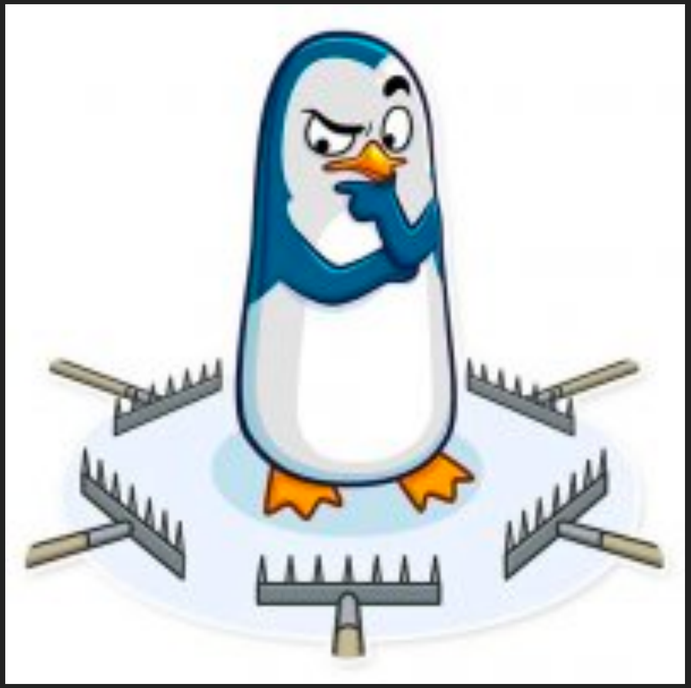
INTRO TO

OAUTH, OPENIDC, SSO & KEYCLOACK

PROLOGUE

- how I dealt with oauth



WHY ?

- Дублирование логики авторизации (login, remind-password, reset-password, logout)
- “up to date” с требованиями по хранению секретный данных (md5, sha256, sha512, bcrypt?)
- “better user experience” технология единого входа во все сервисы SSO
- Контроль доступа сотрудников из одного пункта (active-directory)

PREREQUISITES

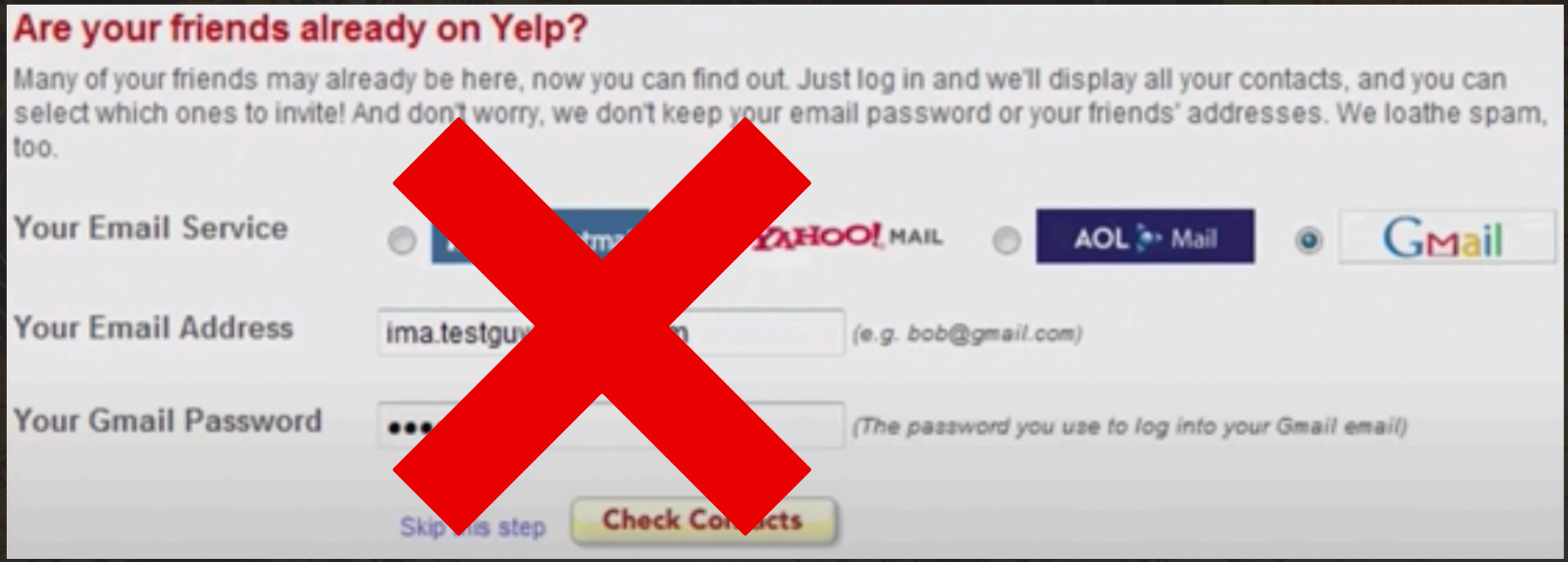
- SYMMETRIC VS ASYMMETRIC CRYPTOGRAPHY (PUB/PRIV KEYS)

