I Don’t Care About Security

And Neither Should You

About Me

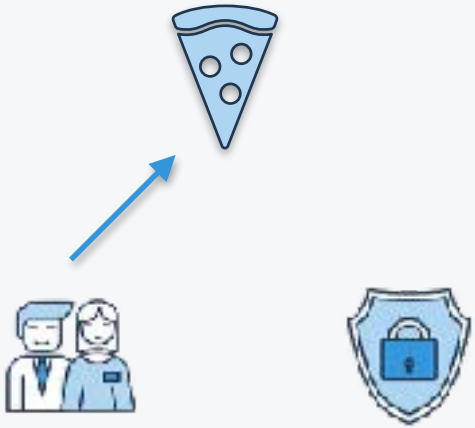
@joel__lord

joellord


99¢ FRESH PIZZA
2 SLICE & 1 CAN OR WATER $ 2.75
PEPPERONI · MUSHROOM
SAUSAGE · EXTRA CHEESE
BLACK OLIVE · PINEAPPLE
SPECIALS!
18" PIE
FOR RENT

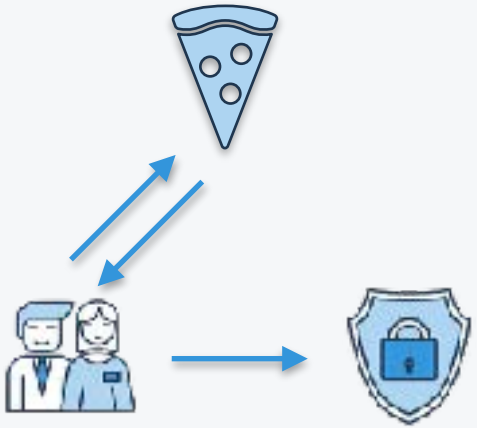
OAuth - Flows

Authorization Code



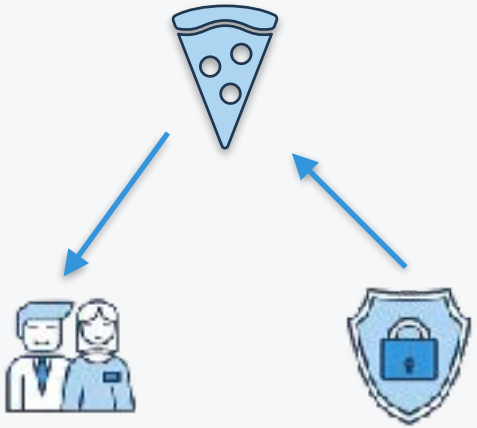
OAuth - Flows

Authorization Code



OAuth - Flows

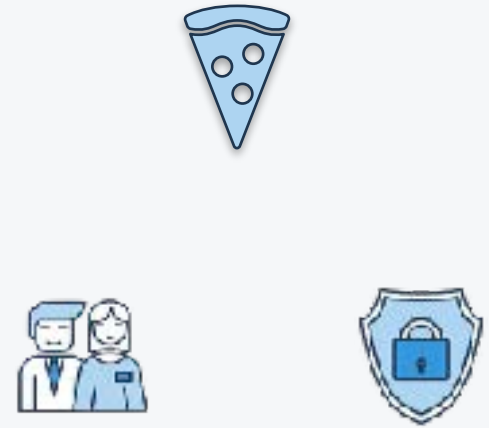
Authorization Code



That reminds me of OAuth!

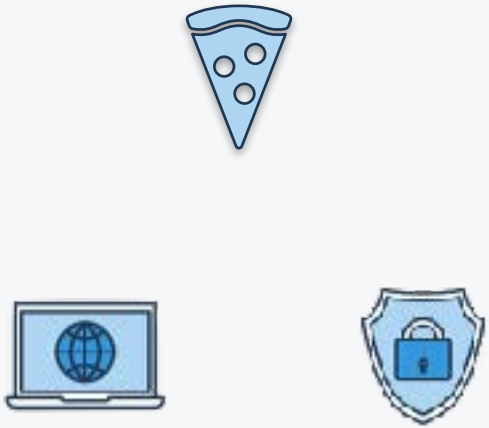
OAuth - Flows

Authorization Code



OAuth - Flows

Authorization Code



OAuth - Flows

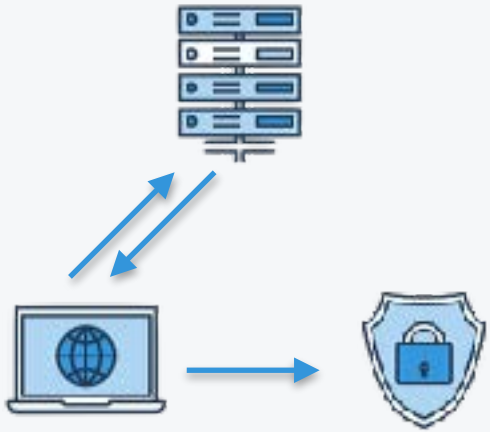
Authorization Code

OAuth - Flows

Authorization Code

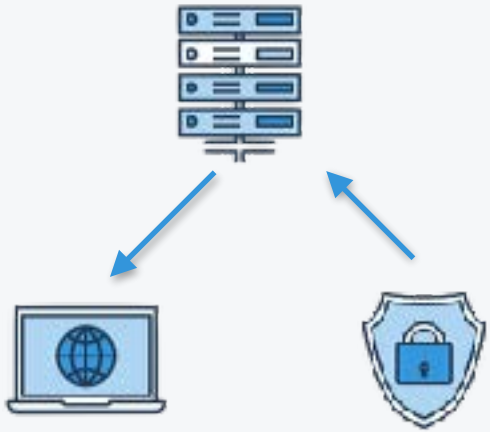
OAuth - Flows

Authorization Code



OAuth - Flows

Authorization Code

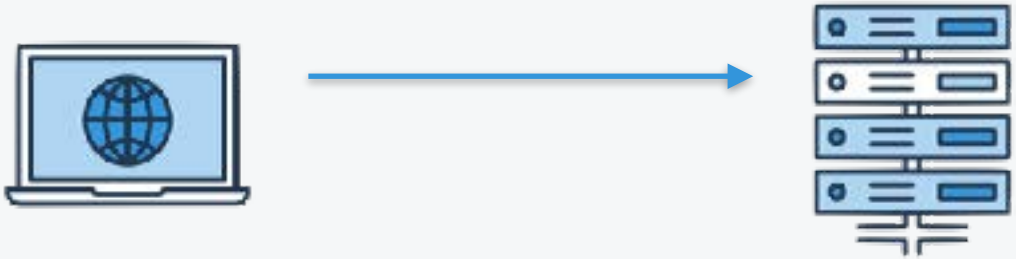


But Why?


Delegation!

