

Securing ASP.NET Web APIs

Dominick Baier
<http://leastprivilege.com>
@leastprivilege

thinktecture
think mobile!

Dominick Baier

- **Security consultant at thinktecture**
- **Focus on**
 - security in distributed applications
 - identity management
 - access control
 - Windows/.NET security
 - mobile app security
- **Microsoft MVP for Developer Security**
- **ASP.NET Web API Advisor**
- **dominick.baier@thinktecture.com**
- **<http://leastprivilege.com>**



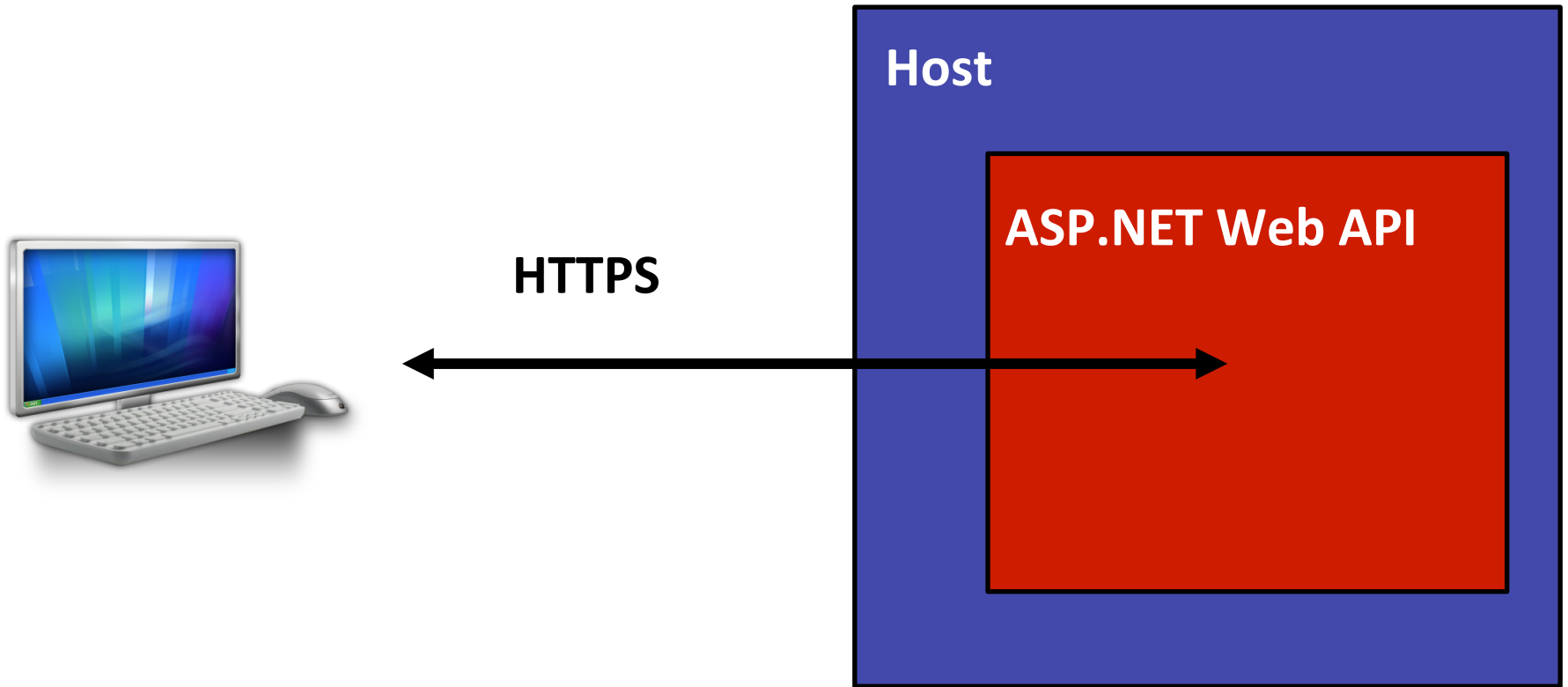
thinktecture
think mobile!

Agenda

- **HTTP security & SSL**
- **ASP.NET Web API v2 architecture**
- **Application scenarios**

- **(Token-based) authentication**
- **Authorization**
- **CSRF**
- **CORS**
- **OAuth2**

ASP.NET Web API: the big picture



Developers & SSL



how to handle SSL validation error



[SSL Certificate Validation Error in .Net « Akbar's Blog](#)

[blog.syedgakbar.com/.../ssl-certificate-validation-error-in-net/](#)

Jul 17, 2012 – This callback method is used to **validate** the certificate in an **SSL** conversation // Changed the **handle** to ignore the **SSL Certificate errors** in the ...