- JSON WEB TOKENS

AUTHORIZATION IS ABOUT:

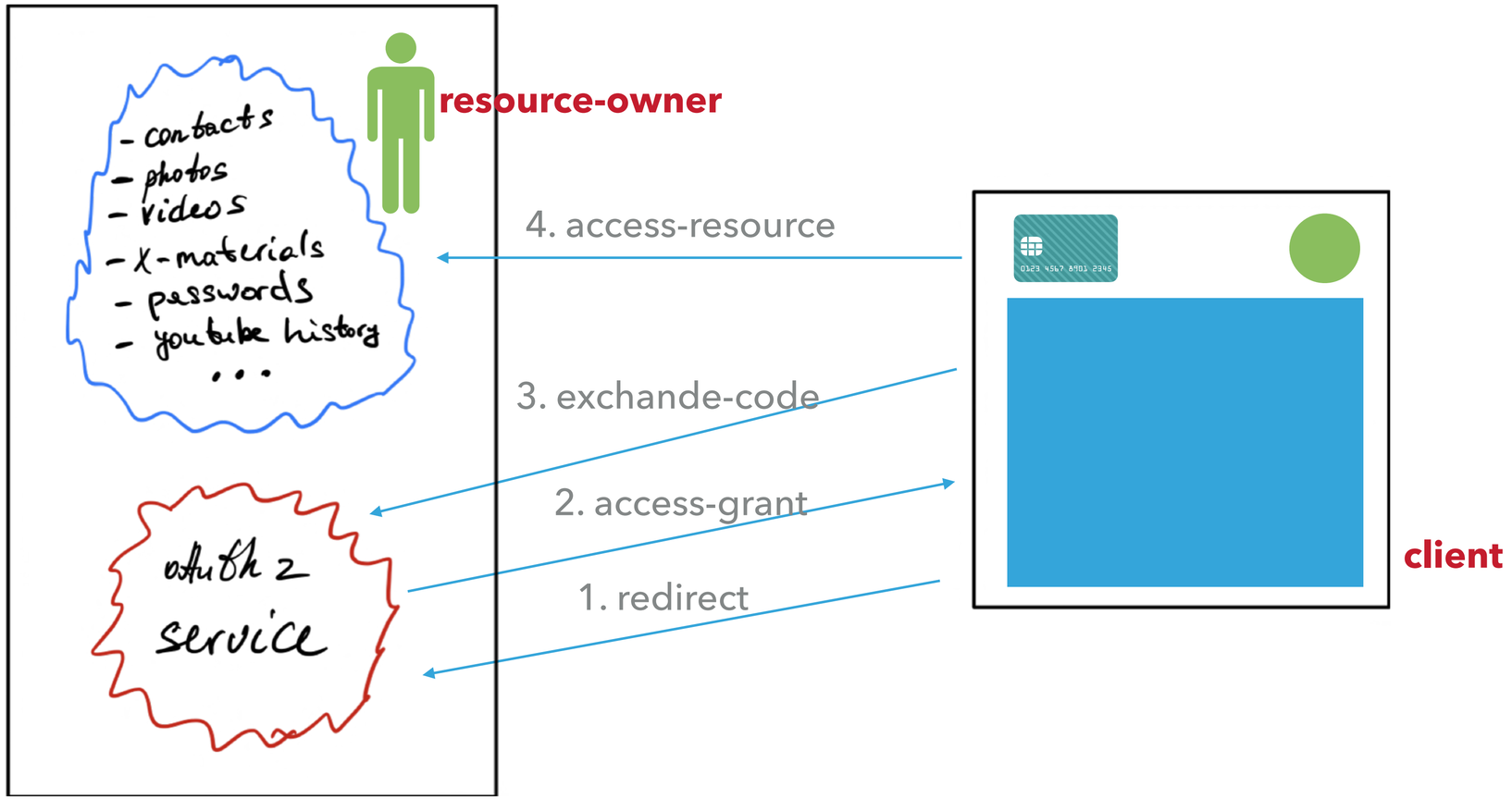
AND NOT EQUALS TO AUTHENTICATION

HISTORY OF THE PROBLEM



HOW OAUTH SOLVES THAT PROBLEM (OVERVIEW)