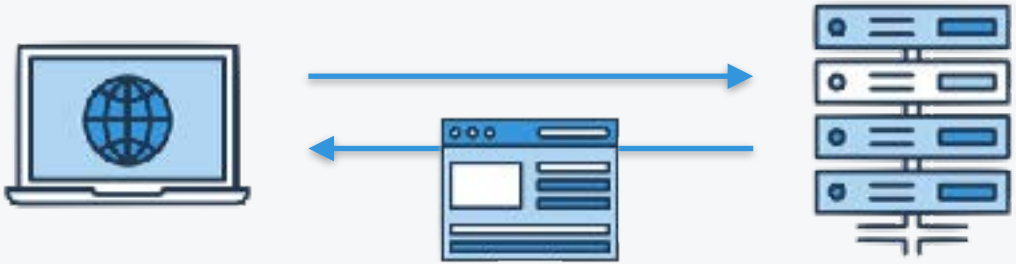
Traditional Applications

- Browser requests a login page



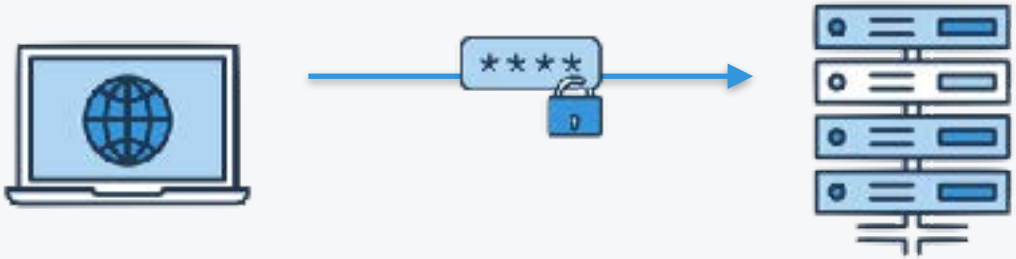
Traditional Applications

- Browser requests a login page



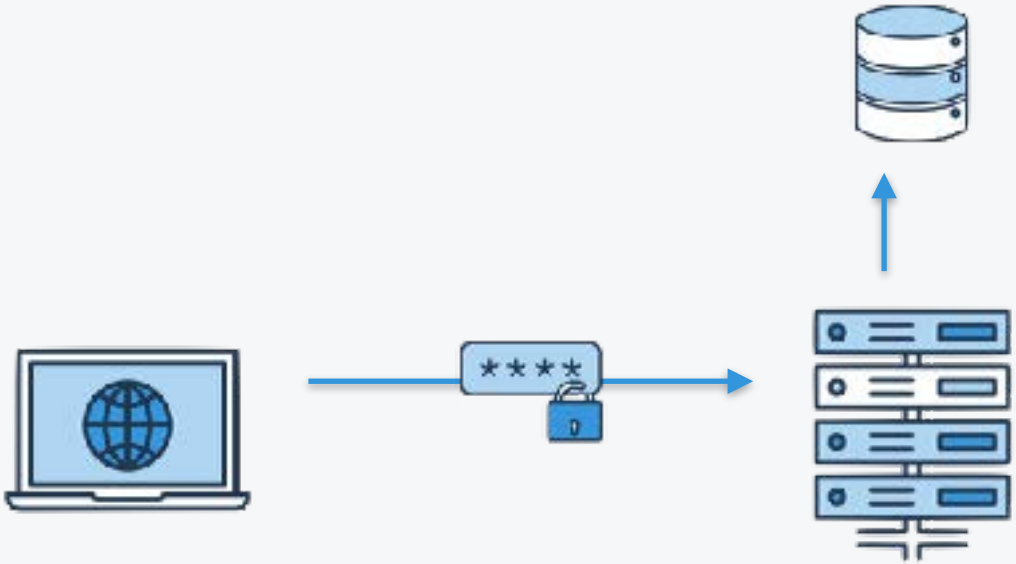
Traditional Applications

- Browser requests a login page



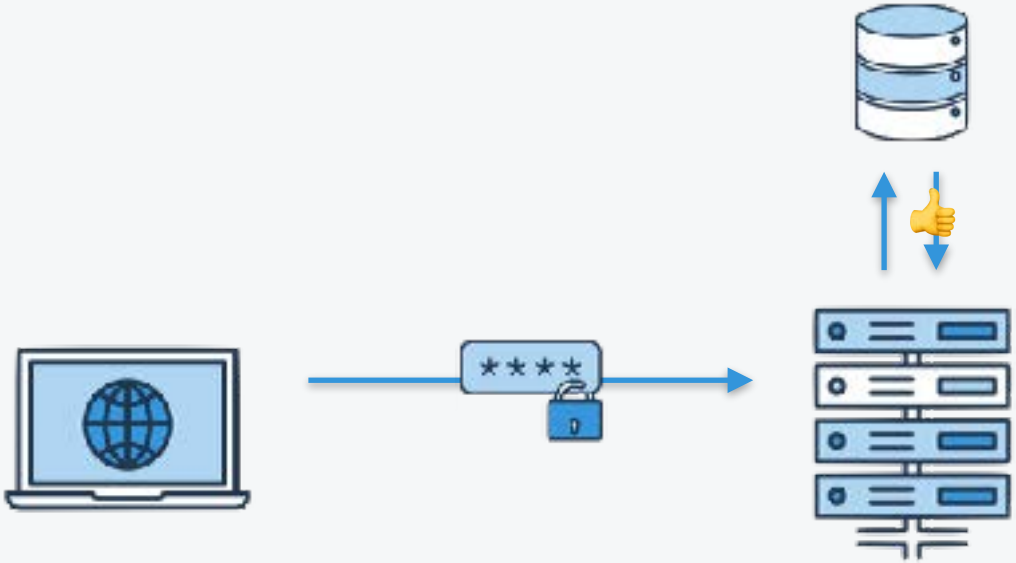
Traditional Applications

- Browser requests a login page
- The server validates on its database



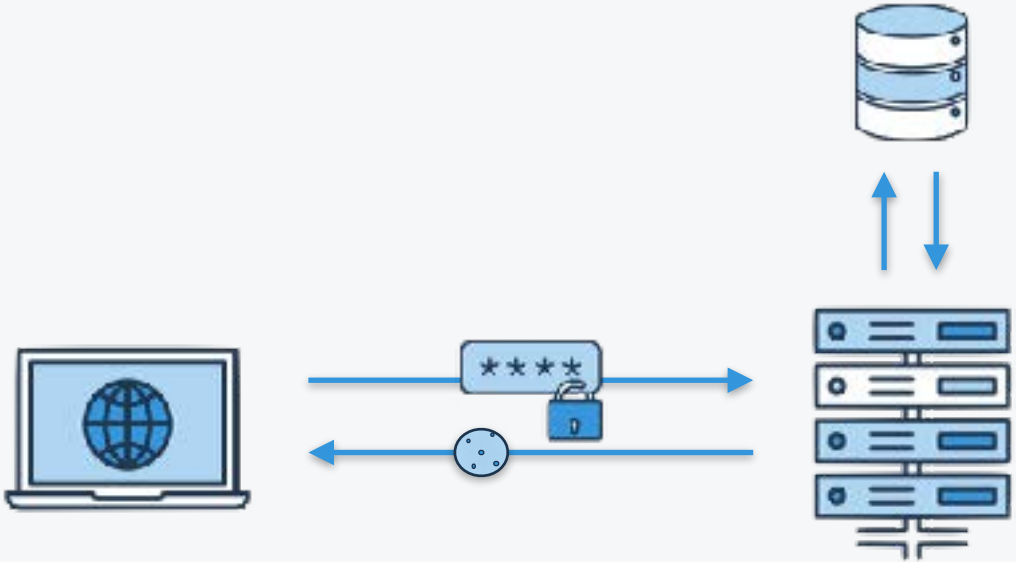
Traditional Applications

- Browser requests a login page
- The server validates on its database