[SSL Function Return Codes](#)

[publib.boulder.ibm.com/infocenter/.../sssl2msg1000885.htm](#)

The environment or **SSL handle** specified on a System **SSL** function call is not ...
Certificate **validation error**. ... An error is detected while validating a certificate.

[Ignoring SSL validation in Java - Stack Overflow](#)

[stackoverflow.com/questions/.../ignoring-ssl-validation-in-java](#)

2 answers - 20 Nov 2012

Foreword: I DO know that skipping **SSL validation** is really ugly. In this ...
ClientStateReceivedServerHello.**handle**(Unknown Source) at ... catch (
KeyManagementException e) { log.**error** ("No **SSL** algorithm support: " + e.

[How to handle invalid SSL certificates with Apache - Stack Overflow](#)

[stackoverflow.com/.../how-to-handle-invalid-ssl-certificates-wi...](#)

9 answers - 1 Dec 2009

... at sun.security.validator.Validator.**validate**(Validator.java:235) at sun.security.**ssl**. ...
When I go to mms.nw.ru, I get a **error** screen in Chrome.

Security model for HTTP-based services

- **Simple model**
 - HTTP + content + SSL
- **Whenever authentication is required**
 - Status code of 401 indicates *unauthorized*
 - *WWW-Authenticate* response header indicates preferred authentication method



Status Code: 401 unauthorized
←
WWW-Authenticate: *Scheme* realm="myapp"



Authentication for HTTP-based services

- **Credentials transmitted (typically) via *Authorization* header**
 - e.g. Basic authentication, access tokens...
 - sometimes other means (query string, cookie...)

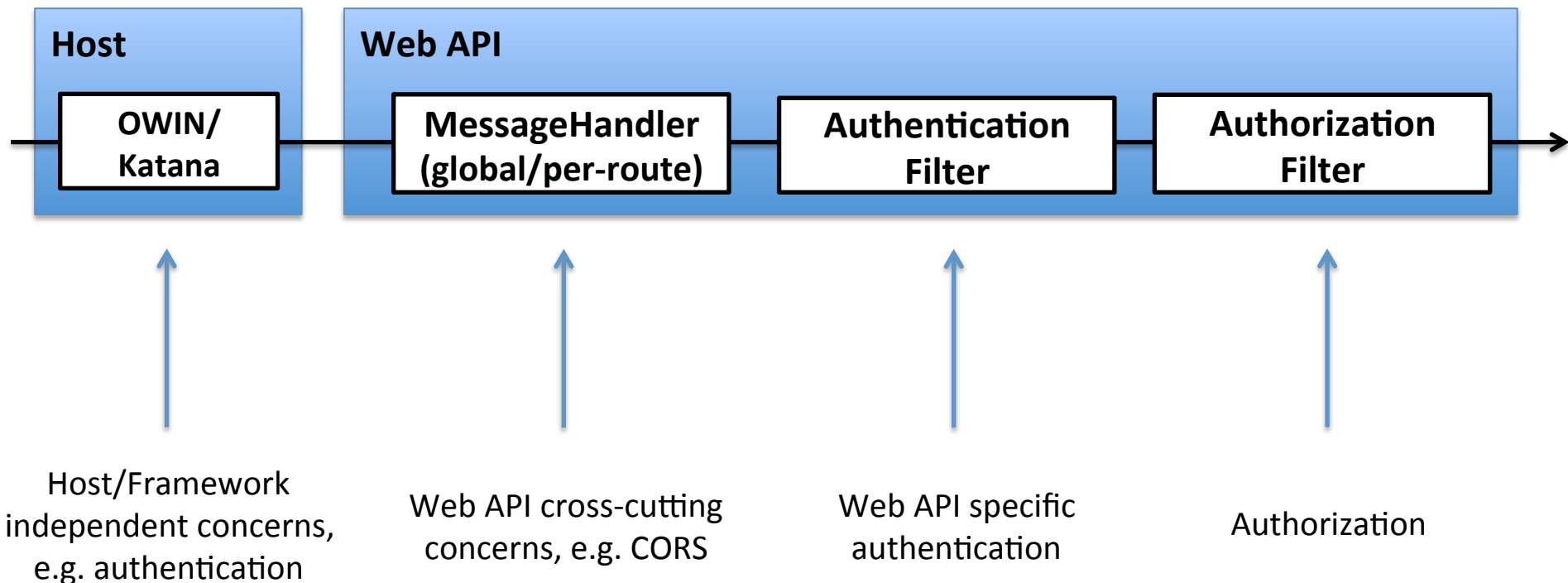


GET /service/resource

Authorization: *scheme* credential



The Web API v2 Security Pipeline



<http://www.asp.net/vnext/overview/owin-and-katana/an-overview-of-project-katana>

