DATAART

[IT Talks | IT NotStop | .Net meetup]

How to do SSO in ~10 lines of code

Speaker

Vyacheslav Mikhaylov (vmikhaylov@dataart.com)

Today’s topics

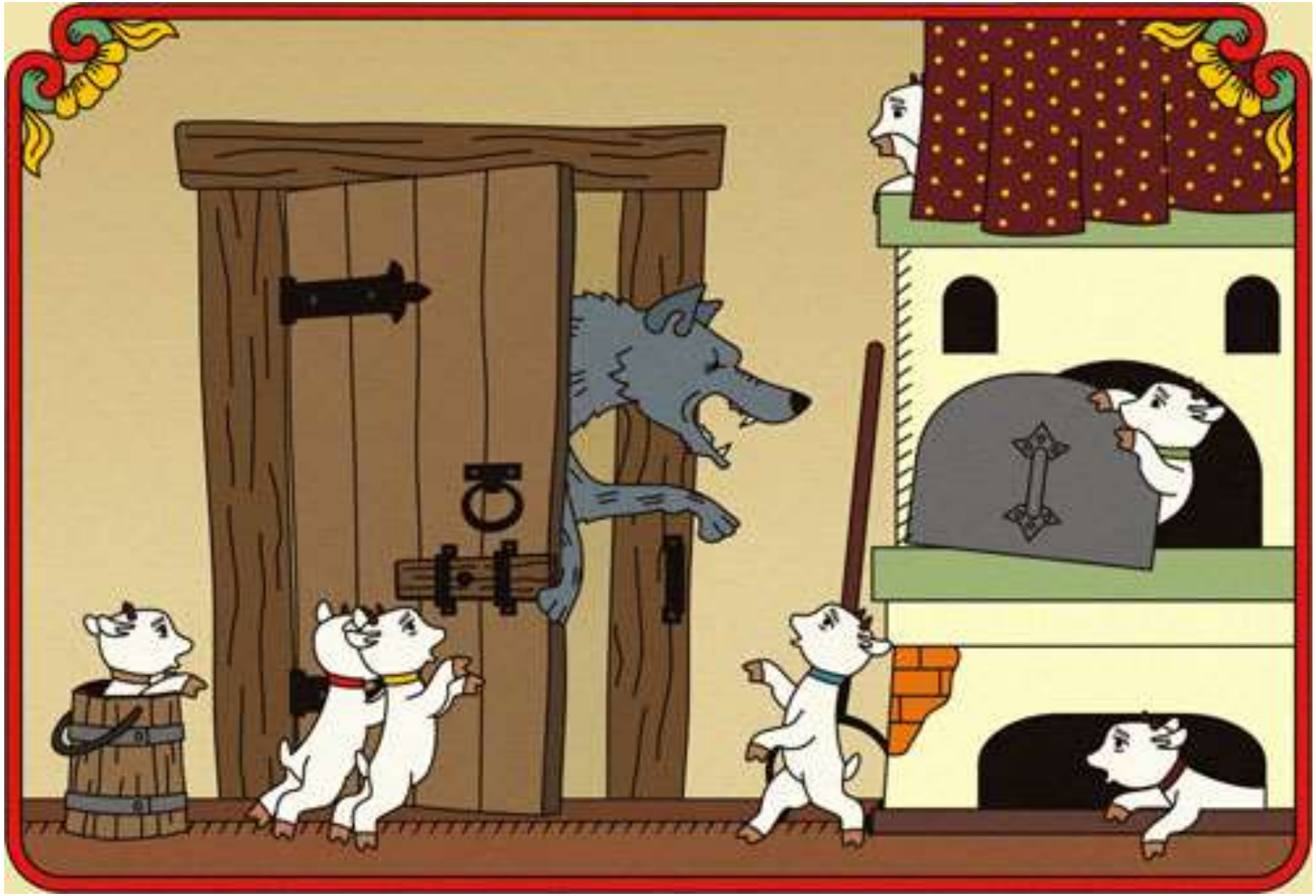
- Some theory and history
- OAuth2 и Open ID Connect
- IndentityServer/IdentityManager
  - Architecture
  - How to use

