMIN_USERNAME}"
      "${KEYCLOAK_URL}" -o "${KEYCLOAK_CONFIG_FILE}"
    - >-
      kcwarden audit --ignore-disabled-clients --fail-on-findings
      --config "./kcwarden-config.yaml" "${KEYCLOAK_CONFIG_FILE}"
```

Example: Continuous Monitoring

iteratec

- Establish a config for your project
- Regular runs for alerting
- E.g., using a scheduled GitLab CI/CD pipeline



```
🚀 Running kcwarden
$ kcwarden download -r "${KEYCLOAK_REALM}" -m password -u "${KEYCLOAK_ADMIN_USERNAME}" "${KEYCLOAK_URL}" -o "${KEYCLOAK_CONFIG_FILE}"
$ kcwarden audit --ignore-disabled-clients  --fail-on-findings --config "./kcwarden-config.yaml" "${KEYCLOAK_CONFIG_FILE}"

| Severity | Type   | Object                | Summary                         | Description                      |
|          |        |                       |                                 |                                  |
| Info     | Client | account               | Clients should not use wildcard | Authorization responses contain  |
|          |        |                       | redirect URIs                   | sensitive data, like the OAuth   |
|          |        |                       |                                 | Response Code, which should not be |
|          |        |                       |                                 | exposed. Therefore, the          |
|          |        |                       |                                 | redirect_uri should not be set   |
|          |        |                       |                                 | with a wildcard, if possible. If a |
|          |        |                       |                                 | wildcard is required, it should  |
|          |        |                       |                                 | still be as specific as possible. |
|          |        |                       |                                 |                                  |
| Info     | Client | account-console       | Clients should not use wildcard | Authorization responses contain  |
|          |        |                       | redirect URIs                   | sensitive data, like the OAuth   |
|          |        |                       |                                 | Response Code, which should not be |
|          |        |                       |                                 | exposed. Therefore, the          |
|          |        |                       |                                 | redirect_uri should not be set   |
|          |        |                       |                                 | with a wildcard, if possible. If a |
|          |        |                       |                                 | wildcard is required, it should  |
|          |        |                       |                                 | still be as specific as possible. |
|          |        |                       |                                 |                                  |
| Info     | Client | security-admin-console | Clients should not use wildcard | Authorization responses contain |
|          |        |                       | redirect URIs                   | sensitive data, like the OAuth   |
|          |        |                       |                                 | Response Code, which should not be |
|          |        |                       |                                 | exposed. Therefore, the          |
|          |        |                       |                                 | redirect_uri should not be set   |
```

Conclusion

iteratec

- Keeping Keycloak secure is difficult.
- Problems can lurk in the basics, but also in the advanced features, role assignments, or many other locations.
- kcwarden can augment your work and let you focus on the interesting parts.
- Contribute your own rules, it's fairly straightforward :)

Find kcwarden on GitHub:

/iteratec/kcwarden

Dr. Max Maaß

max.maass@iteratec.com
@hacksilon@infosec.exchange

Tim Walter

tim.walter@iteratec.com
@twwd@infosec.exchange