Authentication in Angular

Jacob Wenger / @_jwngr

AngularJS Meetup / June 15 & 16, 2015

Goal

Authentication sucks and is difficult...

Goal

Authentication sucks and is difficult...

...but not for you!

Authentication Sucks!

Authentication Sucks!

Requires your own server

Authentication Sucks!

Requires your own server

User management is tedious

Authentication Sucks!

Requires your own server

User management is tedious

Myriad OAuth implementations

Authentication Sucks!

Requires your own server

User management is tedious

Myriad OAuth implementations

Security is hard

Authentication Sucks!

Requires your own server

User management is tedious

Myriad OAuth implementations

Security is hard

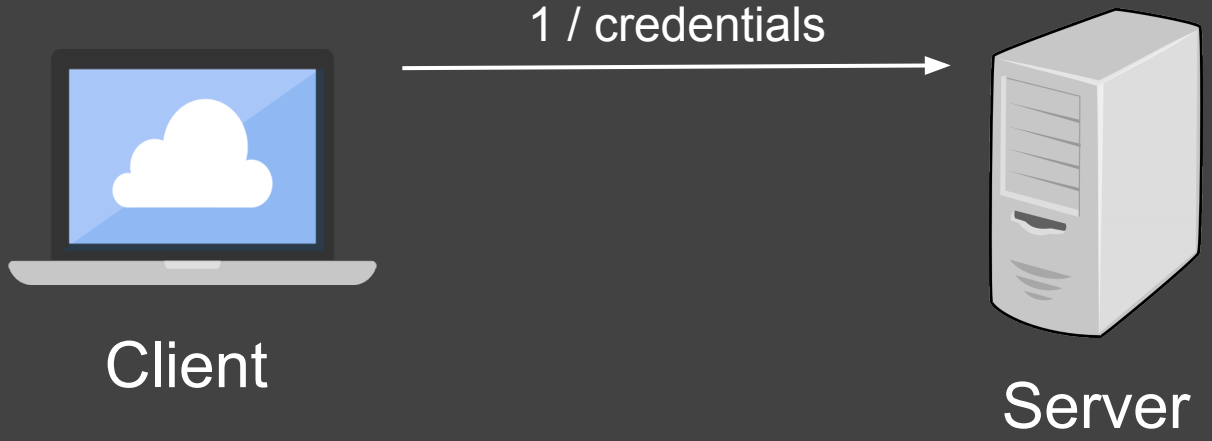
Offline is harder

Cookie-based Auth

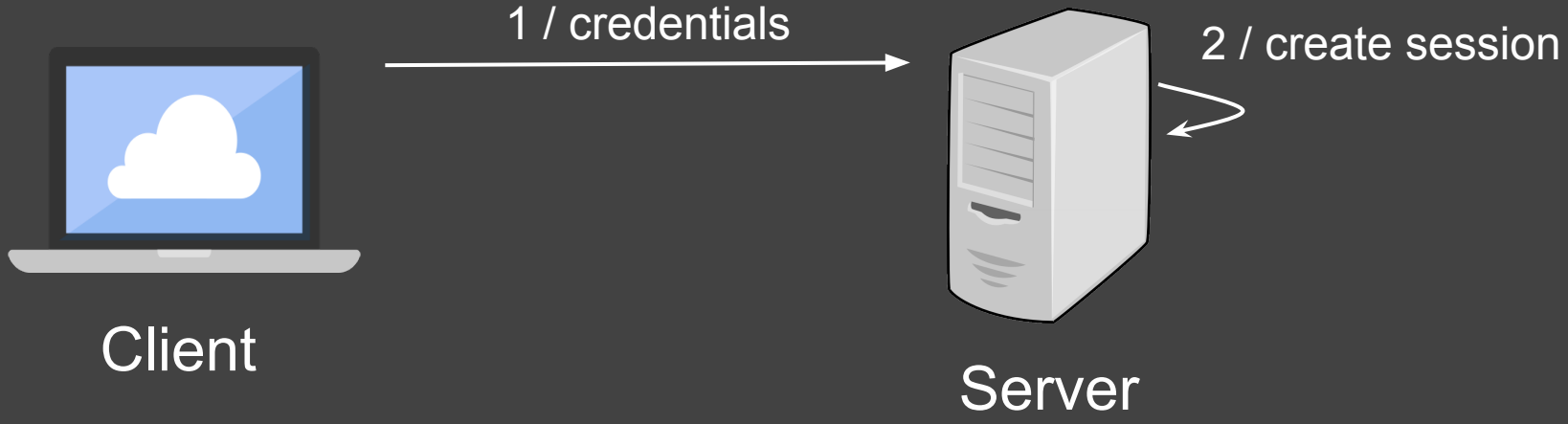
Client

Server

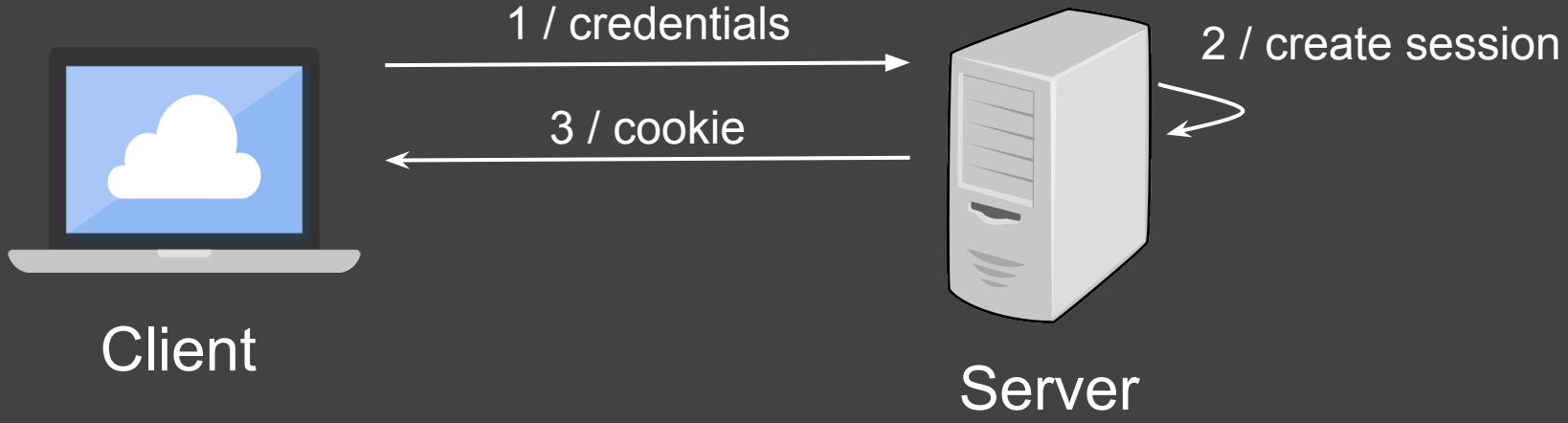
Cookie-based Auth



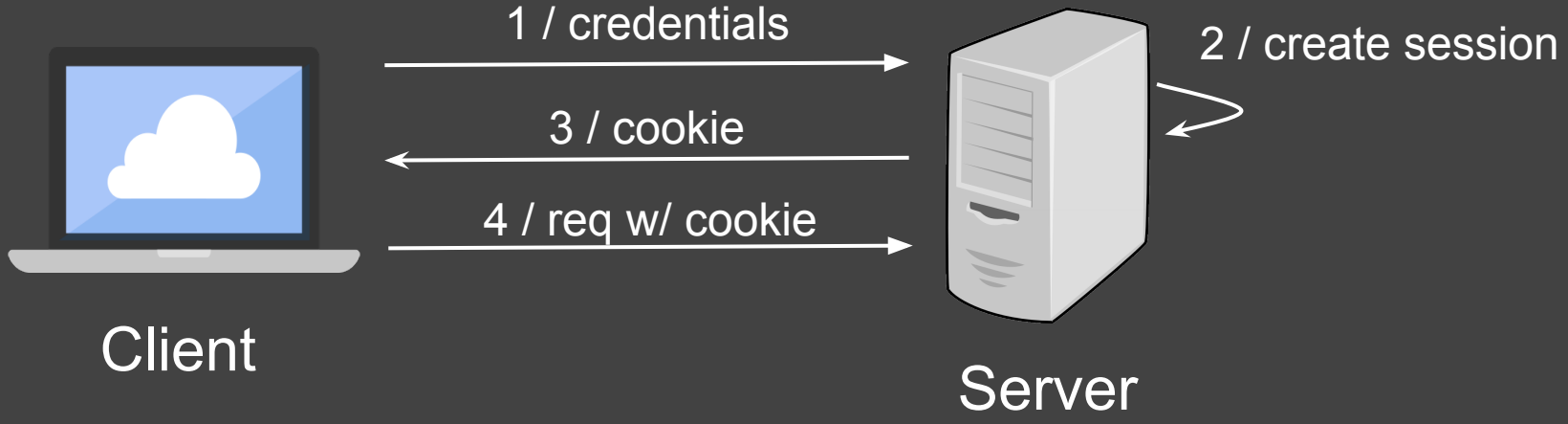
Cookie-based Auth



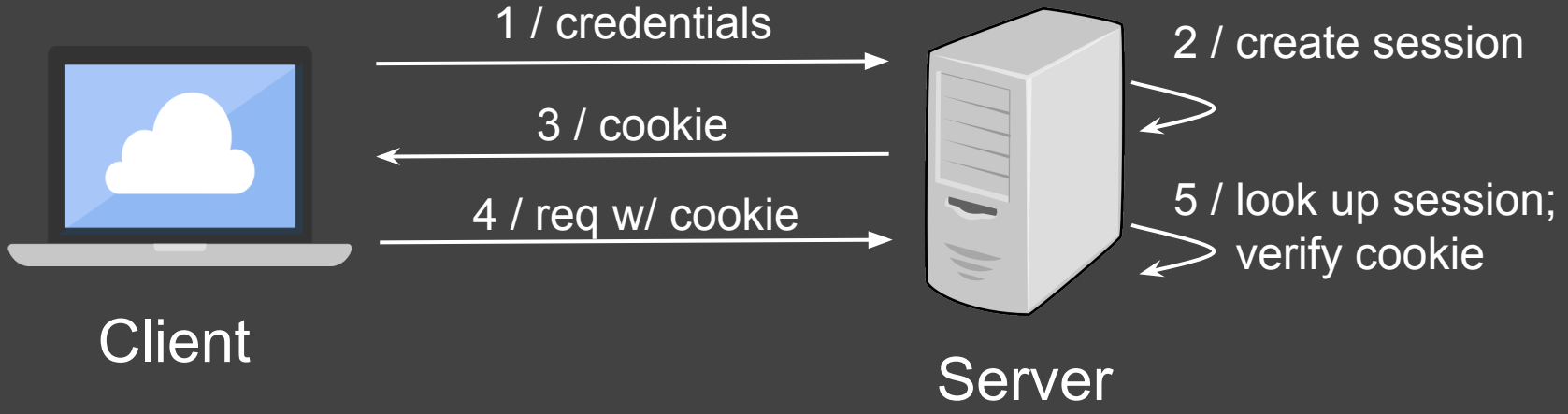
Cookie-based Auth



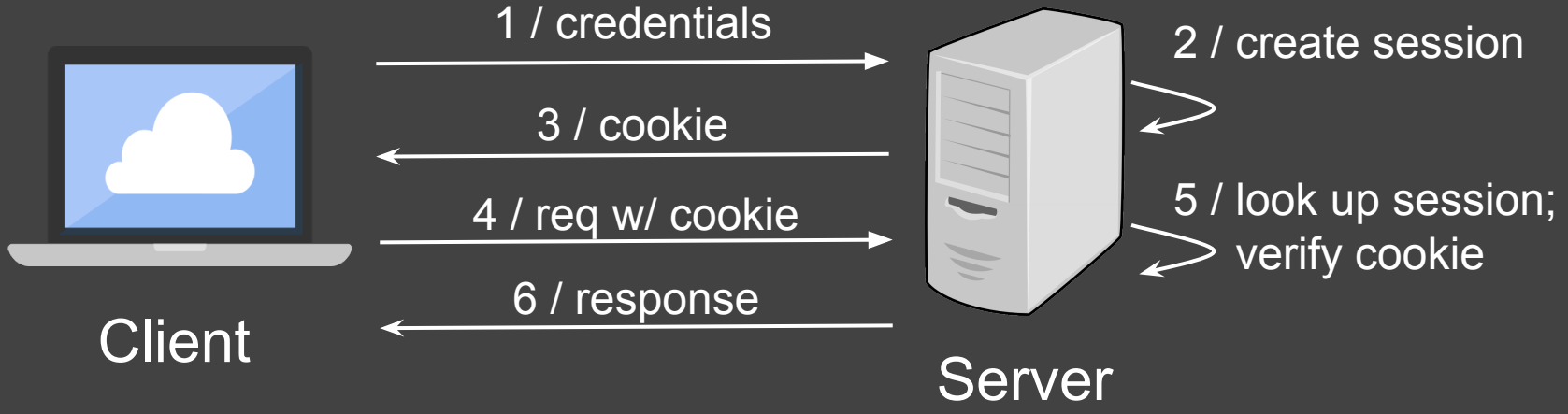
Cookie-based Auth



Cookie-based Auth



Cookie-based Auth



Cookie-based Auth Flaws

Cookie-based Auth Flaws

Requires stateful servers

Cookie-based Auth Flaws

Requires stateful servers

Requires server lookup

Cookie-based Auth Flaws

Requires stateful servers

Requires server lookup

Limited to one domain

Cookie-based Auth Flaws