DATAART

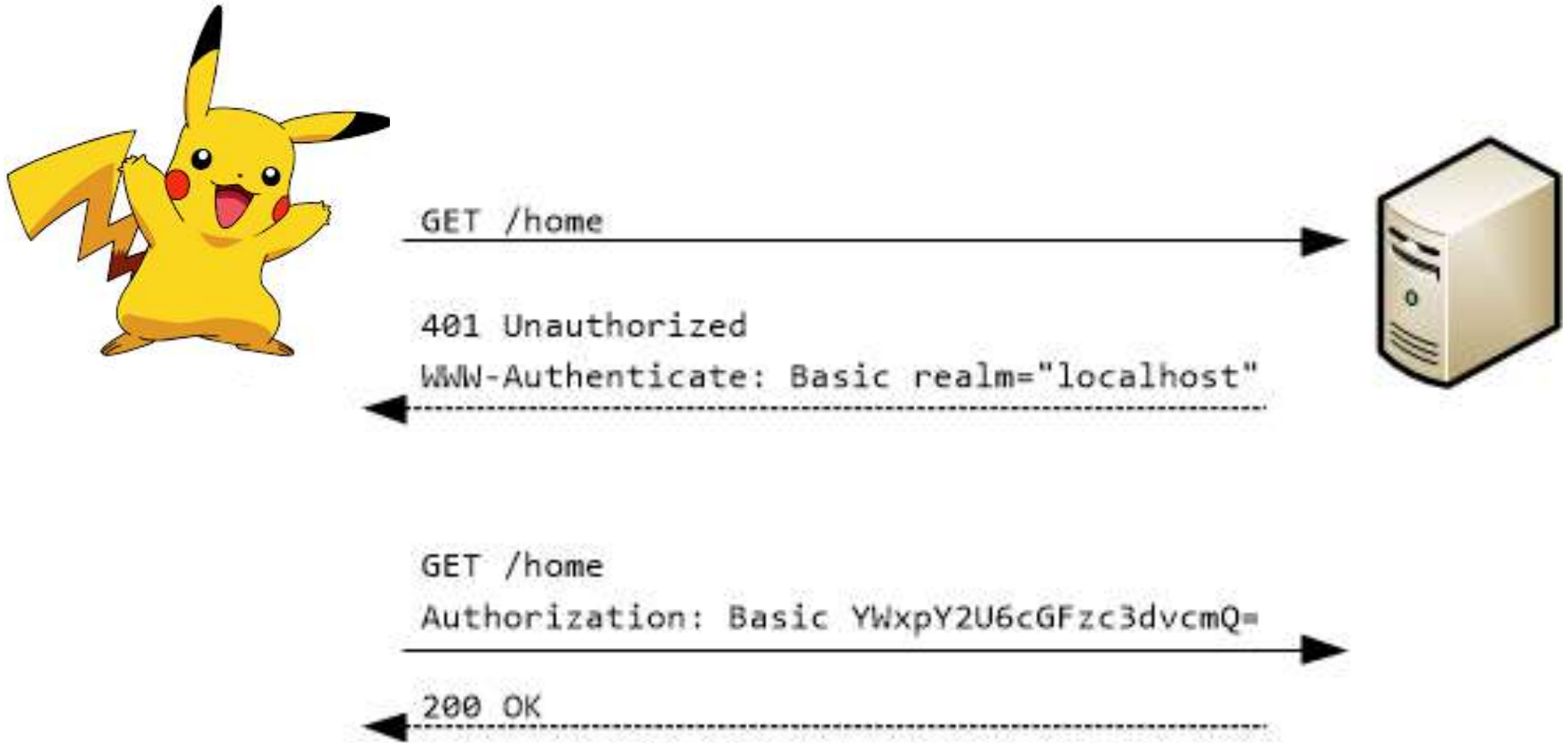
Terminology

- Identification - login
- Authentication – proof login is correct
- Authorization – authenticated user can access to some resource

DATAART



HTTP Basic Authentication



DATAART

HTTP Digest Authentication

DATAART

Forms Authentication

GET /login HTTP/1.1

HTTP/1.1 200 OK

POST /login HTTP/1.1
login=user&password=pass

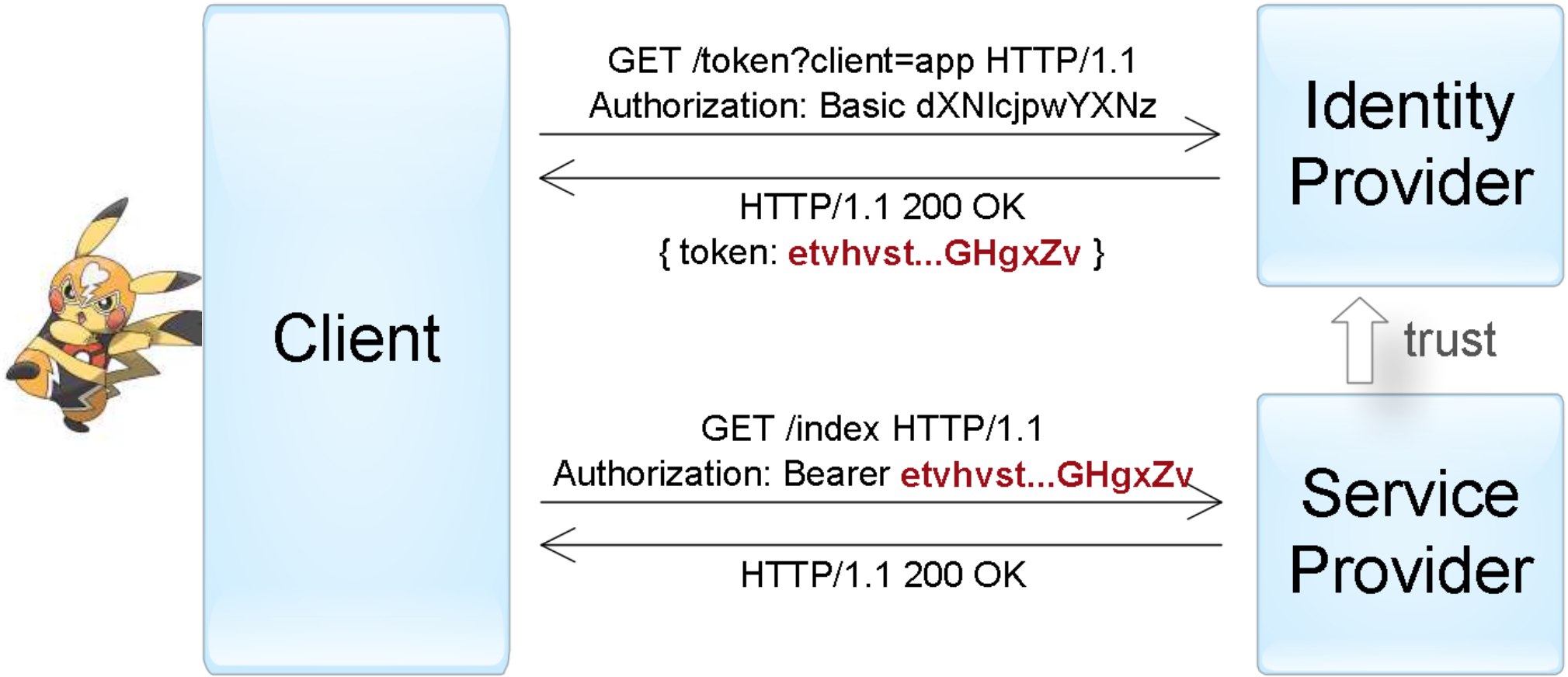
HTTP/1.1 302 Found
Set-Cookie: token=eAEKK9c3vJGQuhuf