Katana Authentication Middleware

```
public class Startup
{
    public void Configuration(IApplicationBuilder app)
    {
        app.UseCookieAuthentication(new CookieAuthenticationOptions
        {
            AuthenticationType = "Cookies",
            // more options
        });

        app.UseGoogleAuthentication(new GoogleAuthenticationOptions
        {
            AuthenticationType = "Google",
            // more options
        });

        app.UseOAuthBearerAuthentication(new OAuthBearerAuthenticationOptions
        {
            AuthenticationType = "Bearer"
            // more options
        });
    }
}
```

Authentication filter

WebApiConfig.cs

```
config.Filters.Add(new HostAuthenticationFilter("Bearer"));
```

```
[HostAuthentication("Bearer")]  
public class TestController : ApiController  
{  
    [HostAuthentication("Google")]  
    public HttpResponseMessage Get()  
    { }  
  
    [OverrideAuthentication]  
    [HostAuthentication("Cookies")]  
    public HttpResponseMessage Delete()  
    { }  
}
```

Authorization filter

- **Determines if a resource needs authentication**
 - *[AllowAnonymous]* to skip authorization for an action
 - emits the 401 status code, if unsuccessful

```
// minimum requirement is successful authentication
[Authorize]
public DataController : ApiController
{
    [AllowAnonymous]
    public Data Get()
    { ... }

    [Authorize(Role = "Foo")]
    public HttpResponseMessage Delete(int id)
    { ... }
}
```

Custom authorization filter

- Derive from *AuthorizeAttribute*

```
public class PremiumUsersOnlyAttribute : AuthorizeAttribute
{
    protected override bool IsAuthorized(HttpContext context)
    {
        var principal = context
            .ControllerContext
            .RequestContext
            .Principal as ClaimsPrincipal;

        // custom authorization logic
    }

    protected override void HandleUnauthorizedRequest(
        HttpContext actionContext)
    {
        // custom response
    }
}
```

Resource/Action-based Authorization

- **Get rid of the tight coupling between application code and security requirements**

```
[ResourceActionAuthorize("Update", "Customer")]  
public IHttpActionResult Put(Customer customer)  
{ ... }
```