Requires stateful servers

Requires server lookup

Limited to one domain

Susceptible to CSRF

Cookie-based Auth Flaws

Requires stateful servers

Requires server lookup

Limited to one domain

Susceptible to CSRF

Lack of mobile support

JSON Web Tokens (JWTs)

JSON Web Tokens (JWTs)

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1

NiJ9.eyJ2IjowLCJkIjp7InVpZCI6ImFu

b255bW91czotSnJRcXhESThMX2ZCL

VFndnVjbiIsInByb3ZpZGVyIjoiYW5vbn

ltb3VzIn0sImlhdCI6MTQzMzkxMjcyNX

0.L9G282A3G6qgAds3YWEFzdX4olcyT

z_S9q9ptJSH0T_nc

JSON Web Tokens (JWTs)

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1

NiJ9.eyJ2IjowLCJkIjp7InVpZCI6ImFu

b255bW91czotSnJRcXhESThMX2ZCL

VFndnVjbiIsInByb3ZpZGVyIjoiYW5vbn

ltb3VzIn0sImlhdCI6MTQzMzkxMjcyNX

0.L9G282A3G6qgAds3YWEFzdX4olcyT

z_S9q9ptJSH0T_nc

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

JSON Web Tokens (JWTs)

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1

NiJ9.eyJ2IjowLCJkIjp7InVpZCI6ImFu

b255bW91czotSnJRcXhESThMX2ZCL

VFndnVjbiIsInByb3ZpZGVyIjoiYW5vbn

ltb3VzIn0sImlhdCI6MTQzMzkxMjcyNX

0.L9G282A3G6qgAds3YWEFzdX4olcyT

z_S9q9ptJSH0T_nc


```
{
  "typ": "JWT",
  "alg": "HS256"
}
{
  "sub": "jwngr",
  "exp": 1433913414641
}
```

JSON Web Tokens (JWTs)

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1

NiJ9.eyJ2IjowLCJkIjp7InVpZCI6ImFu

b255bW91czotSnJRcXhESThMX2ZCL

VFndnVjbiIsInByb3ZpZGVyIjoiYW5vbn