GET /index HTTP/1.1
Cookie: token=eAEKK9c3vJGQuhuf

HTTP/1.1 200 OK

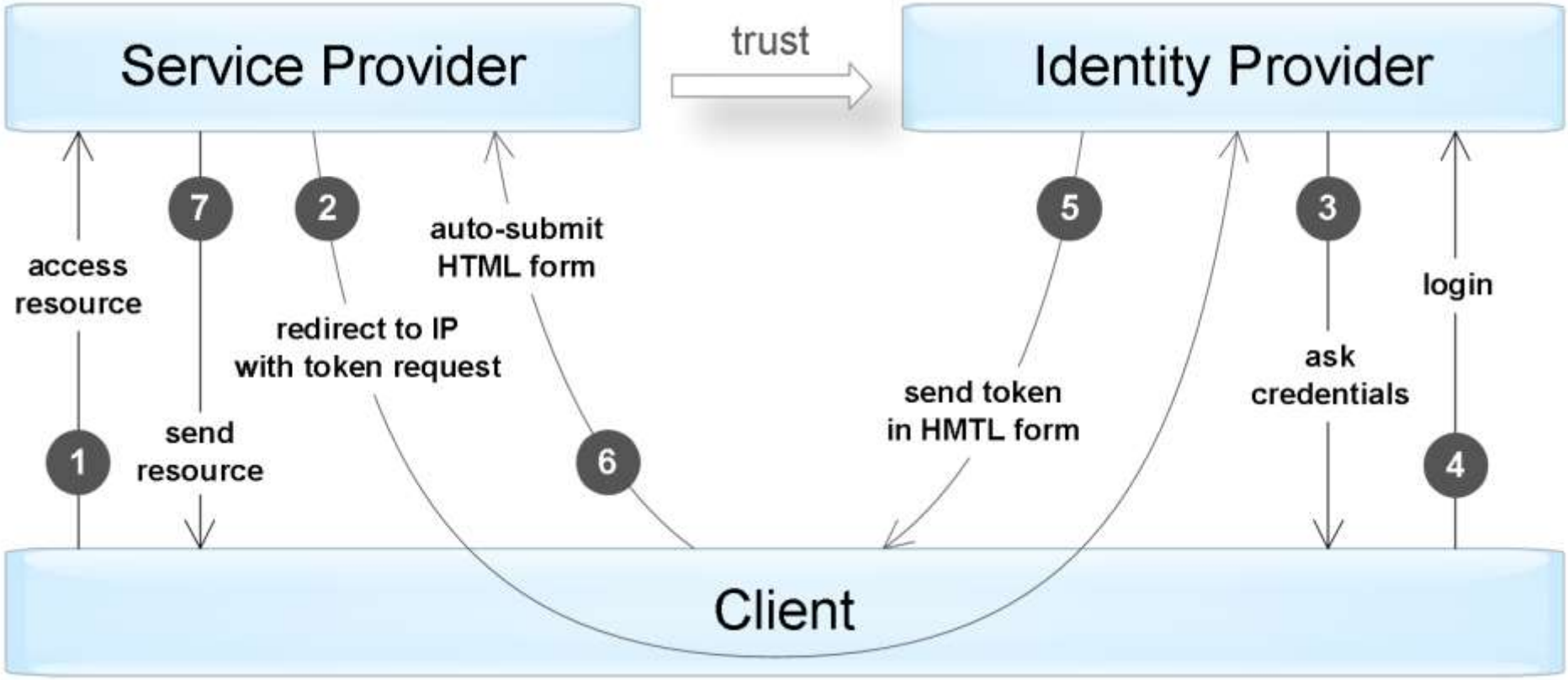
DATAART

Token Authentication



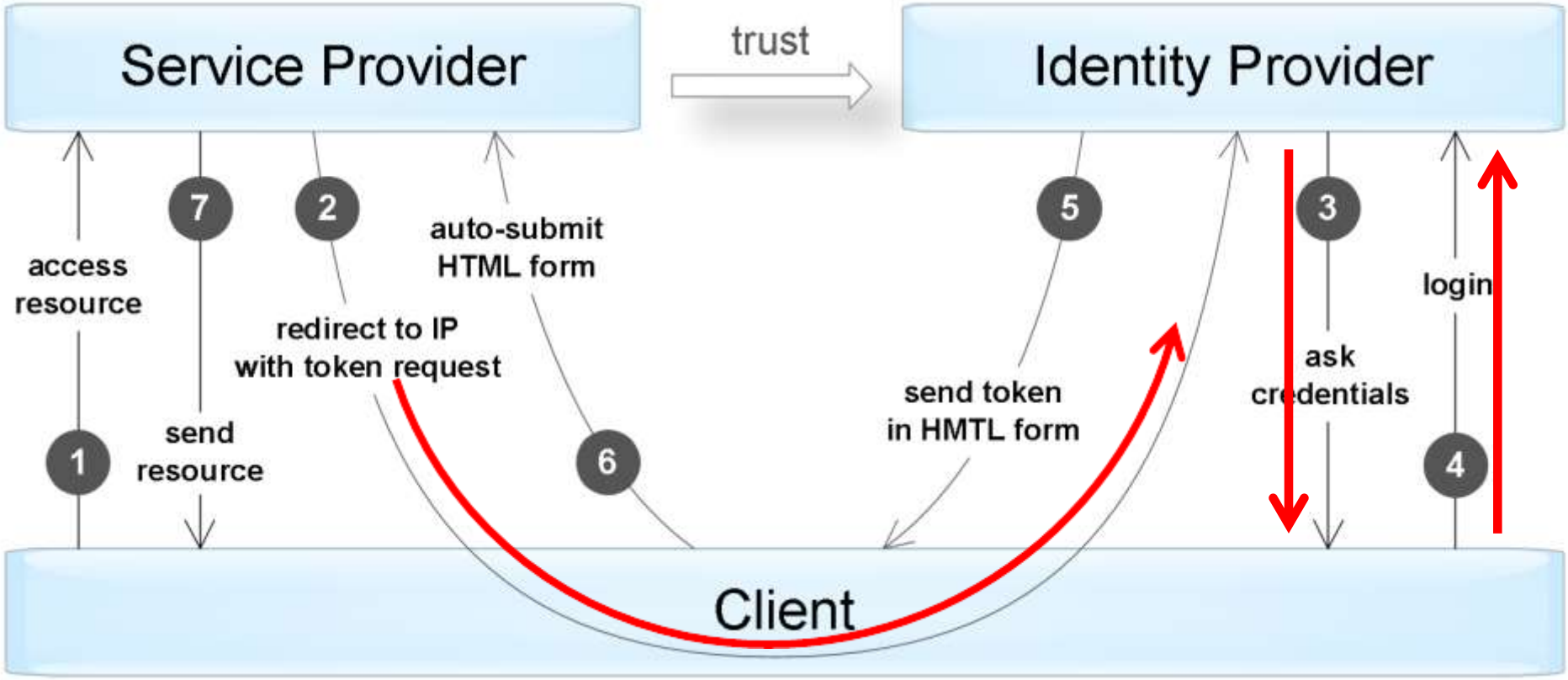
DATAART

Token Authentication



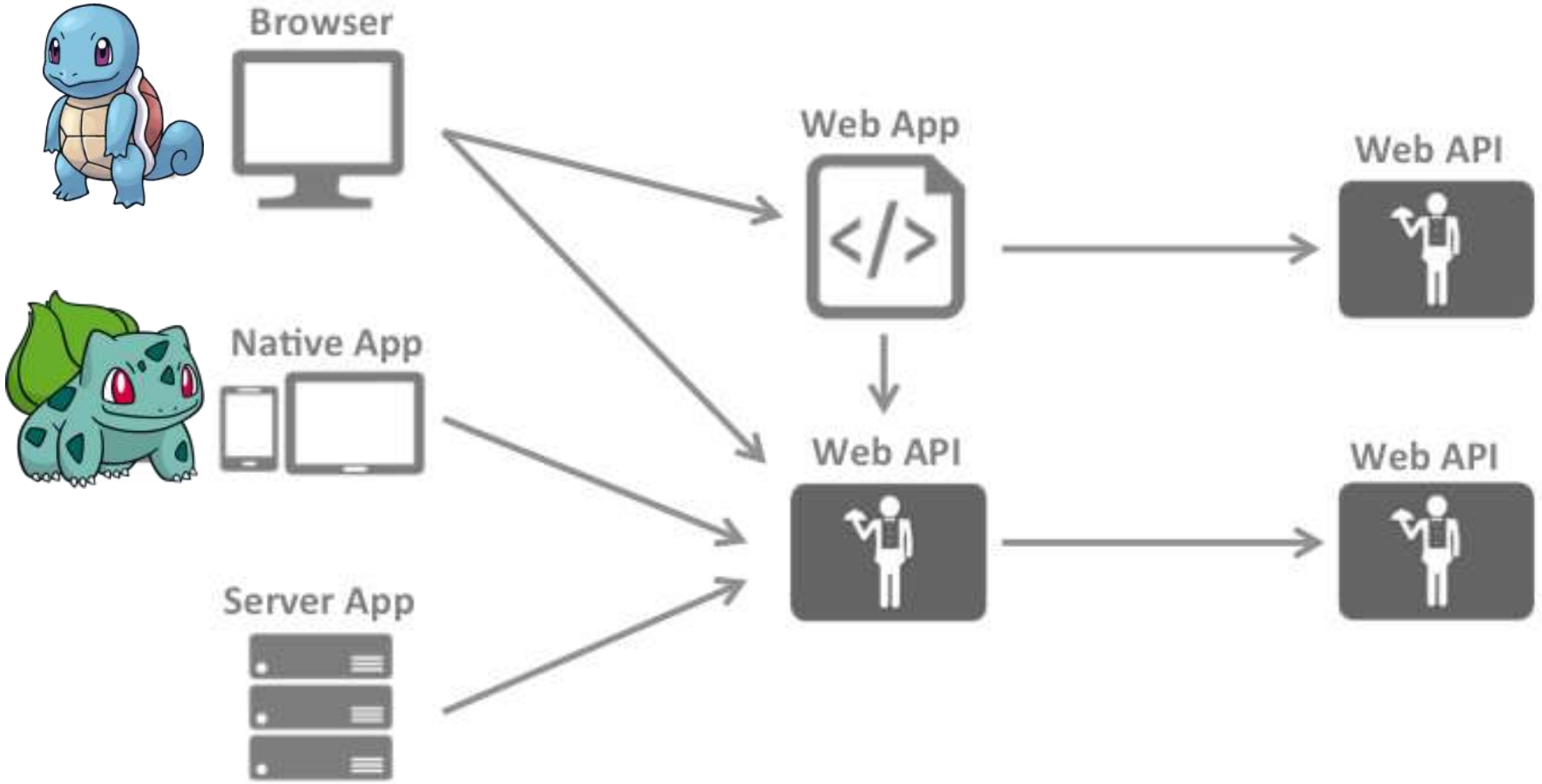
DATAART

Token Authentication



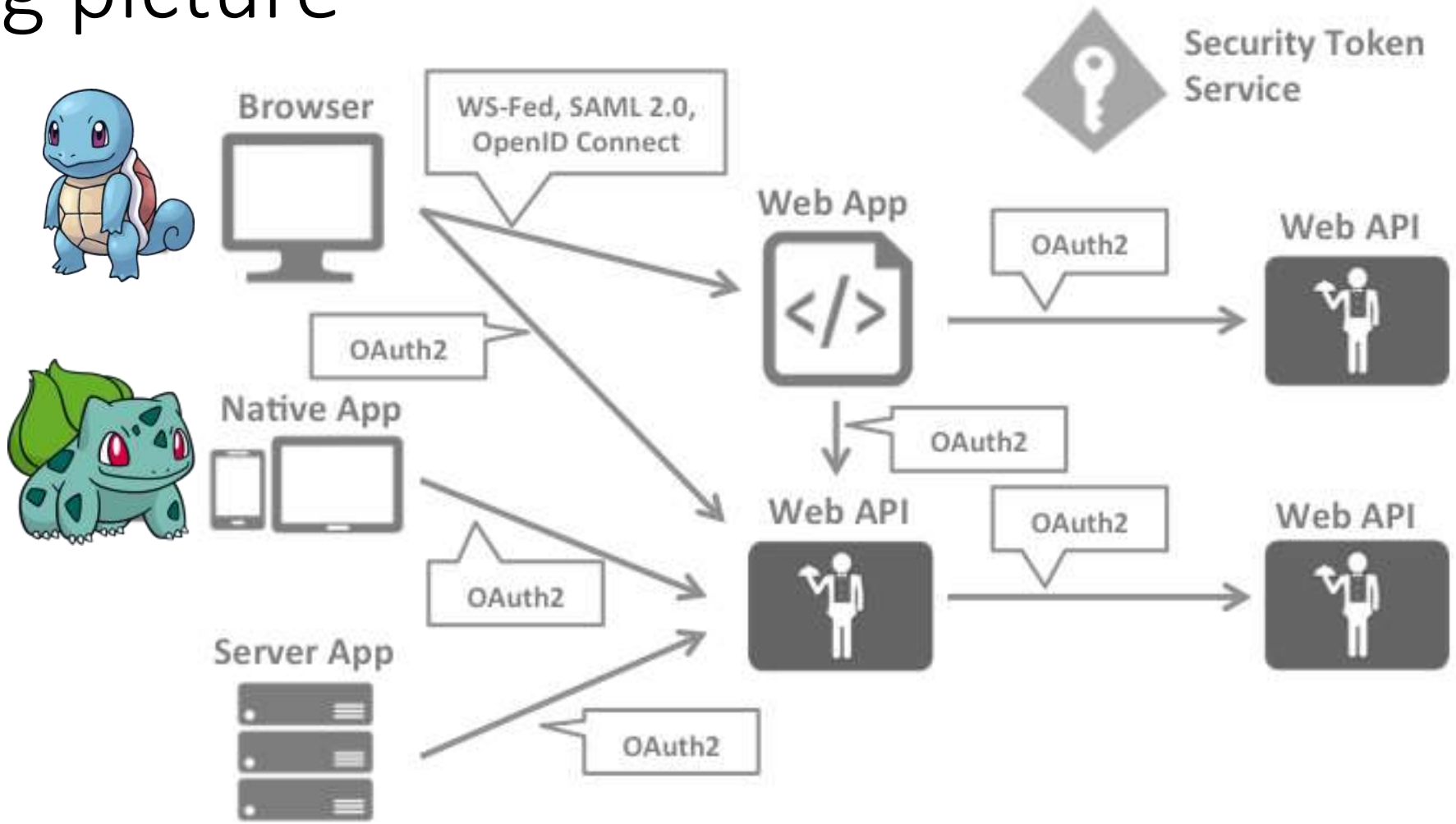
DATAART

Big picture



DATAART

Big picture

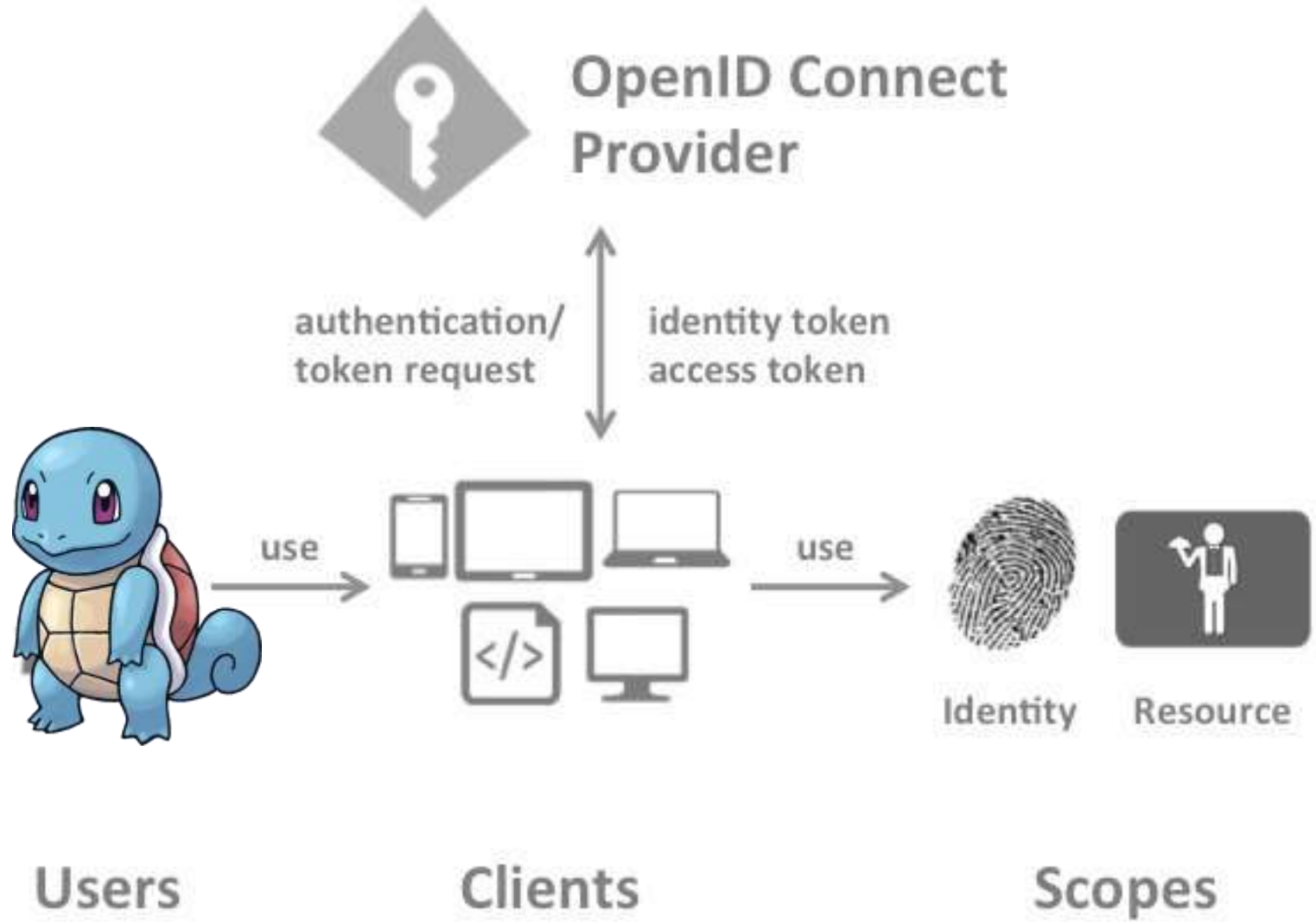