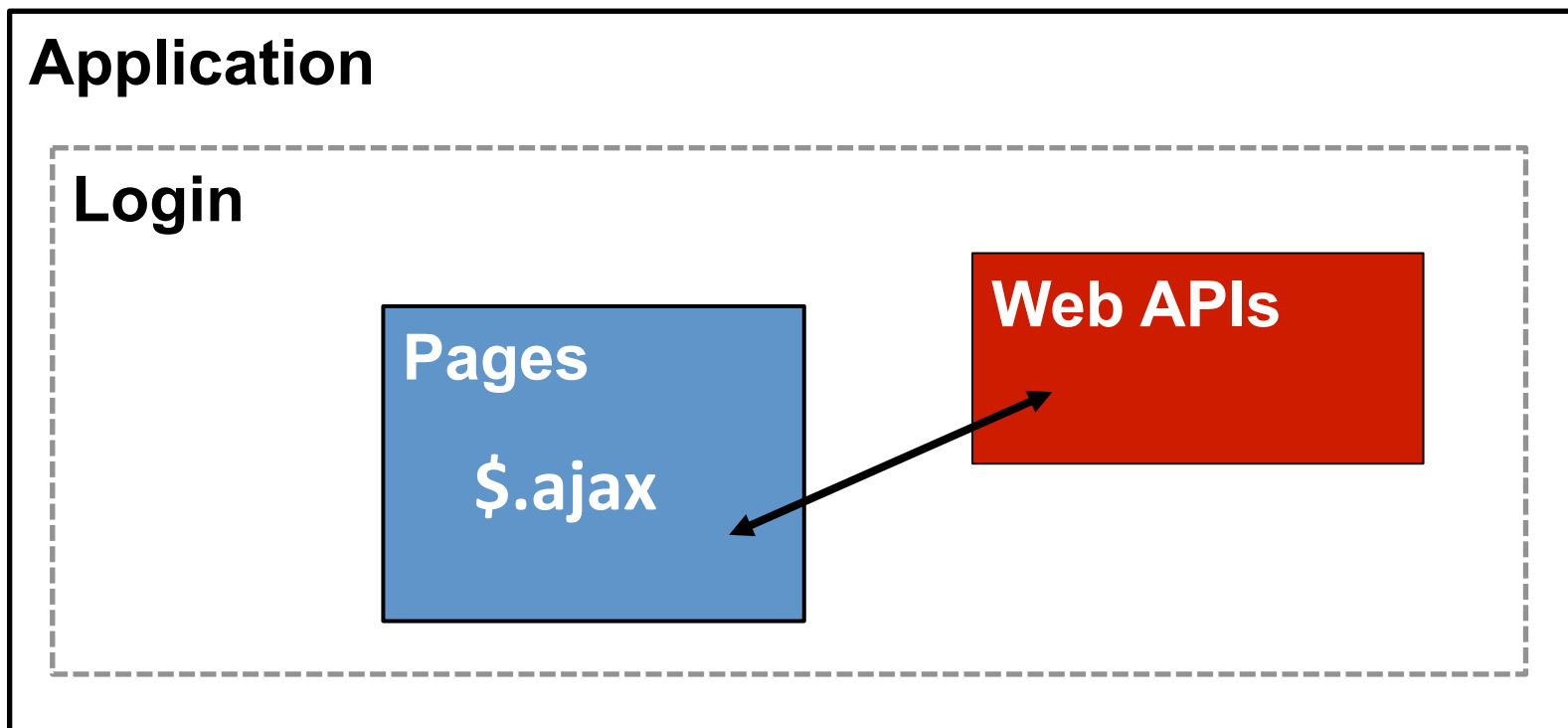
<http://thinktecture.github.com/Thinktecture.IdentityModel/>

Application Styles

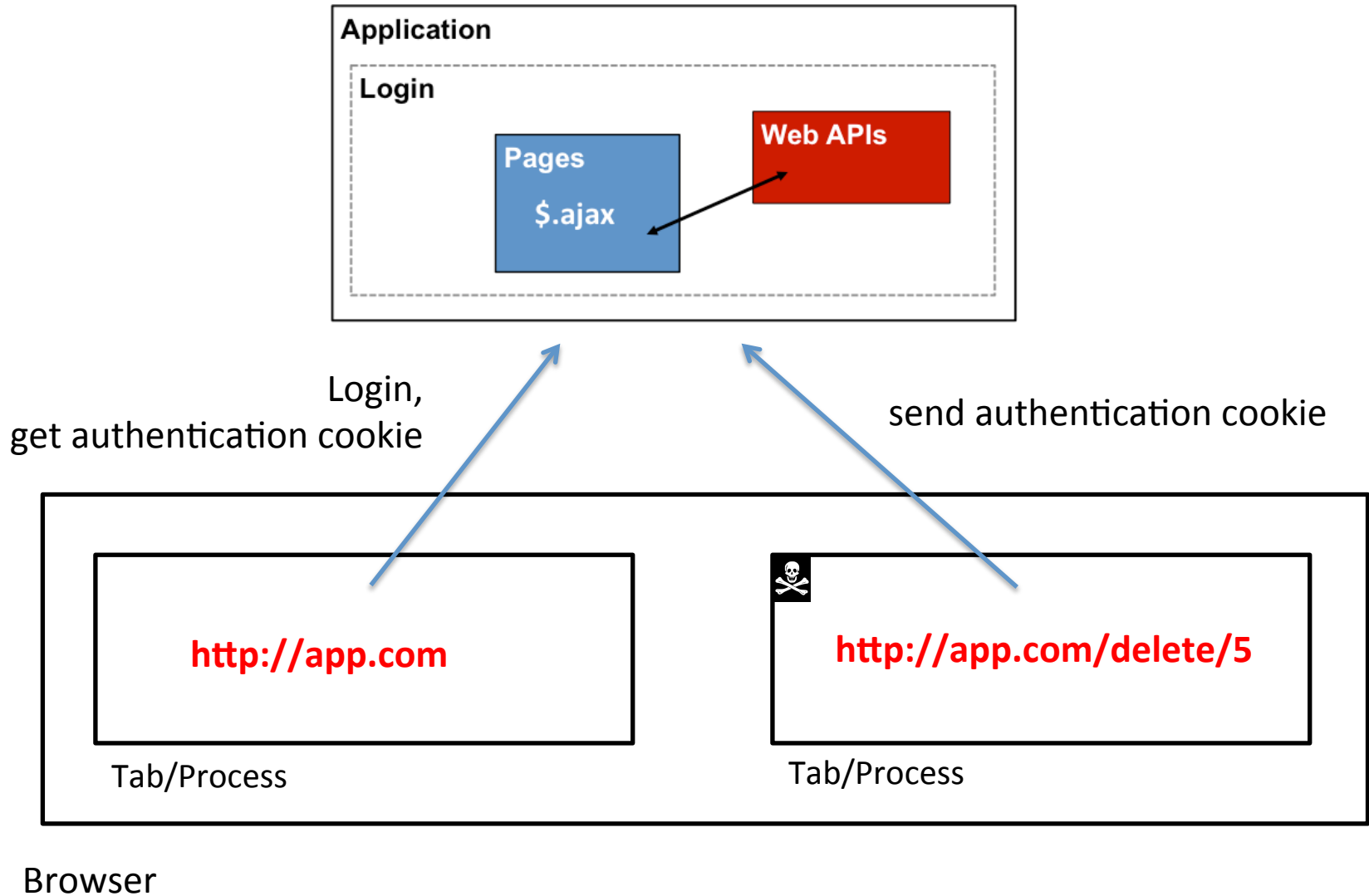
- **Same-Domain & Cross-Domain**
 - classic vs modern
- **Same Domain**
 - Browser based applications
 - Web APIs and clients live in the same domain
 - AJAX style callbacks from server-rendered pages
 - SPA applications (like the built-in template in VS2012)
 - Often cookie based security
 - potential CSRF problems