ltb3VzIn0sImlhdCI6MTQzMzkxMjcyNX

0.L9G282A3G6qgAds3YWEFzdX4olcyT

z_S9q9ptJSH0T_nc

```
{
  "typ": "JWT",
  "alg": "HS256"
}
{
  "sub": "jwngr",
  "exp": 1433913414641
}
```

HMACSHA256(base64(header) + base64(payload) + secret);

JSON Web Tokens (JWTs)

JSON Web Tokens (JWTs)

Not encrypted

JSON Web Tokens (JWTs)

Not encrypted

Claims contains custom data

JSON Web Tokens (JWTs)

Not encrypted

Claims contains custom data

Stored in local storage

JSON Web Tokens (JWTs)

Not encrypted

Claims contains custom data

Stored in local storage

Server generates and validates JWTs

Generating JWTs

```
var jwt = new JWT("<SECRET>");
var token = jwt.generateToken({
  some: "arbitrary",
  data: "here"
});
```

Generating JWTs

```
var jwt = new JWT("<SECRET>");
var token = jwt.generateToken({
  some: "arbitrary",
  data: "here"
});
```

Generating JWTs

```
var jwt = new JWT("<SECRET>");
var token = jwt.generateToken({
  some: "arbitrary",
  data: "here"
});
```

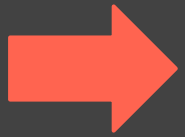
Generating JWTs

```
var jwt = new JWT("<SECRET>");

var token = jwt.generateToken({
  some: "arbitrary",
  data: "here"
});
```

Generating JWTs

```
var jwt = new JWT("<SECRET>");
var token = jwt.generateToken({
  some: "arbitrary",
  data: "here"
});
```



eyJ0eXAiOiJKV1Q
iLCJhbGciOiJIUzI
1NiJ9.
eyJ2IjowLCJkIjp7I
nVpZCI6ImFub25
5bW91czotSnJRc
XhESThMX2ZCLVF
ndnVjbiIsInByb3Z
pZGVyIjoiYW5vbn.