DATAART

Terminology

- OpenID Connect Provider (OP) - security token service, identity provider, authorization server, IP-STS and more.
- Client
- User - human
- Scope
  - Identity scopes – openid, profile, email
  - Resource scopes – various API
- Authentication/Token Request
- Identity Token
- Access Token

DATAART


OpenID Connect
Provider
authentication/
token request
identity token
access token
use
use
Identity
Resource
Users
Clients
Scopes

DATAART

Token structure (jwt.io)

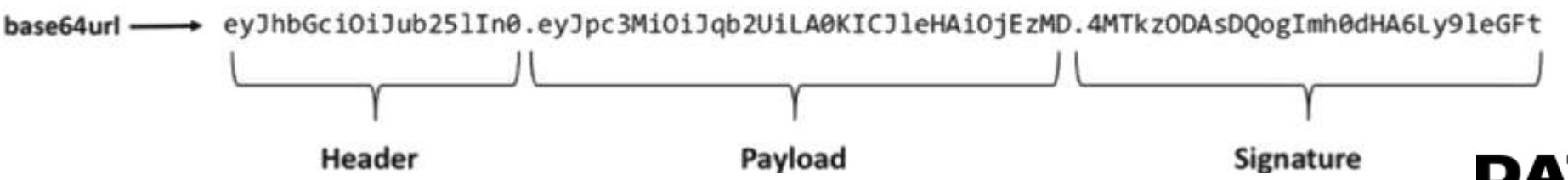
Header

```
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "mj399j…"
}
```

Payload

```
{
  "iss": "https://idsrv",
  "exp": 1340819380,
  "aud": "nativeapp",
  "nonce": "j1y…a23",
  "amr": [ "password", "sms" ],
  "auth_time": 12340819300

  "sub": "182jmm199"
}
```