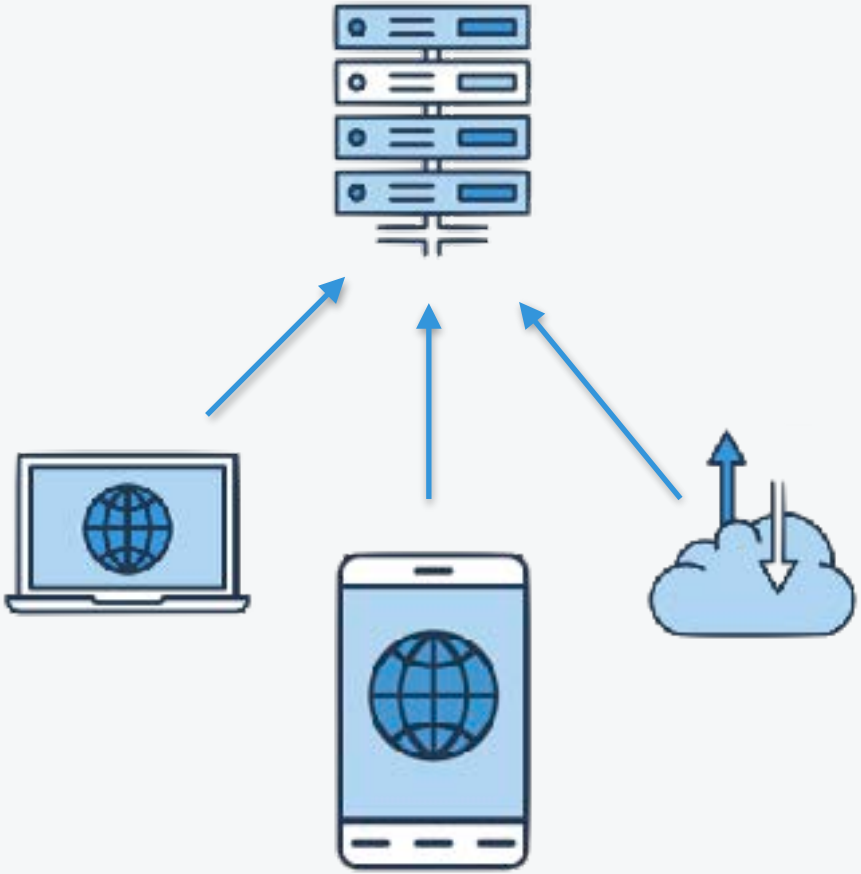
Traditional Applications

- Browser requests a login page
- The server validates on its database
- It creates a session and provides a cookie identifier



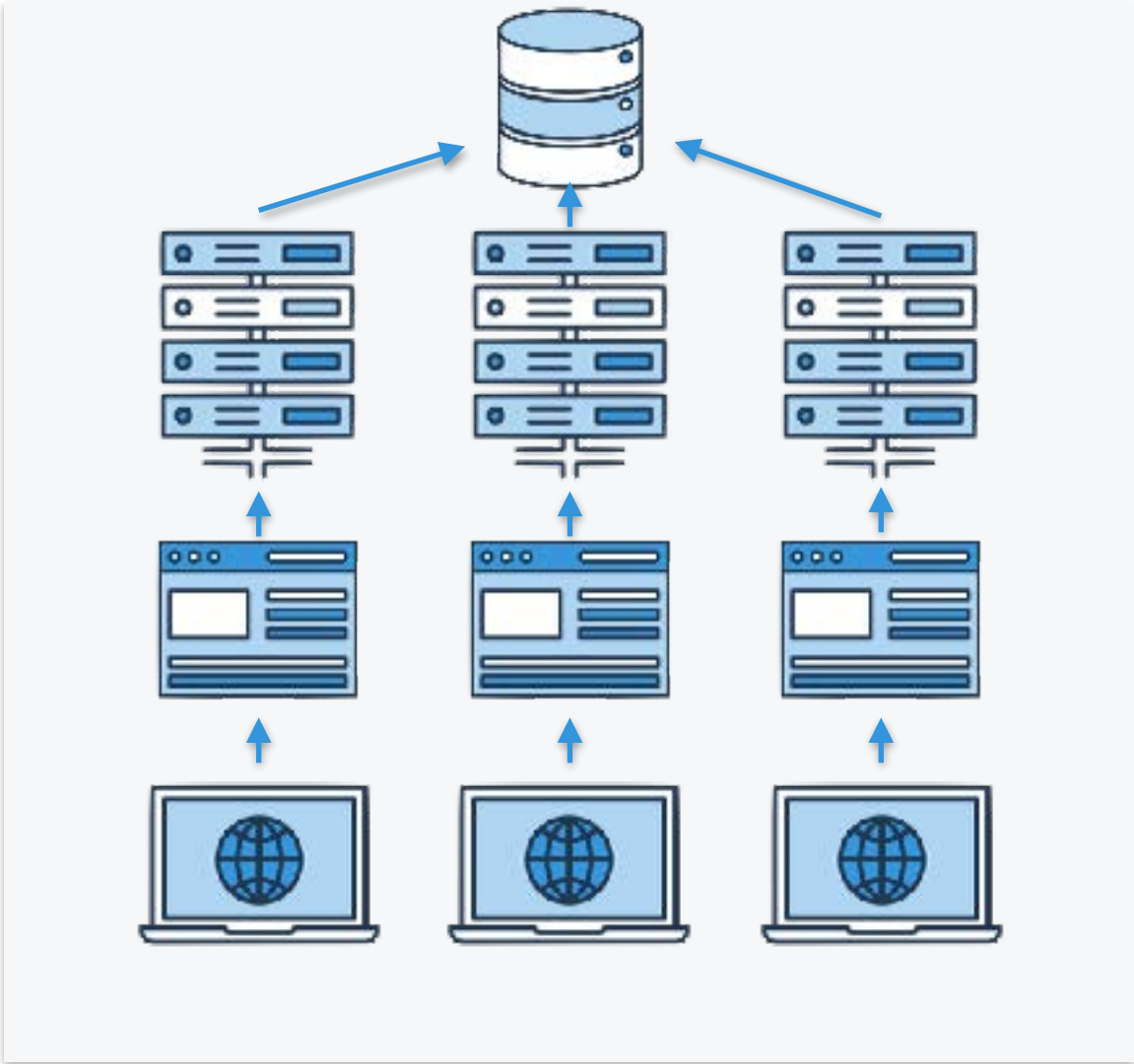
What's wrong with traditional auth?

- Multiple platforms connecting to your application



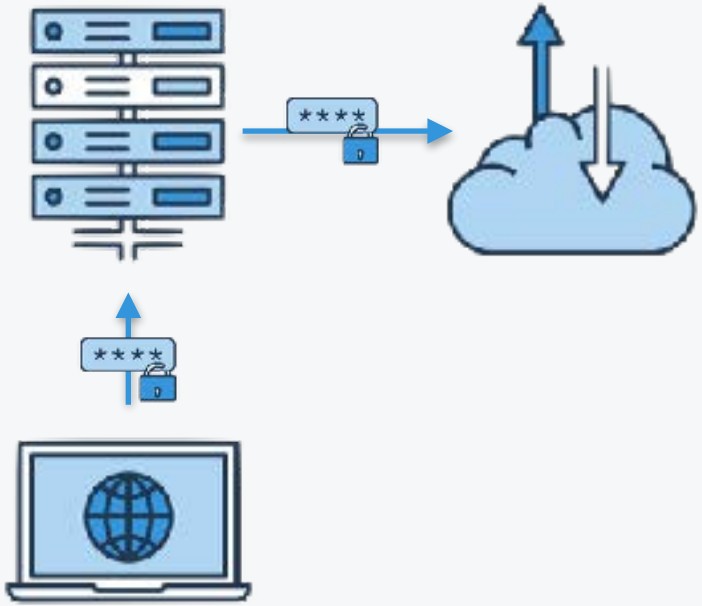
What's wrong with traditional auth?

- Multiple platforms connecting to your application
- Tightly coupled



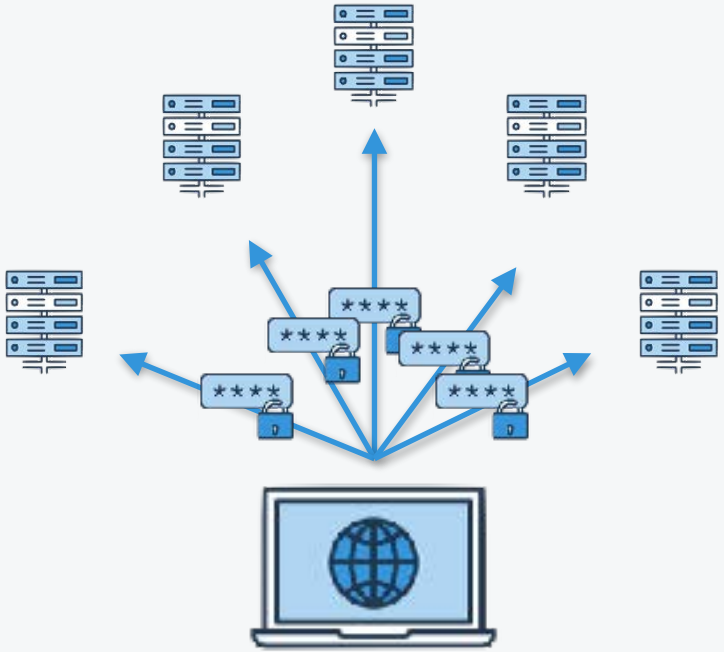
What's wrong with traditional auth?