AUTHORIZATION FLOW: CODE



HOW OAUTH SOLVES THAT PROBLEM (STEP BY STEP)

STEP 1:

- redirect from **client** to **auth-server** with params: *client_id, redirect_url, scope, response_type*
- user performs login (if not already logged in)
- and sees **consent** screen based on **scopes**
- if agreed, **access_grant** action happens

```
REDIRECT https://auth.server.com?
&client_id=e6169686775e4c71f29d
&redirect_uri=https://myapp.com/
&scope=profile
&response_type=code
```

STEP 2:

- auth-server redirects back to client-app with **code**

```
REDIRECT https://myapp.com/?code=12345678
```

STEP 3:

- client-app **exchanges** the code with auth-server and gets the actual **access_token**

```
POST https://auth.server.com
client_id=e6169686775e4c71f29d
secret=secret
code=12345678
```

STEP 4:

- now client-app is ready to access data from **resource-server**

```
RESPONSE
{
 "access_token": "egJsdasdINvmsjn12...",
 "token_type": "Bearer"
}
```

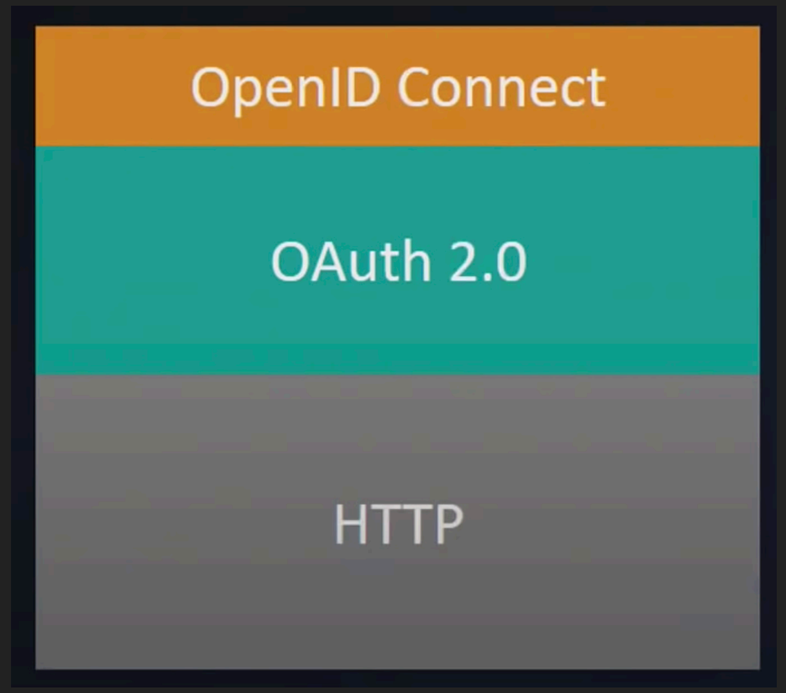
DEMO TIME

- https://oauthdebugger.com
- https://docs.github.com/en/developers/apps/building-oauth-apps/authorizing-oauth-apps
- https://docs.github.com/en/developers/apps/building-oauth-apps/scopes-for-oauth-apps
- https://api.slack.com/scopes

OAUTH IS FOR SOLVING

- DELEGATED AUTHORIZATION PROBLEM

WHAT ABOUT AUTHENTICATION ?



AUTHENTICATION

```
{
  "id_token": "eyJhbGc2iOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ..."
  "access_token": "eyJh78bGc3USUzI1NiIssImtpZCISUzI1NiIsWdkazcifQ...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token":"SUzI1NiIsImtpZCI6IjFlOWdk..."
}
```

AUTHORIZATION

```
{
  "access_token":"gho_16C7e42F292c6912E7710c838347Ae178B4a",
  "scope":"repo,gist",
  "token_type":"bearer"
}
```

- Open source identity & access management solution
- https://github.com/keycloak/keycloak
- https://www.keycloak.org

DEMO TIME

- console
- feature review
- frontend example
- backend example

LINKS

https://github.com/thomasdarimont/awesome-keycloak

https://www.keycloak.org/guides

https://oauthdebugger.com

https://auth0.com

https://www.youtube.com/watch?v=996OiexHze0

https://www.youtube.com/watch?v=duawSV69LDI

QUESTIONS ?