Validating JWTs

```
var jwt = new JWT("<SECRET>");

var data = jwt.decodeToken(token).claims;
```

Validating JWTs

```
var jwt = new JWT("<SECRET>");

var data = jwt.decodeToken(token).claims;
```

Validating JWTs

```
var jwt = new JWT("<SECRET>");

var data = jwt.decodeToken(token).claims;
```

Validating JWTs

```
var jwt = new JWT("<SECRET>");

var data = jwt.decodeToken(token).claims;
```

Validating JWTs

```
var jwt = new JWT("<SECRET>");

var data = jwt.decodeToken(token).claims;
```

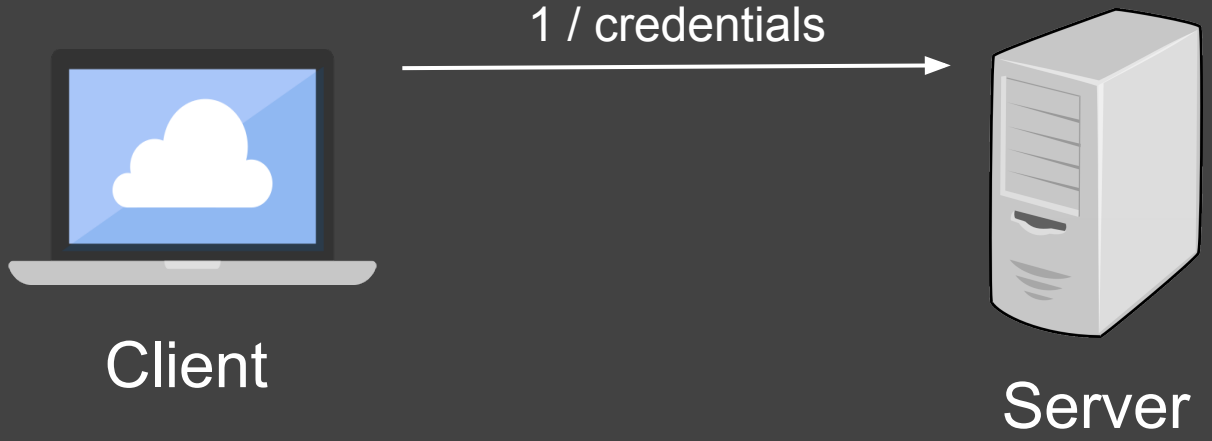
```
{ some: "arbitrary", data: "here" }
```