- Multiple platforms connecting to your application
- Tightly coupled
- Sharing credentials to connect to another API



What's wrong with traditional auth?

- Multiple platforms connecting to your application
- Tightly coupled
- Sharing credentials to connect to another API
- Users have a gazillion passwords to remember, which increases security risks



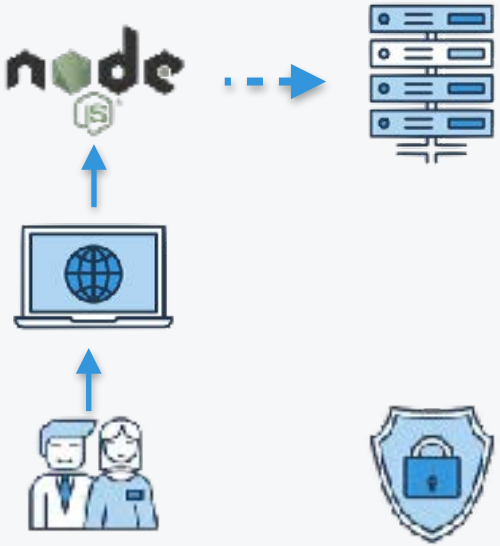
OAuth

OAuth - The Flows

Authorization Code

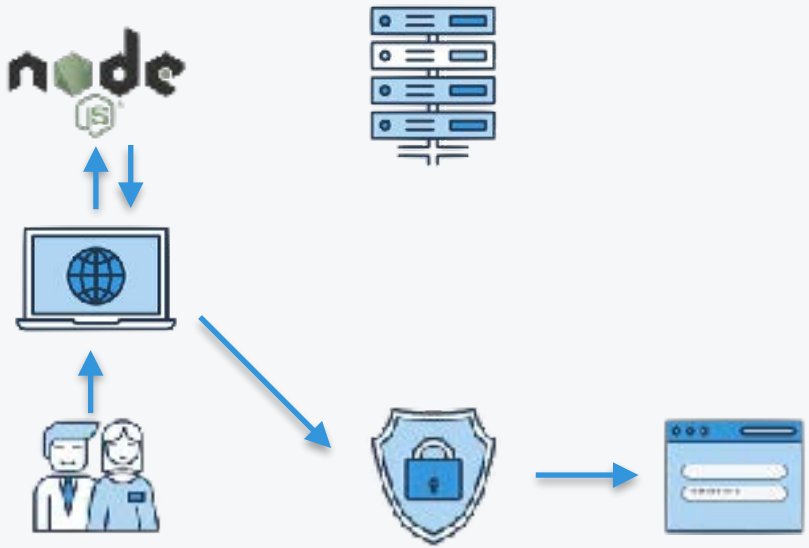
Authentication Flows

Authorization Code



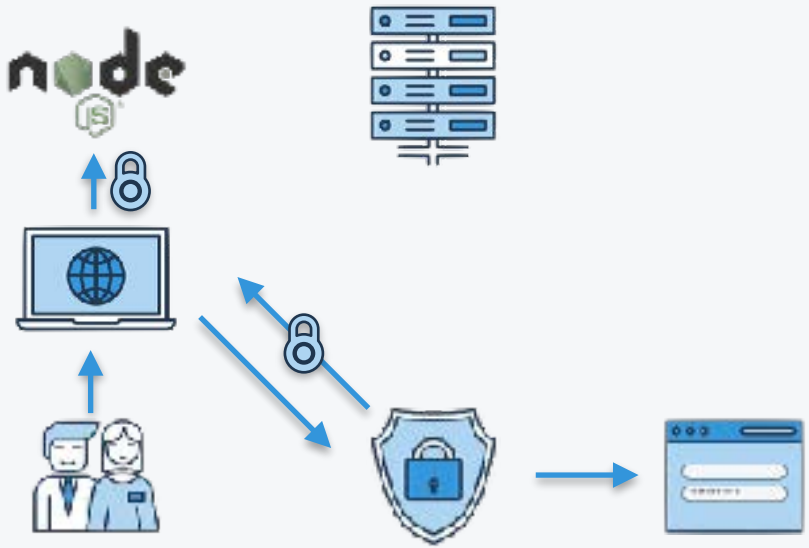
Authentication Flows

Authorization Code



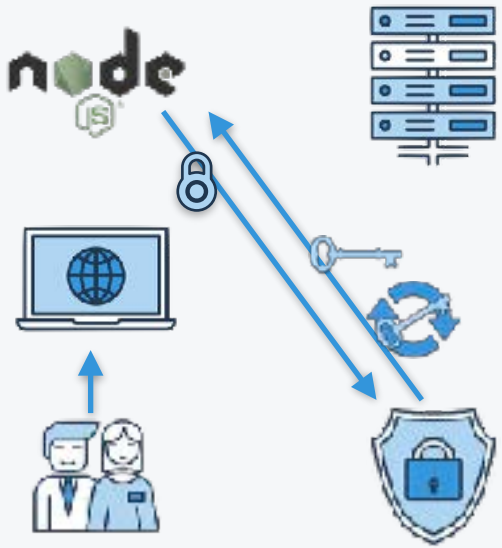
Authentication Flows

Authorization Code



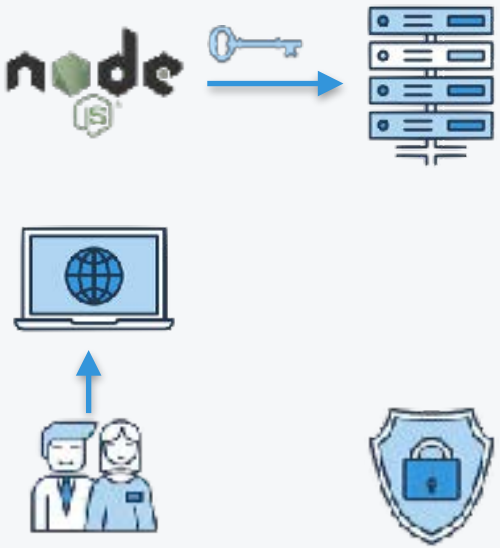
Authentication Flows

Authorization Code



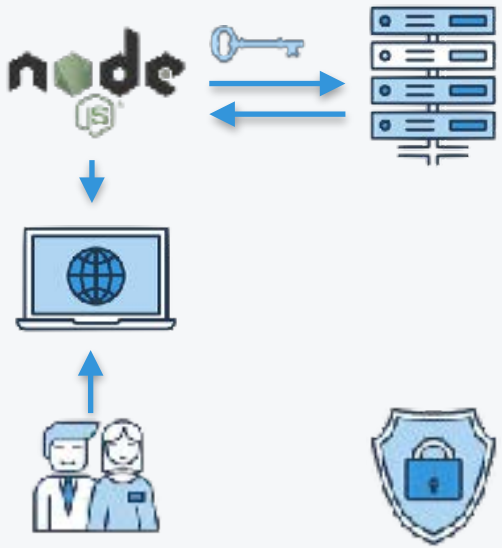