Thinktecture Identity Server

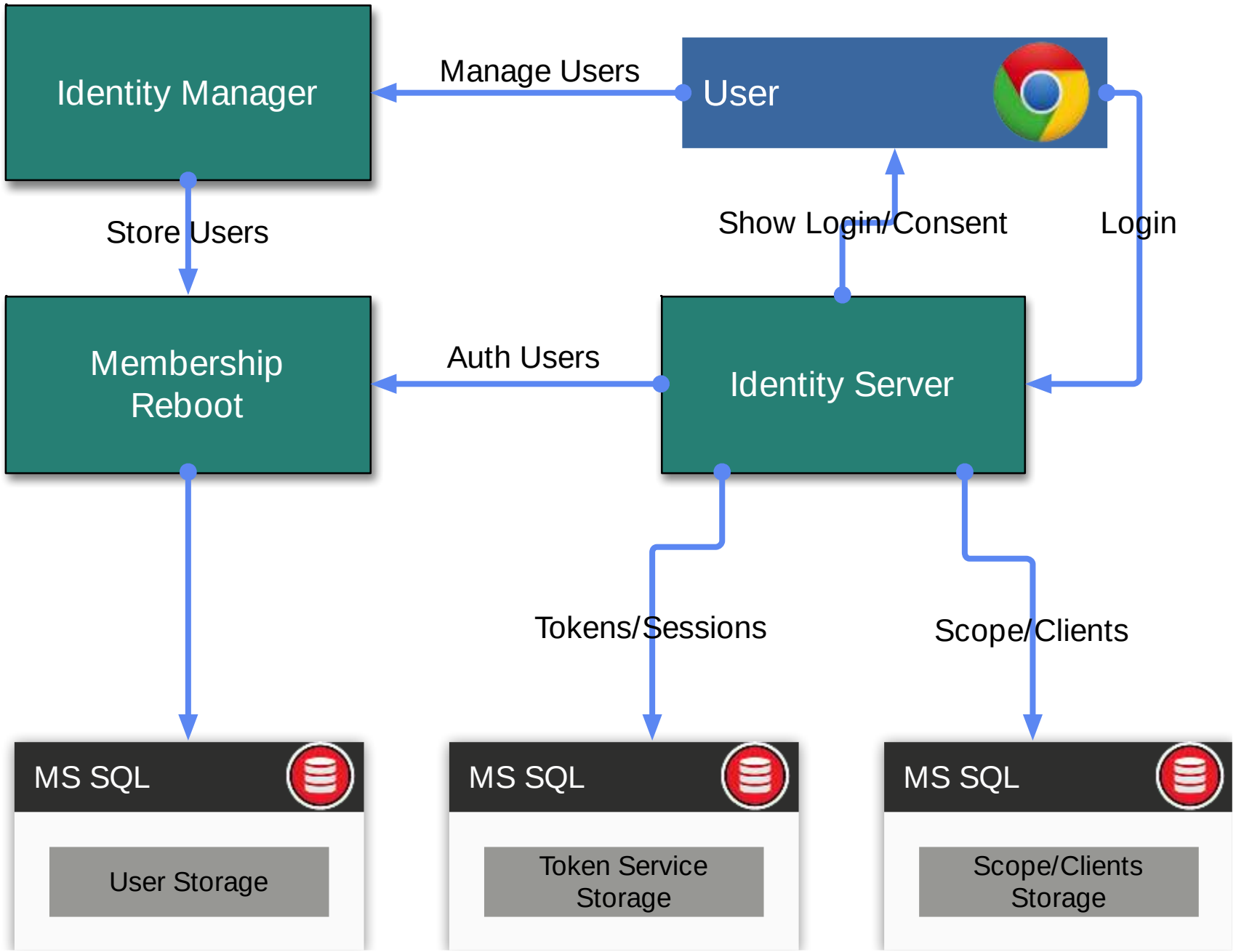
- OpenID Connect and OAuth2
- Авторы
  - **Dominick Baier**
  - **Brock Allen**
- Identity Server
- Identity Manager
- MembershipReboot

DATAART

Features

- Authentication as a Service
- Single Sign-on / Sign-out
- Access Control for APIs
- Federation
- Customization everywhere

DATA ART


Identity Manager
Manage Users
User
Store Users
Show Login/Consent
Login
Membership Reboot
Auth Users
Identity Server
Tokens/Sessions
Scope/Clients
MS SQL
User Storage
MS SQL
Token Service Storage
MS SQL
Scope/Clients Storage

Server

```
public void Configuration(IAppBuilder app)
{
    var factory = new IdentityServerServiceFactory();
    factory
        .UseInMemoryClients(Clients.Get())
        .UseInMemoryScopes(Scopes.Get())
        .UseInMemoryUsers(Users.Get());

    var options = new IdentityServerOptions
    {
        SiteName = "IdentityServer3",

        SigningCertificate = Certificate.Get(),

        Factory = factory,
    };

    app.UseIdentityServer(options);
}
```

Server

```
public void Configuration(IAppBuilder app)
{
    var factory = new IdentityServerServiceFactory();
    factory
        .UseInMemoryClients(Clients.Get())
        .UseInMemoryScopes(Scopes.Get())
        .UseInMemoryUsers(Users.Get());

    var options = new IdentityServerOptions
    {
        SiteName = "IdentityServer3",

        SigningCertificate = Certificate.Get(),

        Factory = factory,
    };

    app.UseIdentityServer(options);
}
```

API

```
public void Configuration(IAppBuilder app)
{
    JwtSecurityTokenHandler.InboundClaimTypeMap.Clear();

    app.UseIdentityServerBearerTokenAuthentication(
        new IdentityServerBearerTokenAuthenticationOptions
        {
            Authority = "https://localhost:44323/core",
            RequiredScopes = new[] { "write" },
            ValidationMode = ValidationMode.Local,

            // credentials for the introspection endpoint
            ClientId = "write",
            ClientSecret = "secret"
        });

    app.UseWebApi(WebApiConfig.Register());
}
```

API

```
public void Configuration(IAppBuilder app)
{
    JwtSecurityTokenHandler.InboundClaimTypeMap.Clear();

    app.UseIdentityServerBearerTokenAuthentication(
        new IdentityServerBearerTokenAuthenticationOptions
        {
            Authority = "https://localhost:44323/core",
            RequiredScopes = new[] { "write" },
            ValidationMode = ValidationMode.Local,

            // credentials for the introspection endpoint
            ClientId = "write",
            ClientSecret = "secret"
        });

    app.UseWebApi(WebApiConfig.Register());
}
```