Same-Domain Scenario

- **Web APIs inherit security settings of web host**
 - e.g. cookies, Windows authentication, client certs...

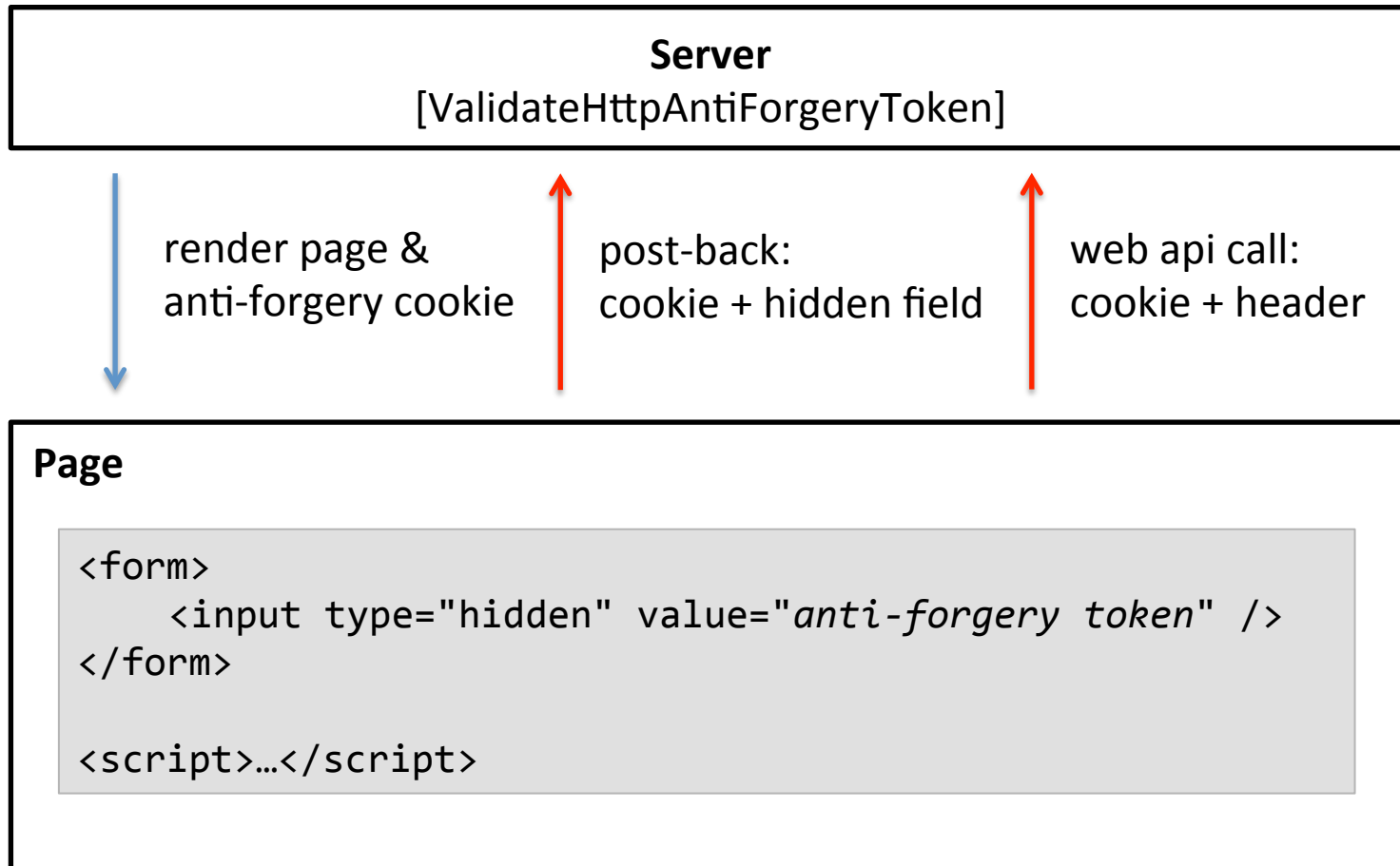


CSRF – The Problem



Web API v1 CSRF Protection

- **Part of the SPA template in MVC 4 (Update 2)**



Web API v2 CSRF Protection

- **No cookies allowed anymore...**

```
// Configure Web API to use only bearer token authentication.  
config.SuppressDefaultHostAuthentication();  
  
config.Filters.Add(new HostAuthenticationFilter(  
    OAuthDefaults.AuthenticationType));
```

WebApiConfig.cs