Authentication Flows

Authorization Code



Authentication Flows

Authorization Code



OAuth - The Flows

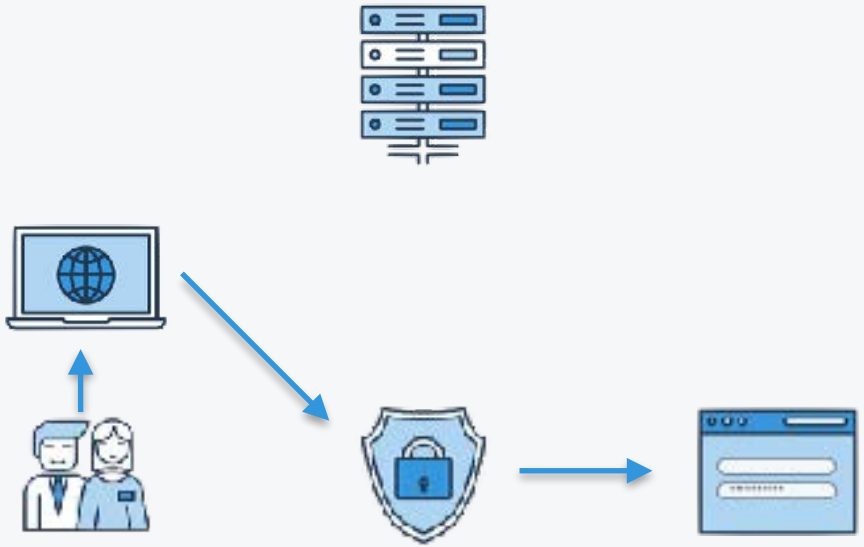
Implicit Flow

Authentication Flows

Implicit Flow

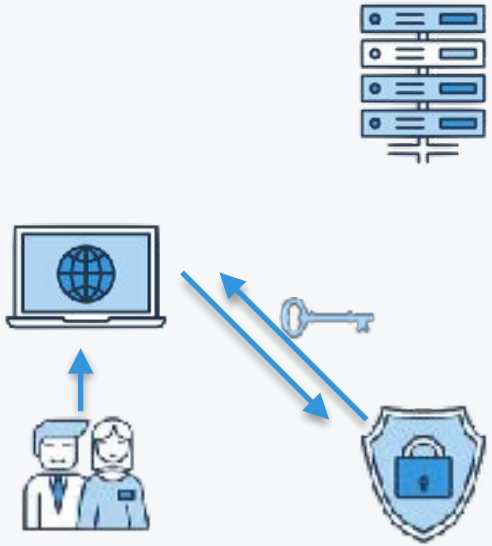
Authentication Flows

Implicit Flow



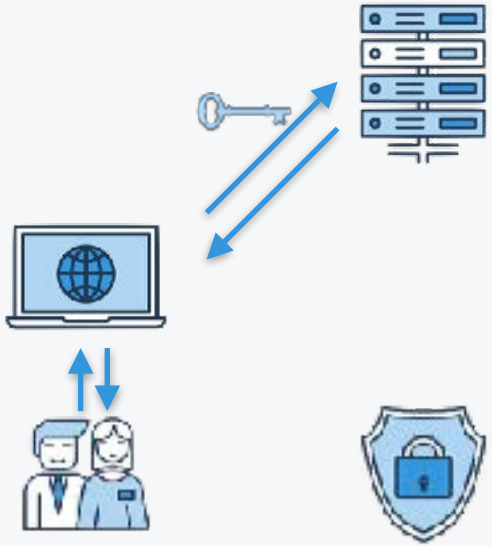
Authentication Flows

Implicit Flow



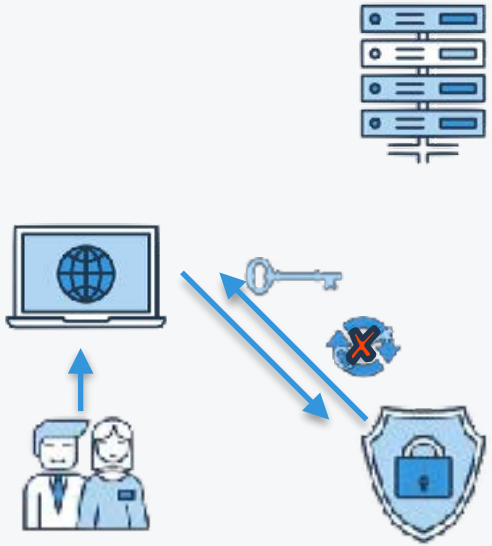
Authentication Flows

Implicit Flow



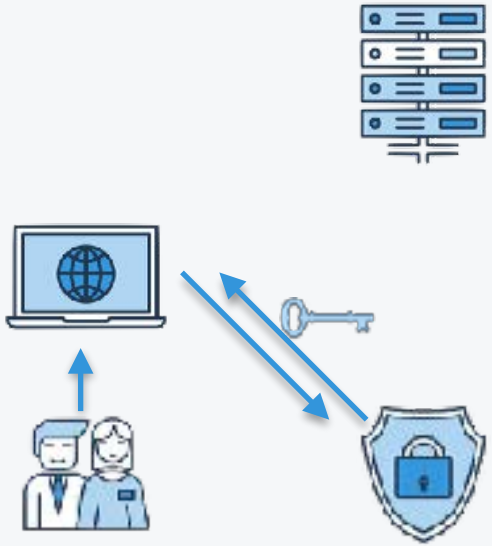
Authentication Flows

Implicit Flow



Authentication Flows

Implicit Flow



Tokens 101

OAuth

Tokens

Access Token

- Give you access to a resource
- Controls access to your API
- Short lived

Refresh Token

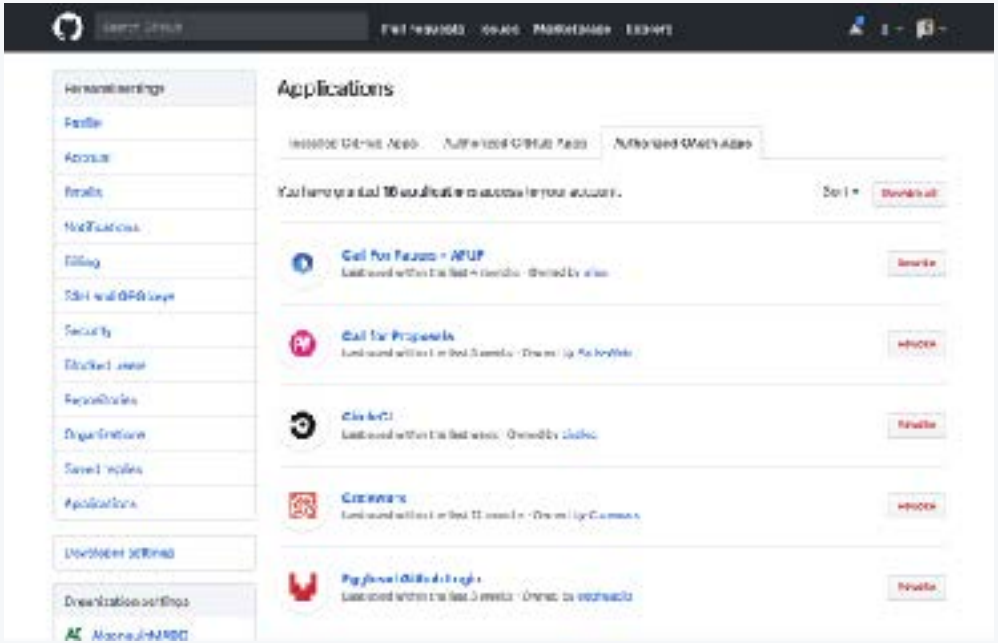
- Enables you to get a new token
- Longed lived
- Can be revoked