JWT-based Auth

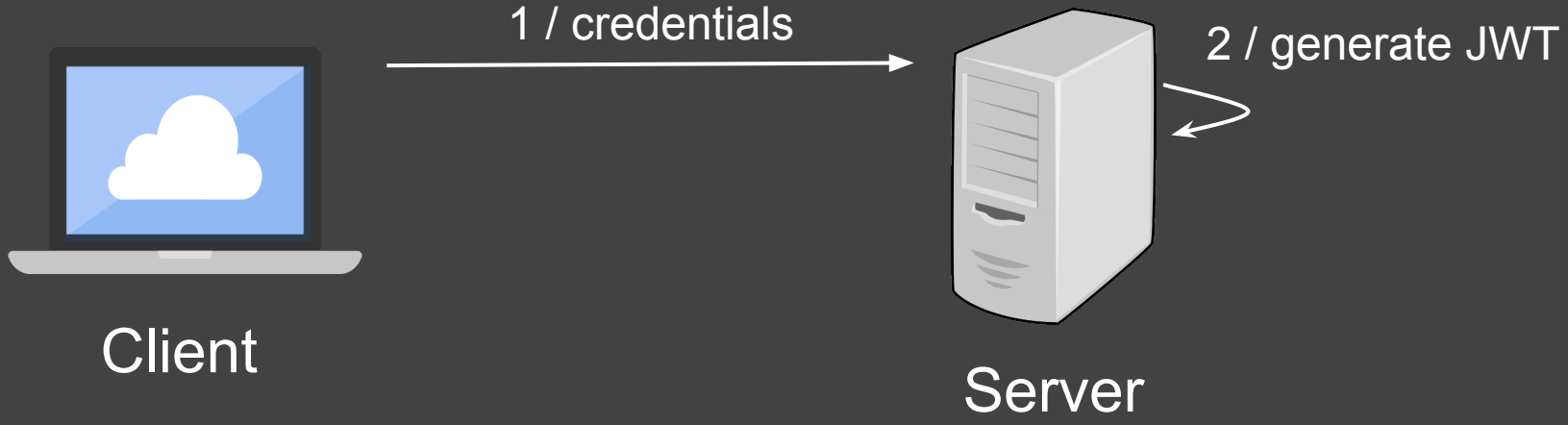
Client

Server

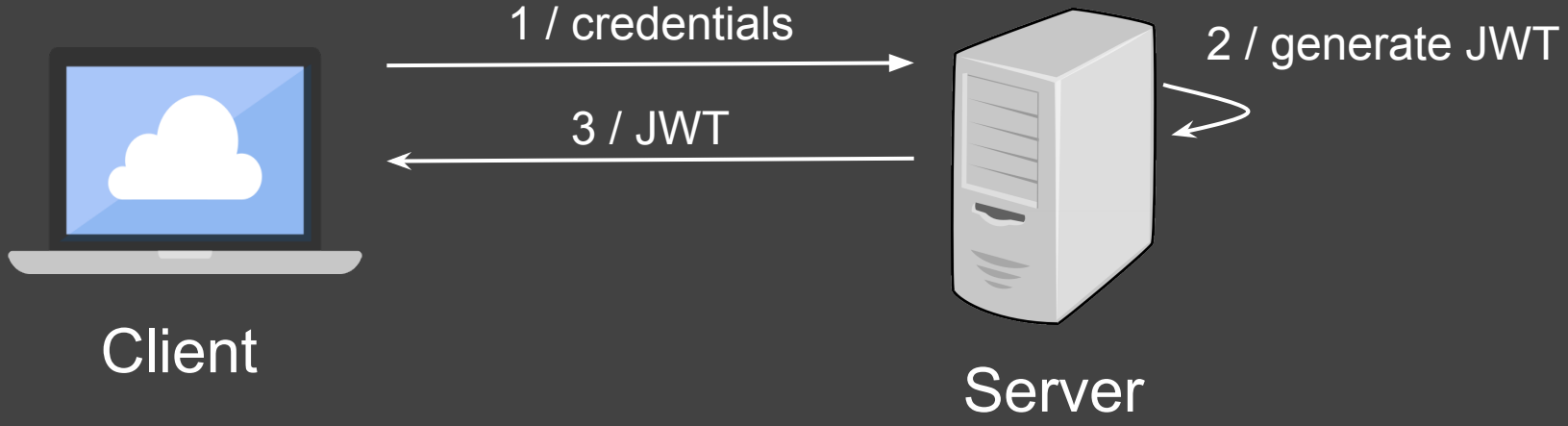
JWT-based Auth



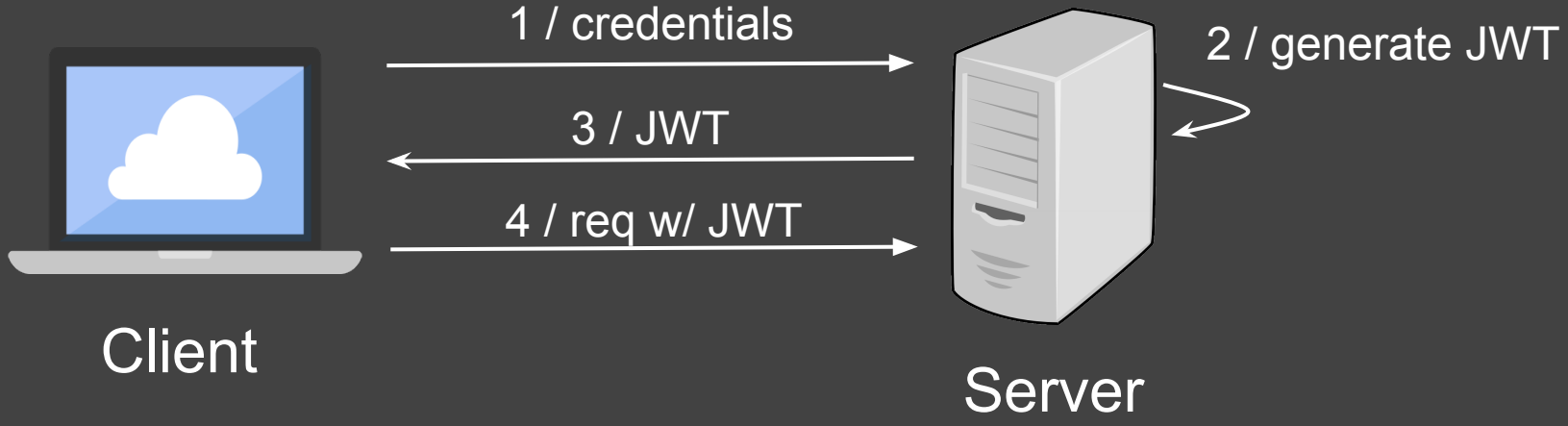
JWT-based Auth



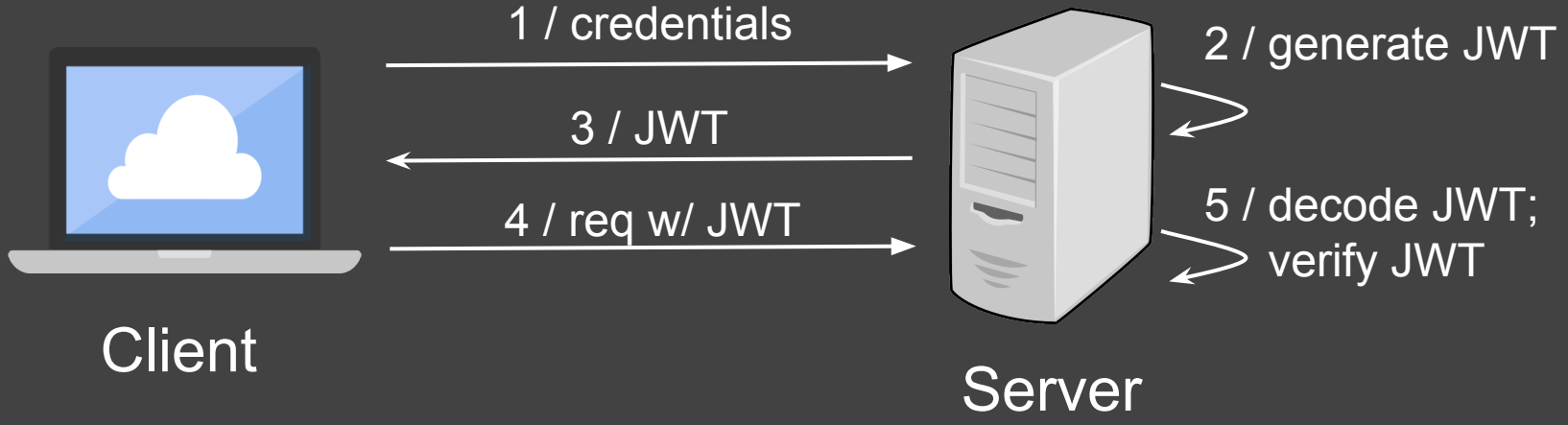
JWT-based Auth



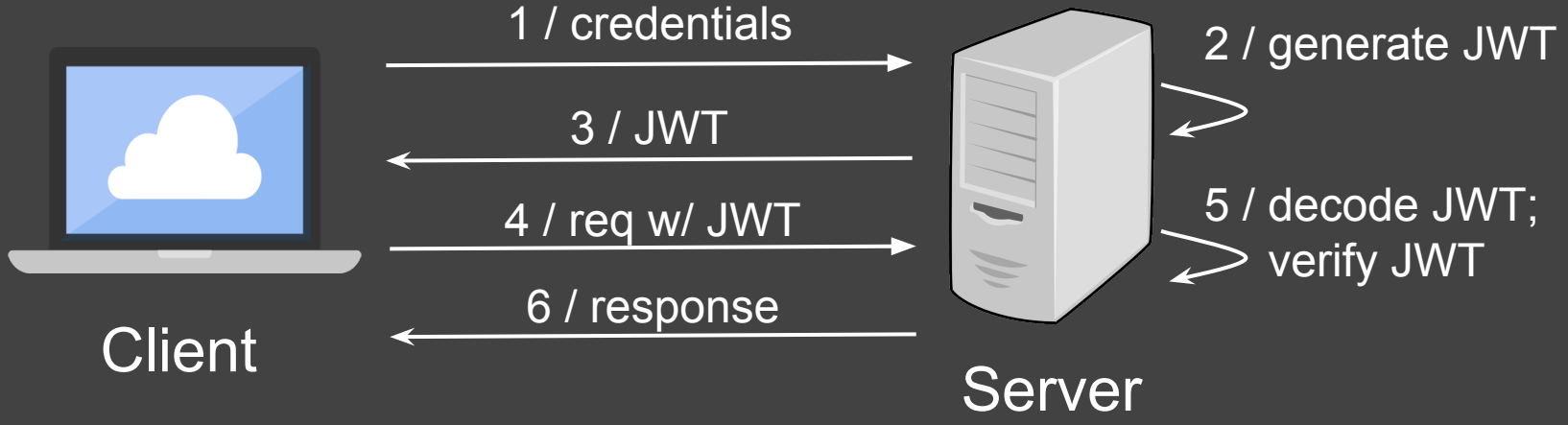
JWT-based Auth



JWT-based Auth



JWT-based Auth



Why JWTs?