Application Styles II

- **Cross-Domain**
 - Web APIs and clients live in different domains
 - native apps (desktop, mobile)
 - client side JavaScript code (browser)
- **Multitude of scenarios**
 - shared secret authentication
 - CORS restrictions for JavaScript-based clients
 - token-based authentication
 - built-in token endpoint
 - OAuth2 authorization server

Shared Secret Authentication

- **HTTP Basic Authentication**
- **Shared signature approaches (e.g. hawk)**



GET /service/resource



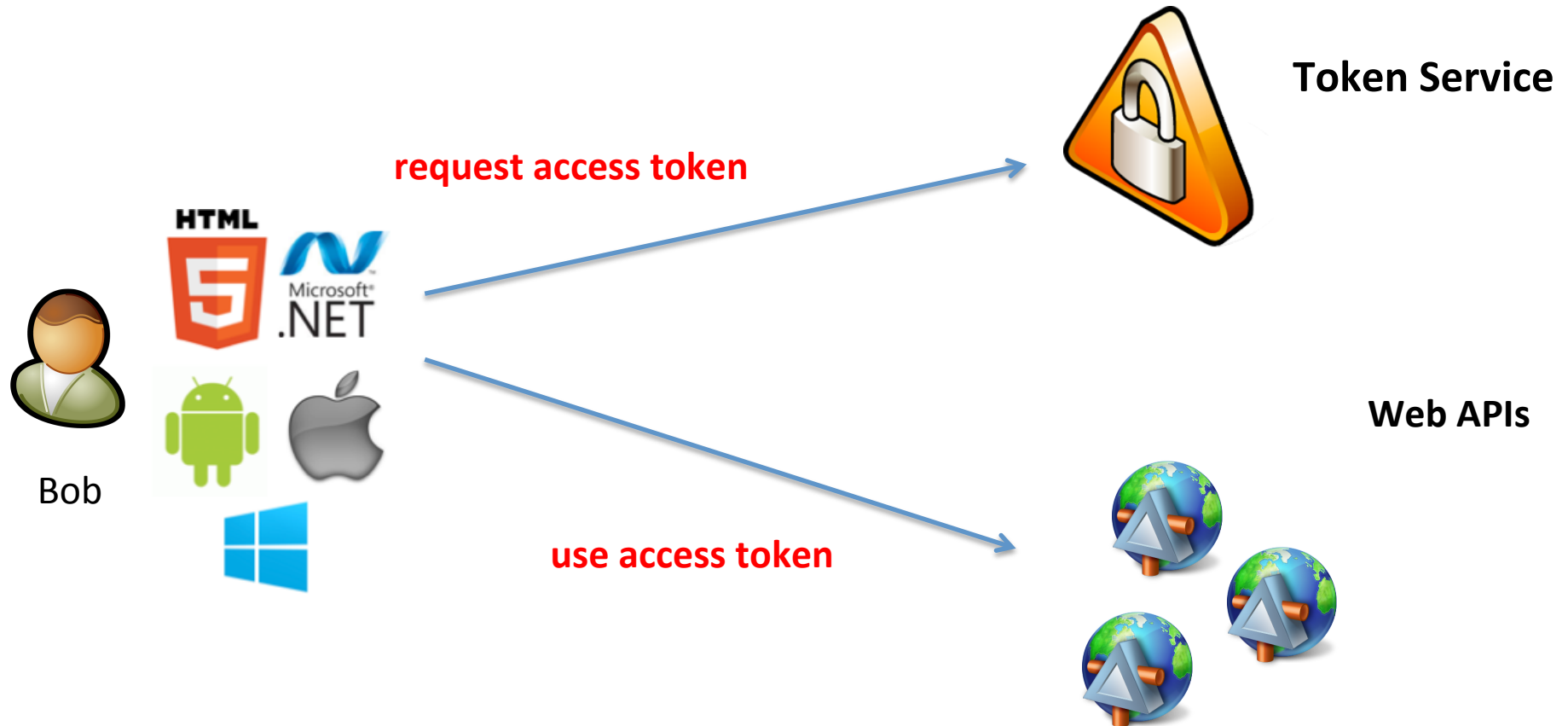
Authorization:
Basic base64(username:password)