OAuth

Tokens

Refresh Token

- Enables you to get a new token
- Longed lived
- Can be revoked



OAuth

Tokens

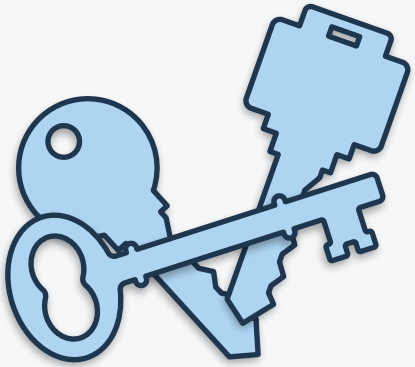
Refresh Token

- Enables you to get a new token
- Longed lived
- Can be revoked

OAuth

Tokens

- WS-Federated
- SAML
- JWT
- Custom stuff
- More…



JSON Web Token

- Header
- Payload
- Signature

Header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Payload

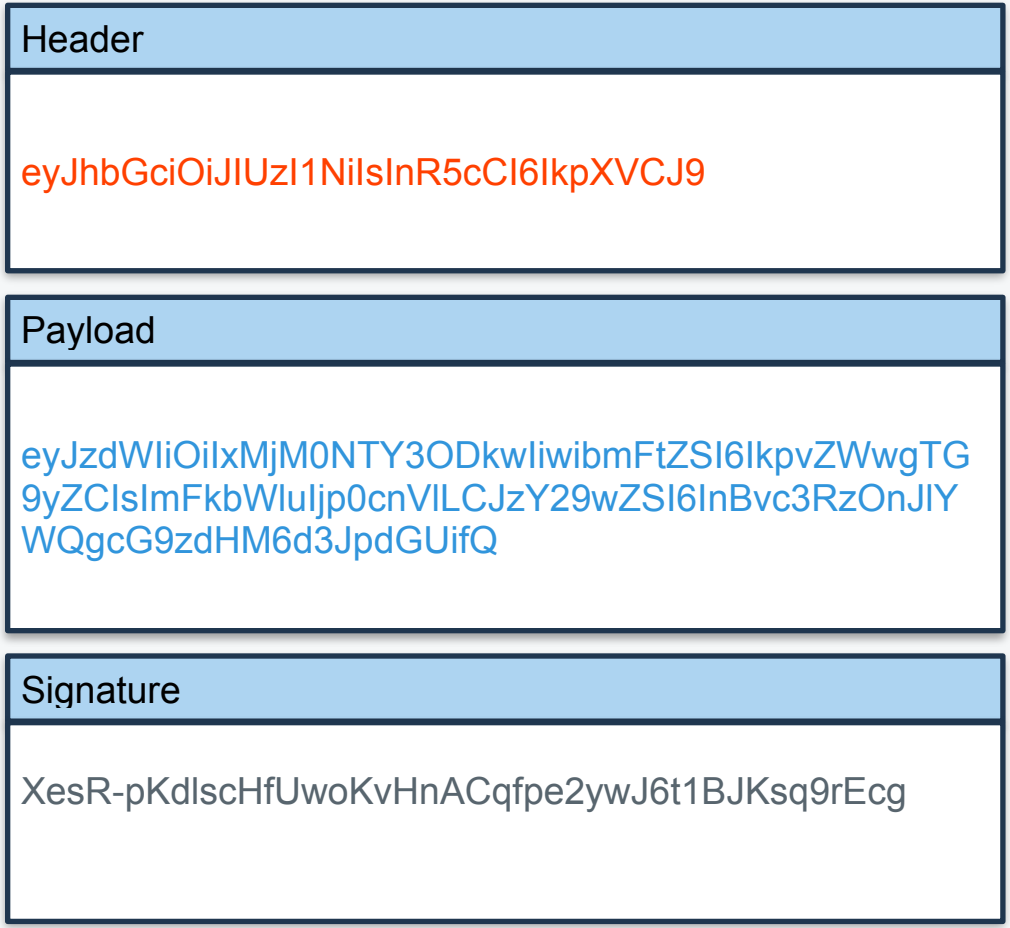
```
{
  "sub": "1234567890",
  "name": "Joel Lord",
  "scope": "posts:read posts:write"
}
```

Signature

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload), secret)
```

JSON Web Token

- Header
- Payload
- Signature



JSON Web Token

- Header
- Payload
- Signature

JSON Web Token

- Header
- Payload
- Signature

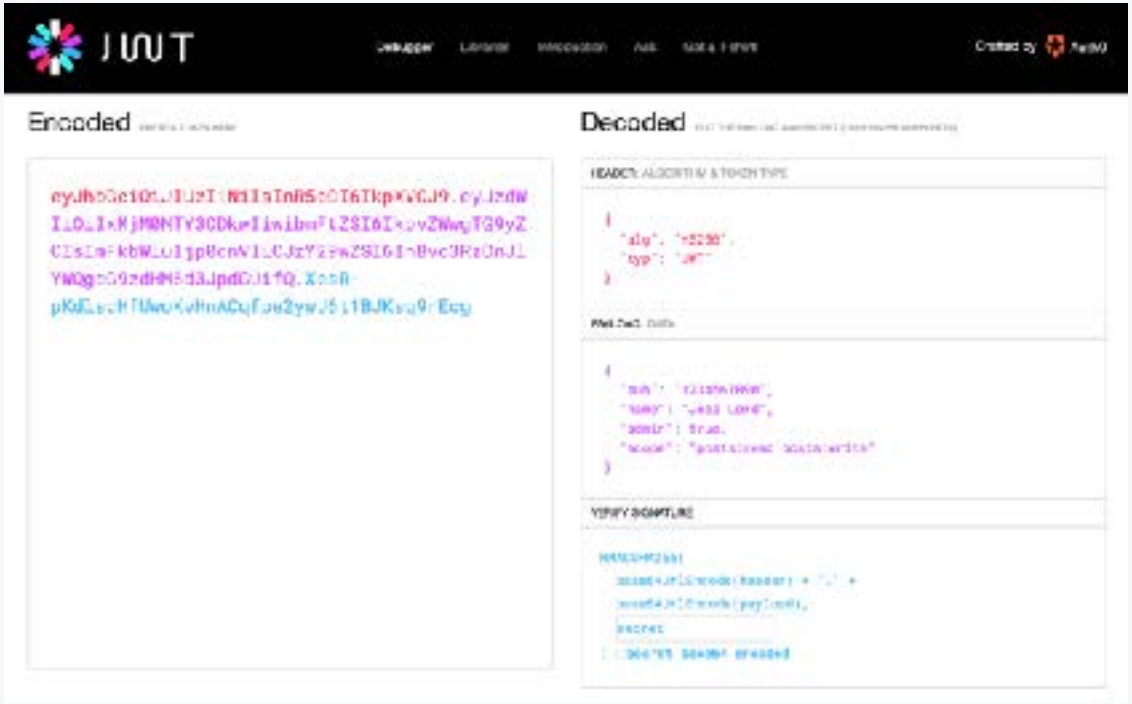


Image: https://jwt.io

Codiiiing Time!

Auth Server

```
var express = require('express');
var Webtask = require('webtask-tools');
var bodyParser = require('body-parser');
var jwt = require("jsonwebtoken");
var app = express();

var users = [
  {id: 1, username: "joellord", password: "joellord"},
  {id: 2, username: "guest", password: "guest"}
];

app.use(bodyParser.urlencoded());

app.get("/login", function(req, res) {
  var loginForm = "<form method='post'><input type=hidden name=callback value='" +
req.query.callback + "'><input type=text name=username /><input type=text name=password /
><input type=submit></form>";
  res.status(200).send(loginForm);
});

app.post("/login", function(req, res) {
  if (!req.body.username || !req.body.password) return res.status(400).send("Need
username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.redirect(req.body.callback + "#access_token=" + token);
});

app.get('*', function (req, res) {
  res.sendStatus(404);
});
```

API

```
var express    = require('express');
var Webtask    = require('webtask-tools');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var jwt = require("jsonwebtoken");
var app = express();

var users = [
  {id: 1, username: "joellord", password: "joellord"},
  {id: 2, username: "guest", password: "guest"}
];

app.use(bodyParser.json());

app.post("/login", function(req, res) {
  if (!req.body.username || !req.body.password) return res.status(400).send("Need
username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.status(200).send({token: token});
});

app.get('*', function (req, res) {
  res.sendStatus(404);
});

module.exports = Webtask.fromExpress(app);
```

Auth Server

```
var express = require('express');
var bodyParser = require('body-parser');
var jwt = require("jsonwebtoken");
var app = express();