Why JWTs?

State is kept on the client, not the server

Why JWTs?

State is kept on the client, not the server

Work across multiple domains

Why JWTs?

State is kept on the client, not the server

Work across multiple domains

Work just as well on mobile as they do on web

Why JWTs?

State is kept on the client, not the server

Work across multiple domains

Work just as well on mobile as they do on web

Generated / validated by anyone with the secret

Why JWTs?

State is kept on the client, not the server

Work across multiple domains

Work just as well on mobile as they do on web

Generated / validated by anyone with the secret

Susceptible to XSS (so be careful!)

JWTs FTW

JWTs FTW

So JWTs are great and all…

JWTs FTW?

So JWTs are great and all…

… but how do get the data to put in them?

Why Is Auth Hard?

Why Is Auth Hard?

Need server to generate JWTs

Why Is Auth Hard?

Need server to generate JWTs

Need database to store users

Why Is Auth Hard?

Need server to generate JWTs

Need database to store users

Storing passwords is tricky

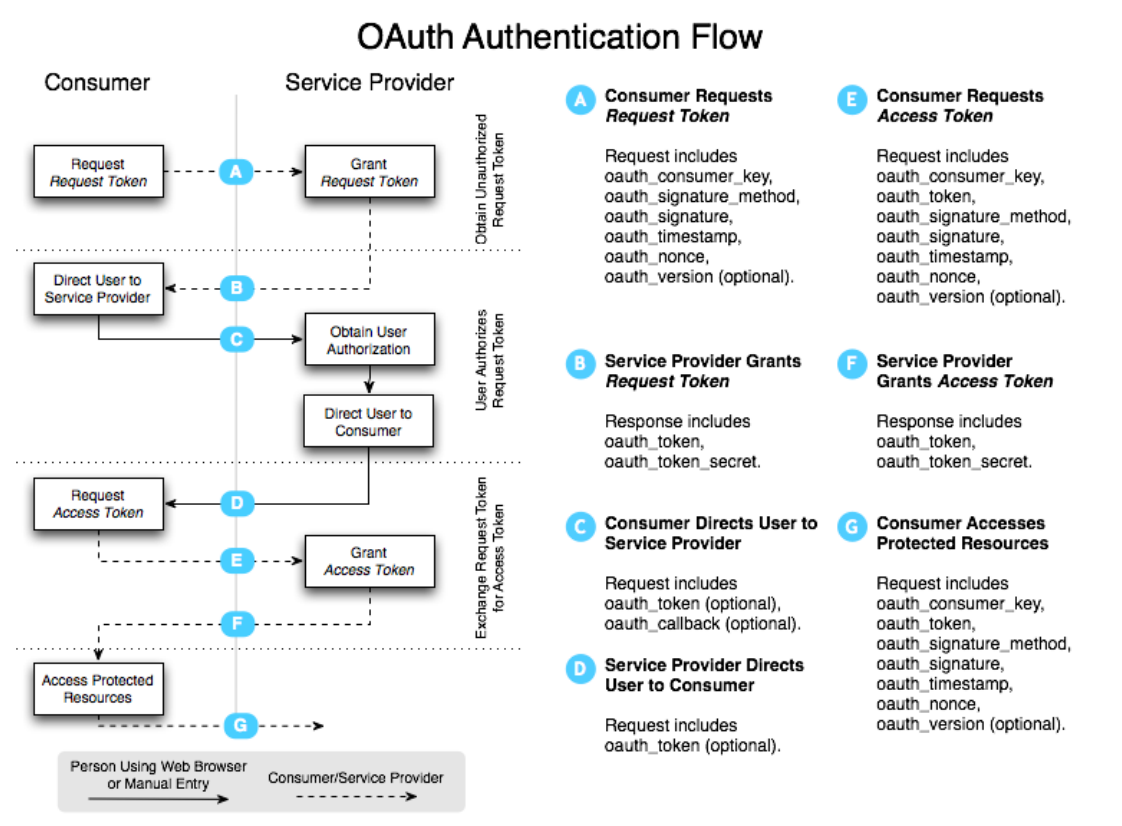
Why Is Auth Hard?