Anti-pattern!

- **The client must store the secret or obtain it from the user (on every request)**
 - storage must be done in clear text (or reversible encryption)
- **Server has to validate the secret on every request**
 - high computational cost due to brute force protection
- **The probability of accidental exposure of the secret is increased**

Token-based Authentication

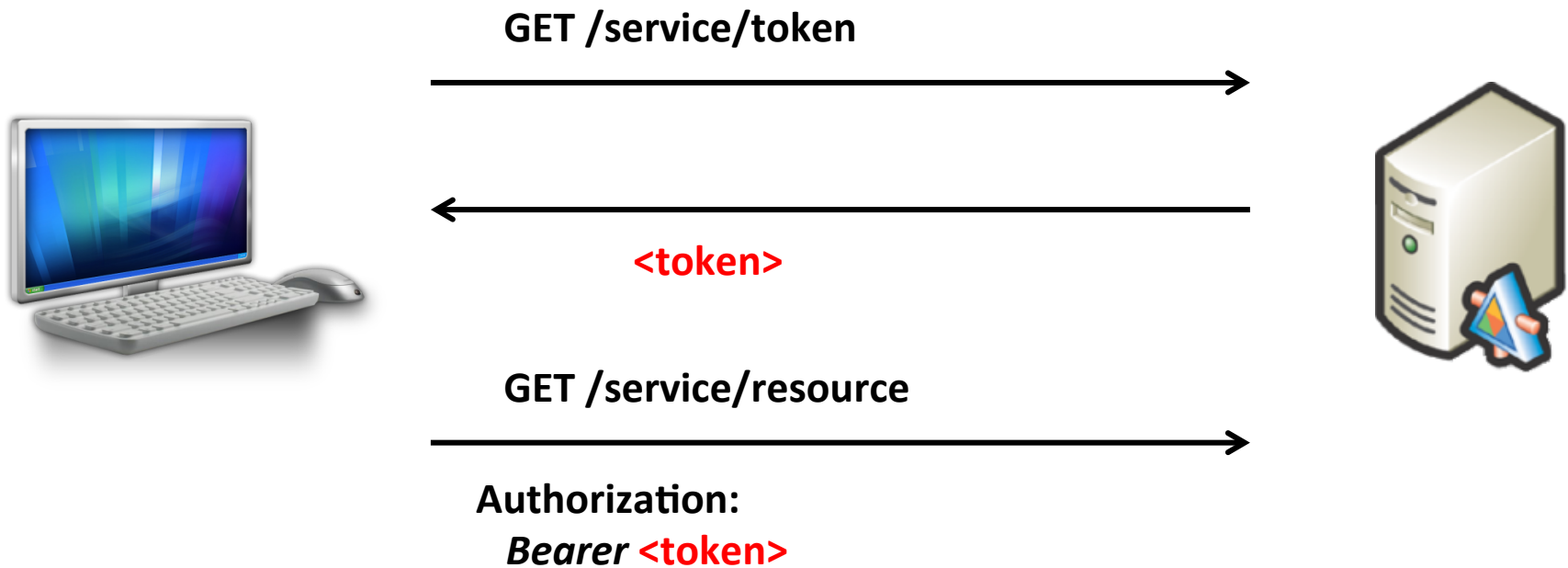


OAuth2 (RFC 6749)