// ...
```

Auth Server

```
var express = require('express');
var bodyParser = require('body-parser');
var jwt = require("jsonwebtoken");
var app = express();

// ...
```

Auth Server

```
var express = require('express');
var bodyParser = require('body-parser');
var jwt = require("jsonwebtoken");
var app = express();

// ...
```

Auth Server

```
var express = require('express');
var bodyParser = require('body-parser');
var jwt = require("jsonwebtoken");
var app = express();

// ...
```

Auth Server

```
// Requires ...

var users = [
  {id: 1, username: "joellord", password: "joellord"},
  {id: 2, username: "guest", password: "guest"}
];
```

Auth Server

```
// Requires ...

var users = [...];

app.use(bodyParser.urlencoded());

app.post("/login", function(req, res) {
  // POST for login
});

app.get('*', function (req, res) {
  res.sendStatus(404);
});
```

Auth Server

```
// Requires ...

var users = [...];

app.use(bodyParser.urlencoded());

app.post("/login", function(req, res) {
  // POST for login
});

app.get('*', function (req, res) {
  res.sendStatus(404);
});
```

Auth Server

```
app.post("/login", function(req, res) {
  // POST for login
  if (!req.body.username || !req.body.password)
    return res.status(400).send("Need username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.redirect(req.body.callback + "#access_token=" + token);
});
```

Auth Server

```
app.post("/login", function(req, res) {
  // POST for login
  if (!req.body.username || !req.body.password)
    return res.status(400).send("Need username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.redirect(req.body.callback + "#access_token=" + token);
});
```

Auth Server

```
app.post("/login", function(req, res) {
  // POST for login
  if (!req.body.username || !req.body.password)
    return res.status(400).send("Need username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.redirect(req.body.callback + "#access_token=" + token);
});
```

Auth Server

```
app.post("/login", function(req, res) {
  // POST for login
  if (!req.body.username || !req.body.password)
    return res.status(400).send("Need username and password");

  var user = users.find(function(u) {
    return u.username === req.body.username && u.password === req.body.password;
  });
  if (!user) return res.status(401).send("User not found");

  var token = jwt.sign({
    sub: user.id,
    scope: "api:read",
    username: user.username
  }, "mysupersecret", {expiresIn: "10 minutes"});

  res.redirect(req.body.callback + "#access_token=" + token);
});
```

Auth Server

```
// Requires ...

var users = [...];

app.use(bodyParser.urlencoded());

app.post("/login", function(req, res) {
  // POST for login
});

app.get('*', function (req, res) {
  res.sendStatus(404);
});

app.listen(8080, () => console.log("Auth server running on 8080"));}
```

API

```
var express    = require('express');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var expressjwt = require("express-jwt");
var app = express();
```

API

```
var express    = require('express');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var expressjwt = require("express-jwt");
var app = express();
```

API

```
var express    = require('express');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var expressjwt = require("express-jwt");
var app = express();
```

API

```
var express    = require('express');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var expressjwt = require("express-jwt");
var app = express();
```

API

```
var express    = require('express');
var bodyParser = require('body-parser');
var randopeep = require("randopeep");
var expressjwt = require("express-jwt");
var app = express();
```

API

```
// Requires ...

var jwtCheck = expressjwt({
    secret: "mysupersecret"
});
```

API

```
// Requires and config ...

app.get("/headline", function(req, res) {
    // Unprotected
    res.status(200).send(randopeep.clickbait.headline());
});

app.get("/protected/headline", jwtCheck, function(req, res) {
    // Protected
    res.status(200).send(randopeep.clickbait.headline("Joel Lord"));
});

app.get('*', function (req, res) {
    res.sendStatus(404);
});
```

API

```
// Requires and config ...

app.get("/headline", function(req, res) {
    // Unprotected
    res.status(200).send(randopeep.clickbait.headline());
});

app.get("/protected/headline", jwtCheck, function(req, res) {
    // Protected
    res.status(200).send(randopeep.clickbait.headline("Joel Lord"));
});

app.get('*', function (req, res) {
    res.sendStatus(404);
});
```

API

```
// Requires and config ...

app.get("/headline", function(req, res) {
    // Unprotected
    res.status(200).send(randopeep.clickbait.headline());
});

app.get("/protected/headline", jwtCheck, function(req, res) {
    // Protected
    res.status(200).send(randopeep.clickbait.headline("Joel Lord"));
});

app.get('*', function (req, res) {
    res.sendStatus(404);
});
```

API

```
// Requires and config ...

app.get("/headline", function(req, res) {
    // Unprotected
    res.status(200).send(randopeep.clickbait.headline());
});

app.get("/protected/headline", jwtCheck, function(req, res) {
    // Protected
    res.status(200).send(randopeep.clickbait.headline("Joel Lord"));
});
```

```
app.get('*', function (req, res) {
    res.sendStatus(404);
});
```

API

```
// Requires and config ...

app.get("/headline", function(req, res) {
    // Unprotected
});

app.get("/protected/headline", jwtCheck, function(req, res) {
    // Protected
});

app.get('*', function (req, res) {
    res.sendStatus(404);
});