Need server to generate JWTs

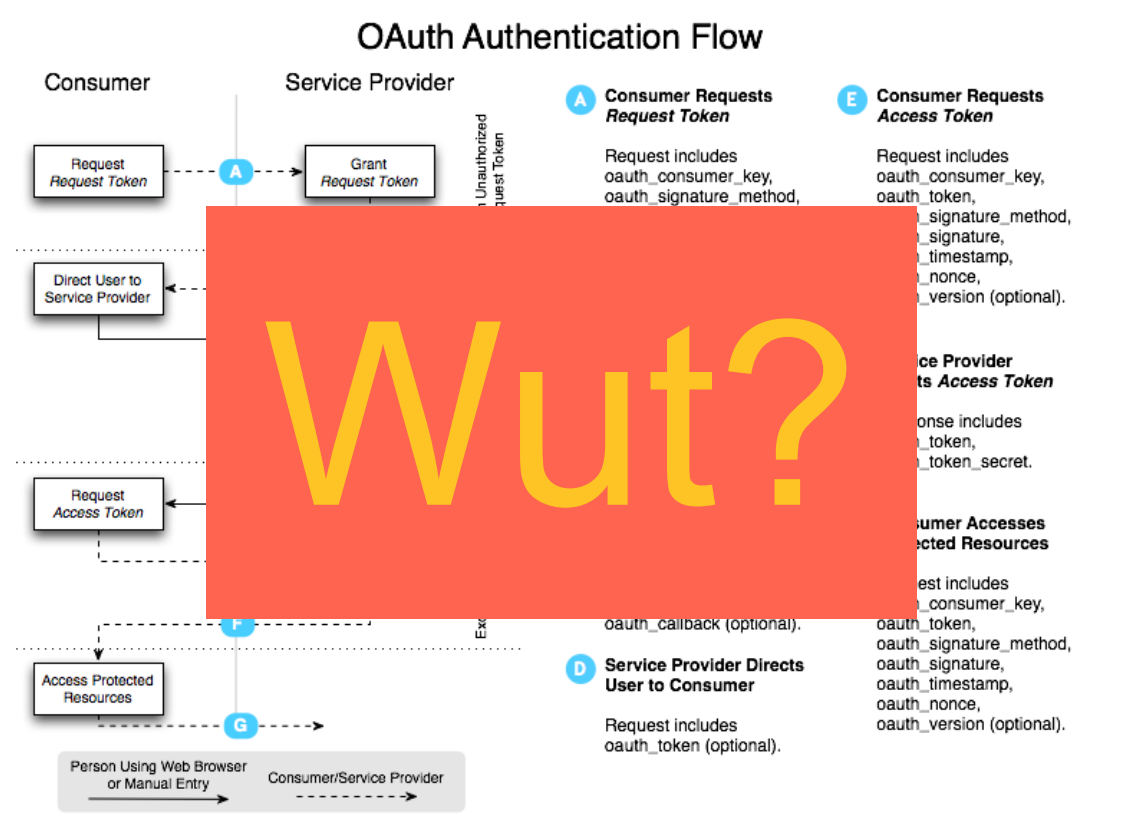
Need database to store users

Storing passwords is tricky

OAuth is a nightmare...



http://p2p.wrox.com/content/sites/default/files/users/17/image/figures%20ch6/531327%20f0602.png



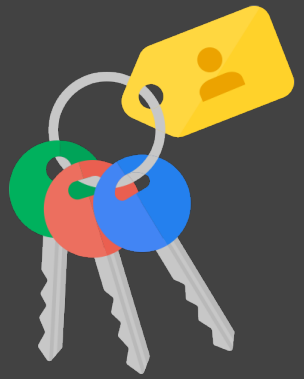
http://p2p.wrox.com/content/sites/default/files/users/17/image/figures%20ch6/531327%20f0602.png

Firebase

A powerful platform for
building extraordinary apps

Firebase Is Your Backend

Realtime Database

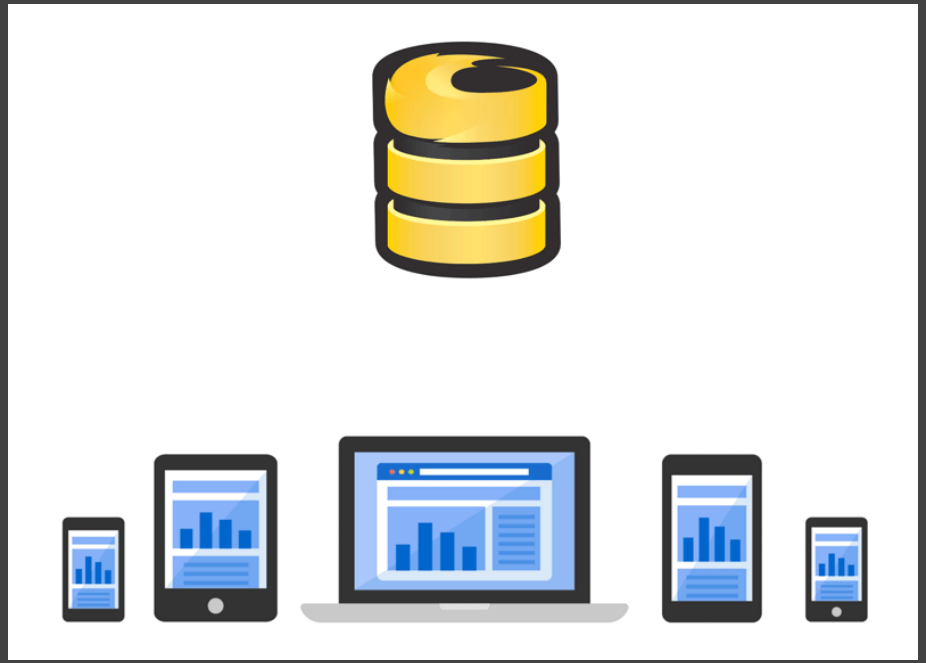


Authentication

Hosting

Cross-platform SDKs

Realtime Sync



Authentication

Authentication

JWT-based authentication

Authentication

JWT-based authentication

No server required

Authentication

JWT-based authentication

No server required

Many auth types (anonymous, email, OAuth, etc.)

Authentication

JWT-based authentication

No server required

Many auth types (anonymous, email, OAuth, etc.)

Custom auth if you want to make your own JWTs

Authentication

JWT-based authentication

No server required

Many auth types (anonymous, email, OAuth, etc.)

Custom auth if you want to make your own JWTs

Works offline

Authenticating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, authData) {
    // User authenticated
});
```

Authenticating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, authData) {
    // User authenticated
});
```

Authenticating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, authData) {
    // User authenticated
});
```

Authenticating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, authData) {
    // User authenticated
});
```

Authenticating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, authData) {
    // User authenticated
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Creating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, userData) {
    // User created
});
```

Creating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, userData) {
    // User created
});
```

Creating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, userData) {
    // User created
});
```

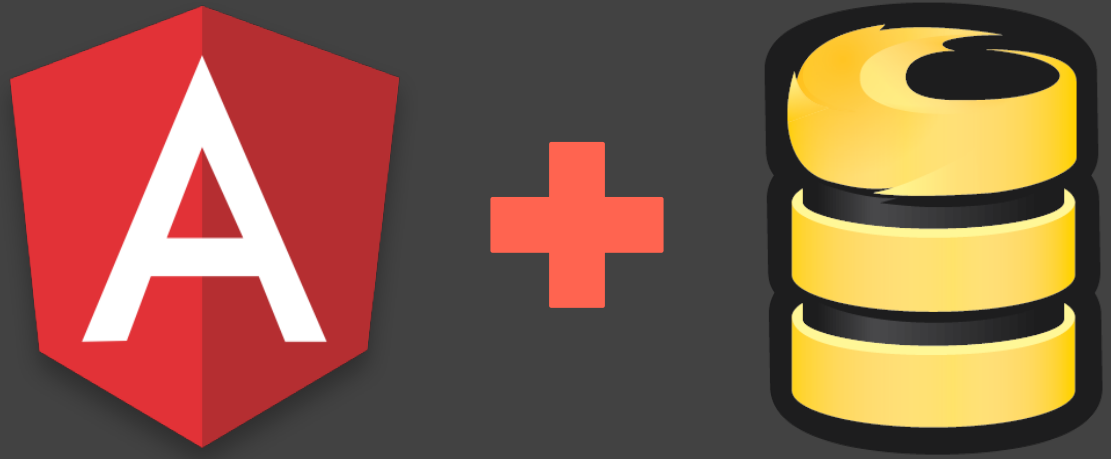
Creating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, userData) {
    // User created
});
```