API

```
public void Configuration(IAppBuilder app)
{
    JwtSecurityTokenHandler.InboundClaimTypeMap.Clear();

    app.UseIdentityServerBearerTokenAuthentication(
        new IdentityServerBearerTokenAuthenticationOptions
        {
            Authority = "https://localhost:44323/core",
            RequiredScopes = new[] { "write" },
            ValidationMode = ValidationMode.Local,

            // credentials for the introspection endpoint
            ClientId = "write",
            ClientSecret = "secret"
        });

    app.UseWebApi(WebApiConfig.Register());
}
```

Client

```
public void Configuration(IAppBuilder app)
{
    JwtSecurityTokenHandler.InboundClaimTypeMap =
        new Dictionary<string, string>();

    app.UseCookieAuthentication(new CookieAuthenticationOptions
    {
        AuthenticationType = "Cookies"
    });

    app.UseOpenIdConnectAuthentication(
        new OpenIdConnectAuthenticationOptions
    {
        ClientId = "mvc.owin.implicit",
        Authority = "https://localhost:44323/core",
        RedirectUri = "https://localhost:44301/",
        ResponseType = "id_token",
        Scope = "openid email",
        SignInAsAuthenticationType = "Cookies",
    });
}
```

Client

```
public void Configuration(IAppBuilder app)
{
    JwtSecurityTokenHandler.InboundClaimTypeMap =
        new Dictionary<string, string>();

    app.UseCookieAuthentication(new CookieAuthenticationOptions
    {
        AuthenticationType = "Cookies"
    });

    app.UseOpenIdConnectAuthentication(
        new OpenIdConnectAuthenticationOptions
    {
        ClientId = "mvc.owin.implicit",
        Authority = "https://localhost:44323/core",
        RedirectUri = "https://localhost:44301/",
        ResponseType = "id_token",
        Scope = "openid email",
        SignInAsAuthenticationType = "Cookies",
    });
}
```

What is Identity Server

- Authorization/Authentication
- Token
- UserInfo
- Discovery
- Logout
- Token Revocation
- Token Introspection
- Access Token Validation
- Identity Token Validation

DATAART

Customization

- AuthenticationSessionValidator, AuthorizationCodeStore
- ClaimsProvider, ClientPermissionsService
- ClientStore, ConsentService, ConsentStore
- CorsPolicyService, CustomGrantValidators, CustomRequestValidator, CustomTokenResponseGenerator, CustomTokenValidator
- EventService, ExternalClaimsFilter, LocalizationService, RedirectUriValidator
- RefreshTokenService, RefreshTokenStore, ScopeStore
- SecretParsers, SecretValidators, SigningKeyService
- TokenHandleStore, TokenService, TokenSigningService, UserService
- ViewService

DATAART

Customization

- AuthenticationSessionValidator, AuthorizationCodeStore
- ClaimsProvider, ClientPermissionsService
- **ClientStore**, ConsentService, ConsentStore
- CorsPolicyService, **CustomGrantValidators**, CustomRequestValidator, CustomTokenResponseGenerator, CustomTokenValidator
- EventService, ExternalClaimsFilter, LocalizationService, RedirectUriValidator
- RefreshTokenService, RefreshTokenStore, **ScopeStore**
- SecretParsers, SecretValidators, SigningKeyService
- TokenHandleStore, TokenService, TokenSigningService, **UserService**
- **ViewService**

DATAART

Customization

- **ClientStore**
- **ScopeStore**
- **UserService**
- **ViewService**

DATA ART

What is Identity Manager

- Simple creating users, editing user information (passwords, email, claims, roles, etc.) and deleting users.
- Replacement for the ASP.NET WebSite Administration tool User Management

DATAART

What is MembershipReboot

- single- or multi-tenant account management
- flexible account storage design (relational/SQL or object/NoSql)
- claims-aware user identities
- support for account registration, email verification, password reset, etc.
- account lockout for multiple failed login attempts (password guessing)
- extensible templating for email notifications
- customizable username, password and email validation
- notification system for account activity and updates (e.g. for auditing)
- account linking with external identity providers (enterprise or social)
- supports certificate based authentication
- proper password storage (via PBKDF2)
  - configurable iterations
  - defaults to OWASP recommendations for iterations (e.g. 64K in year 2012)
- two factor authentication support via mobile phone SMS messages or client certificates