app.listen(8888, () => console.log("API listening on 8888"));
```

Front-End

Add the headers

Live Demo

https://github.com/joellord/secure-spa-auth0


Delegation!


BATES
MOTEL
OFFICE


HOME


CANADA
PASSPORT
PASSEPORT


Human Resources Development Canada
Développement des ressources humaines Canada
SOCIAL INSURANCE NUMBER
NUMÉRO D'ASSURANCE SOCIALE
123 123 123
PETAR PETROVIC

Front


Québec
Permis de conduire
4a L1531-171274-08
1 LAPOINTE
2 ANNE-MARIE
3 Date de naissance (a-m-j) : 1978-12-17
8 333, BOULEVARD JEAN-LESAGE
APP. 432
QUÉBEC (QC) G1K 8J6
15 Sexe : F
9 Classe(s) : 1 2 3 4A 4B 4C 5 6A
12 Cond. : A C
16 Taille (cm) : 168
18 Yeux : BLEU
5 No de référence : PBMH92V70
4b Valide le 2014-10-01 au Expire le 2022-12-17
Paiement exigé chaque année à votre date anniversaire de naissance
Driver's licence number (4a)
Last name (1)
First name(s) (2)
Date of birth (3)
Cardholder's address (8)
Driver's licence class(es) (9)*
Sex (15)
Condition(s) (12)*
Height (cm) (16)
Eye colour (18)
Endorsement(s) (9a)*
Reference number (5)
Date of expiry (4b)
Date of issue (4a)

Back


CLASSE(S) : Tous les types de véhicules routiers. Taxis automobiles.
CONDITION(S) : A : Lunettes ou lentilles correctrices, C : Appareil auditif, classe 2, 4A, 4B ou 4C.
MENTION(S) : F : Freinage personnalisé, M : Transmission manuelle, T : Train routier.
Apposer l'autocollant de votre nouvelle adresse
Pour tout renseignement
1 800 361-7620
Société de l'assurance automobile
Québec
Pour vérifier la validité d'un permis
1 900 565-1212 (frais exigés)
Services aussi disponibles au www.saaq.gouv.qc.ca
(2009-10)
Definitions of the licence classes, conditions and endorsements listed on the front
Bar code
The bar code corresponds to the driver's licence number.
Telephone number for more information
Telephone number to check the validity of a licence (fees apply)
Two-dimensional bar code
The two-dimensional bar code (2D) contains all licence information except the photo and signature.

Introducing OpenID Connect

OpenID Connect

- Built on top of OAuth 2.0
- OpenID Connect (OIDC) is to OpenID what Javascript is to Java
- Provides Identity Tokens in JWT format
- Uses a /userinfo endpoint to provide the info

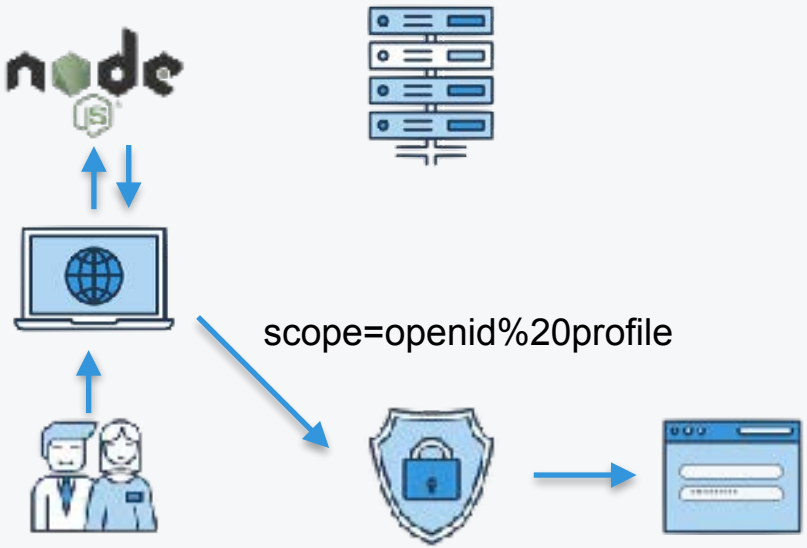
OpenID Connect

Scopes

- openid
- profile
- email
- address
- phone

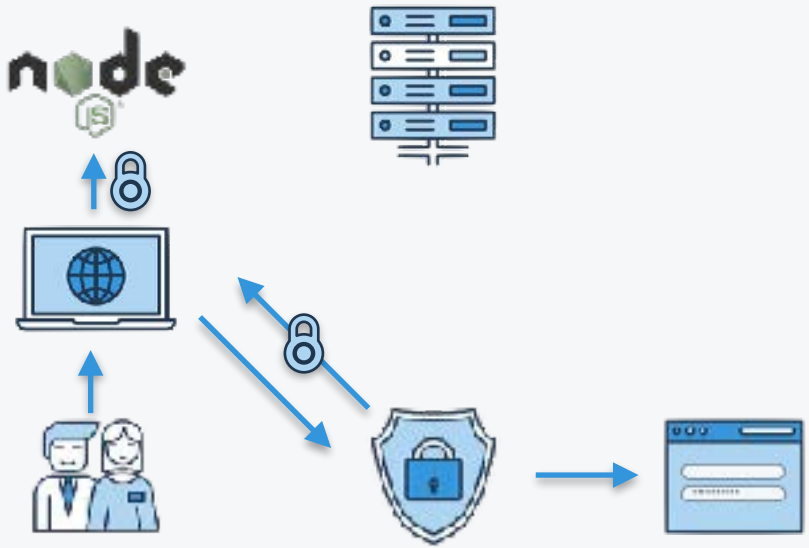
OpenID Connect Flows

Authorization Code



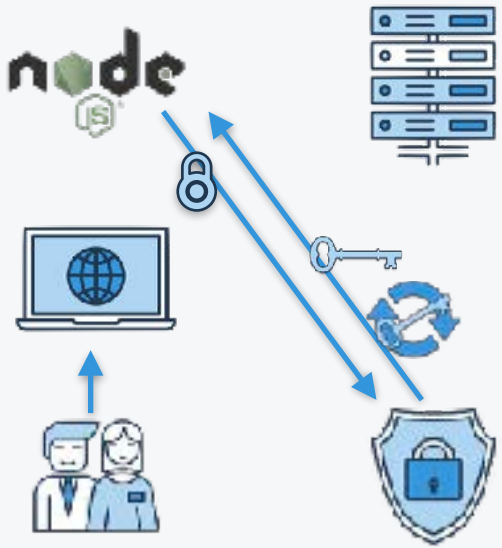
Authentication Flows

Authorization Code



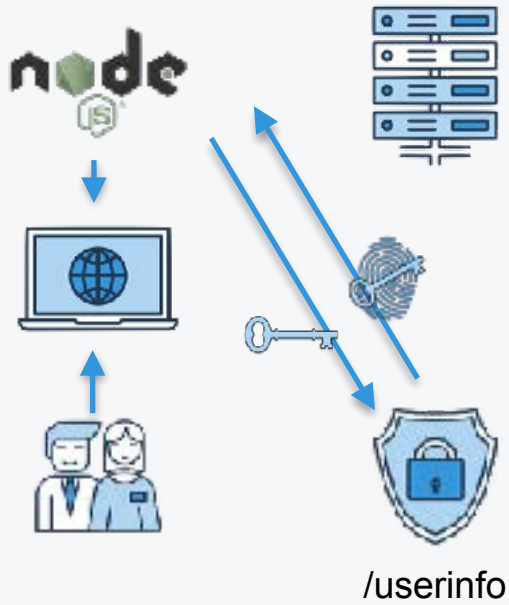
Authentication Flows

Authorization Code



Authentication Flows

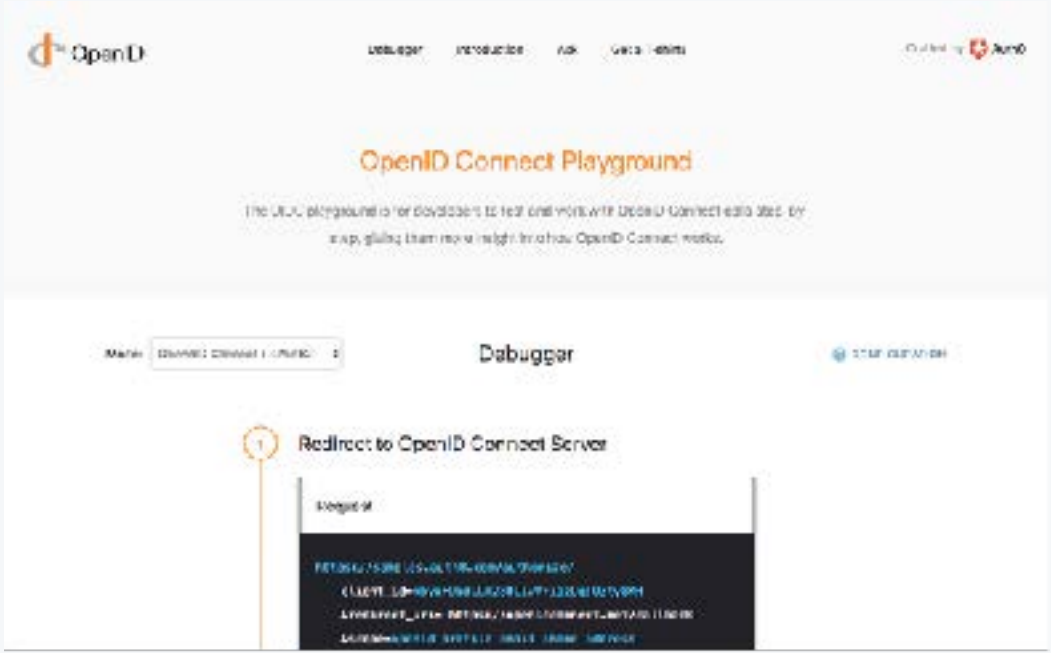
Authorization Code



OpenID Connect

Full flow

https://openidconnect.net




Delegation!

I Don’t Care About Security

iJS, London, UK
April 11, 2018

@joel__lord

joellord