Creating Users

```
var ref = new Firebase("https://ng-auth.firebaseio.com");
ref.createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}, function(error, userData) {
    // User created
});
```

Better Together



AngularFire

AngularFire provides a set of services to bind data between Firebase and Angular and to authenticate and manage users.

AngularFire Services

AngularFire Services

`$firebaseArray` - synchronized collections

AngularFire Services

`$firebaseArray` - synchronized collections

`$firebaseObject` - synchronized objects

AngularFire Services

`$firebaseArray` - synchronized collections

`$firebaseObject` - synchronized objects

`$firebaseAuth` - authentication, user management, routing

$firebaseAuth

```
var app = angular.module("app", ["firebase"]);
app.controller("Ctrl", function($scope, $firebaseAuth) {
    var ref = new Firebase("https://...");
    $scope.authObj = $firebaseAuth(ref);
    // ...
});
```

$firebaseAuth

```
var app = angular.module("app", ["firebase"]);
app.controller("Ctrl", function($scope, $firebaseAuth) {
    var ref = new Firebase("https://...");
    $scope.authObj = $firebaseAuth(ref);
    // ...
});
```

$firebaseAuth

```
var app = angular.module("app", ["firebase"]);
app.controller("Ctrl", function($scope, $firebaseAuth) {
    var ref = new Firebase("https://...");
    $scope.authObj = $firebaseAuth(ref);
    // ...
});
```

$firebaseAuth

```
var app = angular.module("app", ["firebase"]);
app.controller("Ctrl", function($scope, $firebaseAuth) {
    var ref = new Firebase("https://...");
    $scope.authObj = $firebaseAuth(ref);
    // ...
});
```

$firebaseAuth

```
var app = angular.module("app", ["firebase"]);
app.controller("Ctrl", function($scope, $firebaseAuth) {
    var ref = new Firebase("https://...");
    $scope.authObj = $firebaseAuth(ref);
    // ...
});
```

Authenticating Users

```
$scope.authObj.$authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(authData) {
    // User authenticated
}).catch(function(error) {
    // Authentication error
});
```

Authenticating Users

```
$scope.authObj.$authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(authData) {
    // User authenticated
}).catch(function(error) {
    // Authentication error
});
```

Authenticating Users

```
$scope.authObj.$authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(authData) {
    // User authenticated
}).catch(function(error) {
    // Authentication error
});
```

Authenticating Users

```
$scope.authObj.$authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(authData) {
    // User authenticated
}).catch(function(error) {
    // Authentication error
});
```

Authenticating Users

```
$scope.authObj.$authWithPassword({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(authData) {
    // User authenticated
}).catch(function(error) {
    // Authentication error
});
```

Detecting Auth State

```
$scope.authObj.$onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
$scope.authObj.$onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
$scope.authObj.$onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
$scope.authObj.$onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Detecting Auth State

```
$scope.authObj.$onAuth(function(authData) {
    if (authData) {
        // User logged in
    } else {
        // User logged out
    }
});
```

Creating Users

```
$scope.authObj.ref.$createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(userData) {
    // User created
}).catch(function(error) {
    // Error creating user
});
```

Creating Users

```
$scope.authObj.ref.$createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(userData) {
    // User created
}).catch(function(error) {
    // Error creating user
});
```

Creating Users

```
$scope.authObj.ref.$createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(userData) {
    // User created
}).catch(function(error) {
    // Error creating user
});
```

Creating Users

```
$scope.authObj.ref.$createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(userData) {
    // User created
}).catch(function(error) {
    // Error creating user
});
```

Creating Users

```
$scope.authObj.ref.$createUser({
    email: "foo@bar.com",
    password: "correcthorsebatterystaple"
}).then(function(userData) {
    // User created
}).catch(function(error) {
    // Error creating user
});
```

Route-based Security

Route-based Security

Helper promises for use in `resolve()`

Route-based Security

Helper promises for use in `resolve()`

`$waitForAuth()`

Route-based Security

Helper promises for use in `resolve()`

`$waitForAuth()`

`$requireAuth()`

Route-based Security

Helper promises for use in `resolve()`

`$waitForAuth()`

`$requireAuth()`

Works with `ngRoute` and `ui-router`

ngRoute

```
$routeProvider.when("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

ngRoute

```
$routeProvider.when("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

ui-router

```
$stateProvider.state("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

ui-router

```
$stateProvider.state("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

ui-router

```
$stateProvider.state("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

ui-router

```
$stateProvider.state("/account", {
  controller: "AccountCtrl",
  templateUrl: "views/account.html",
  resolve: {
    currentAuth: function(Auth) {
      return Auth.$waitForAuth();
    }
  }
});
```

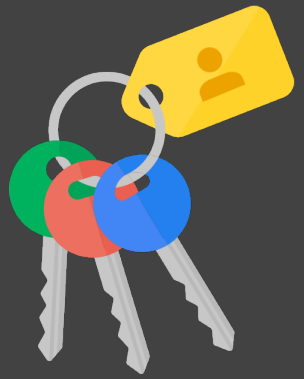
Live Coding

Try The Demo

https://ng-auth.firebaseapp.com

Firebase Is Your Backend

Realtime Database



Authentication

Hosting

Get Started Now!

firebase.com/docs

firebase.com/tutorial

Authentication in Angular

Jacob Wenger / @_jwngr
Firebase / @firebase

https://github.com/jwngr/ng-auth