DATAART

Demo

DATAART

Источники

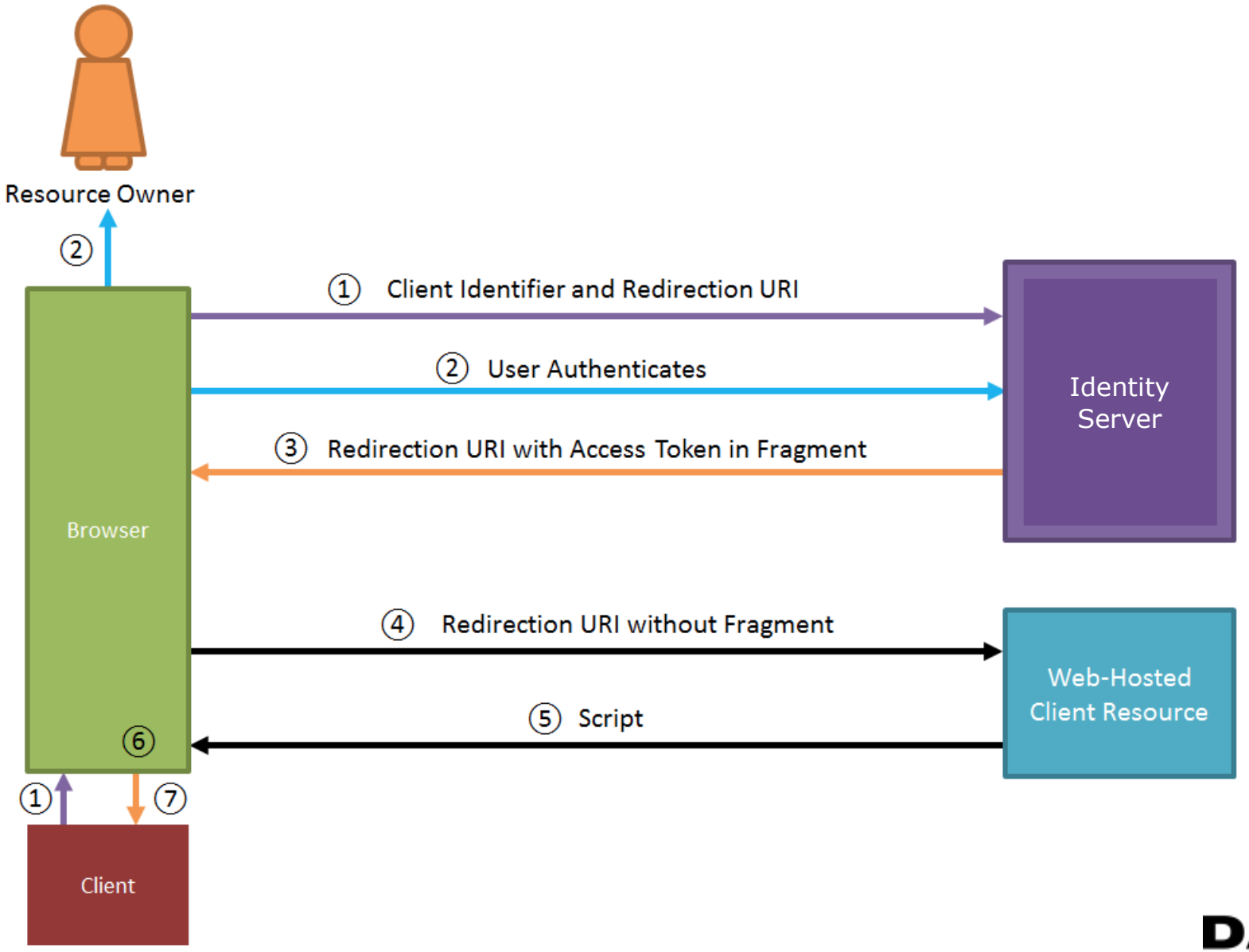
- https://habrahabr.ru/company/dataart/blog/262817/
- https://identityserver.github.io/Documentation/
- http://openid.net/connect/
- https://tools.ietf.org/html/rfc6749

DATAART

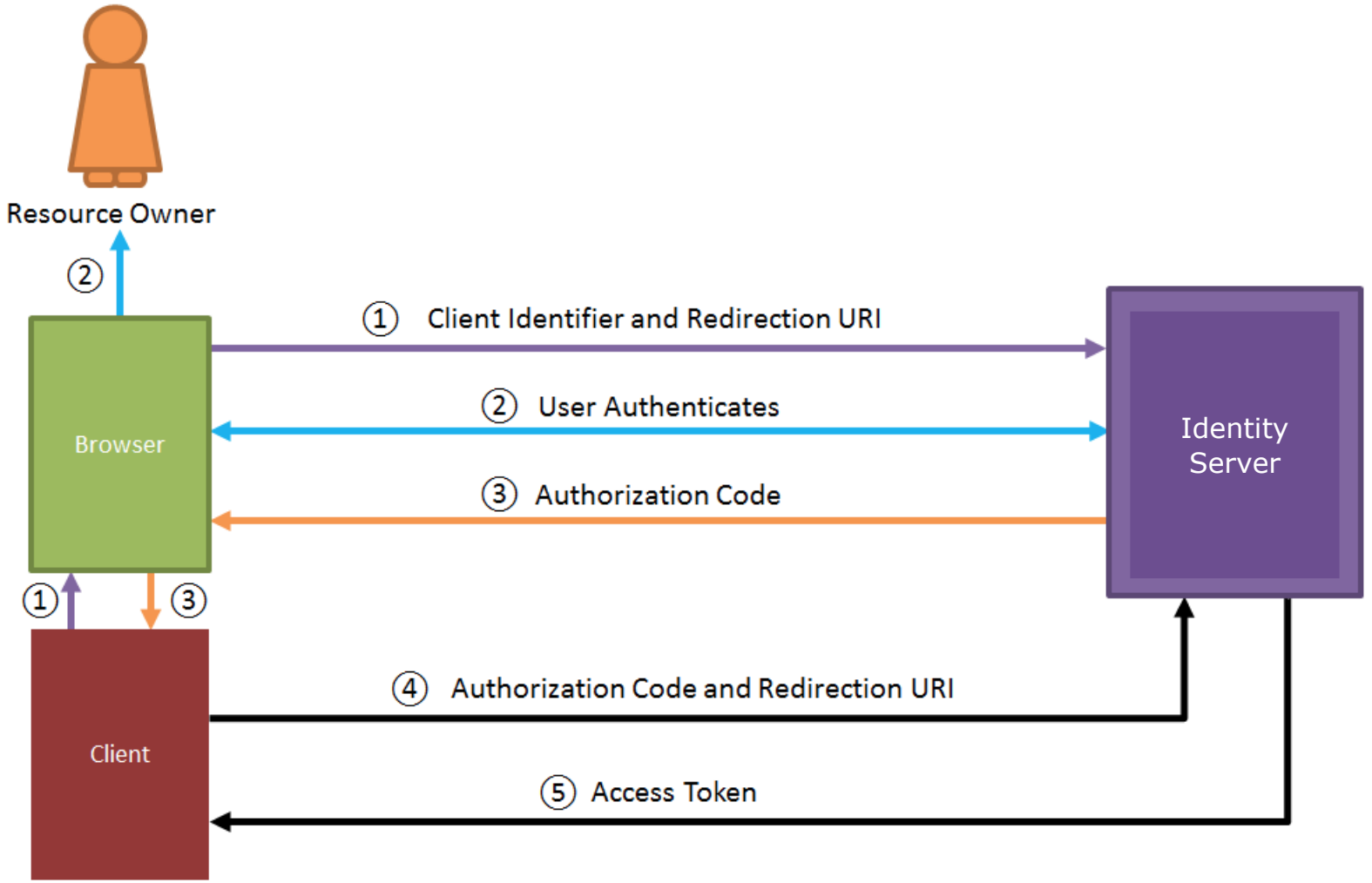
Thank you

To be continued…

DATAART


Resource Owner
2
1 Client Identifier and Redirection URI
2 User Authenticates
3 Redirection URI with Access Token in Fragment
Identity Server
Browser
4 Redirection URI without Fragment
Web-Hosted Client Resource
5 Script
6
1
7
Client

DATAART


Resource Owner
2
Browser
1
3
Client
1 Client Identifier and Redirection URI
2 User Authenticates
3 Authorization Code
4 Authorization Code and Redirection URI
5 Access Token
Identity Server

DATAART