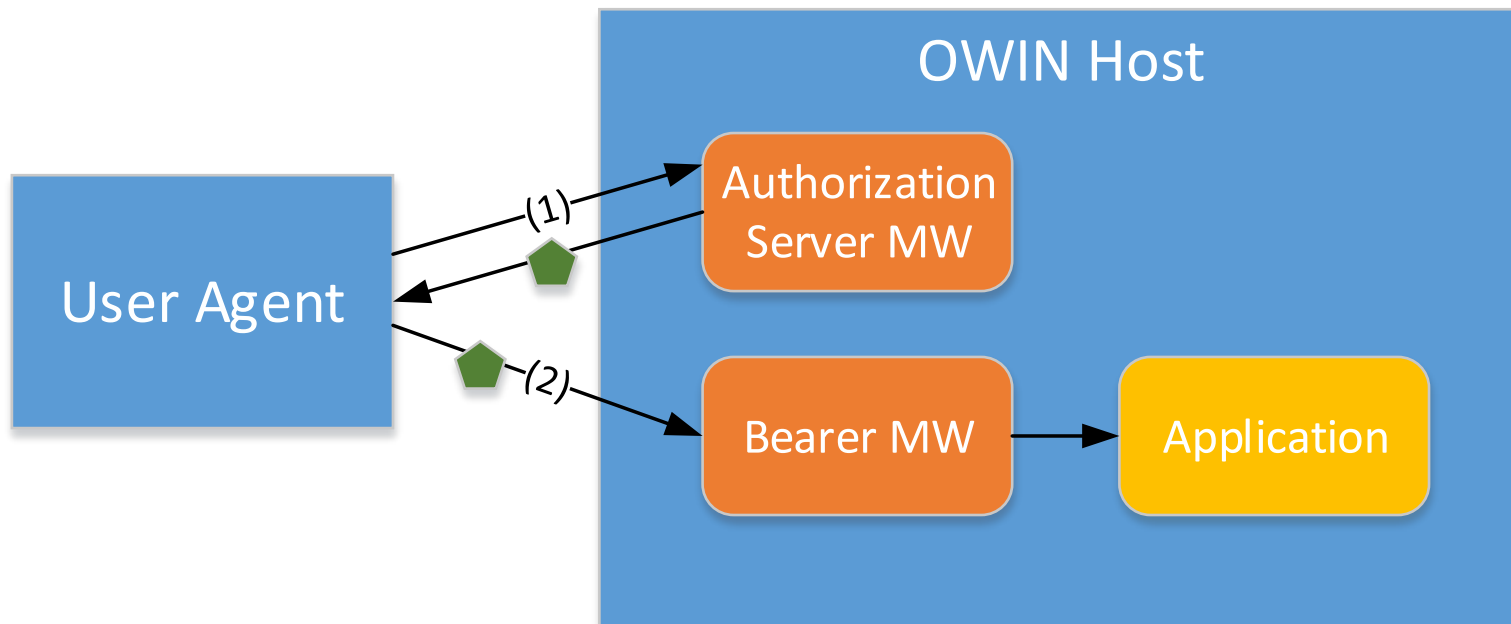
- **Framework for requesting and using access tokens for**
 - native clients
 - web clients
 - browser-based clients
- **OAuth2 introduces the concept of an Authorization Server**
 - traffic cop between clients, users and services

Embedded Authorization Server

- e.g. Swap credential with (long-lived) token



Embedded Authorization Server (Katana View)



Step 1a: Token Request

Resource Server



Authorization Server

POST /token

Authorization: Basic (client_id:secret)

**grant_type=password&
scope=resource&
user_name=owner&
password=password&**



Resource Owner



Client

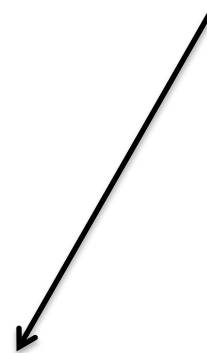
Step 1b: Token Response

Resource Server



Authorization Server

```
{  
  "access_token" : "abc",  
  "expires_in" : "3600",  
  "token_type" : "Bearer",  
  "refresh_token" : "xyz"  
}
```

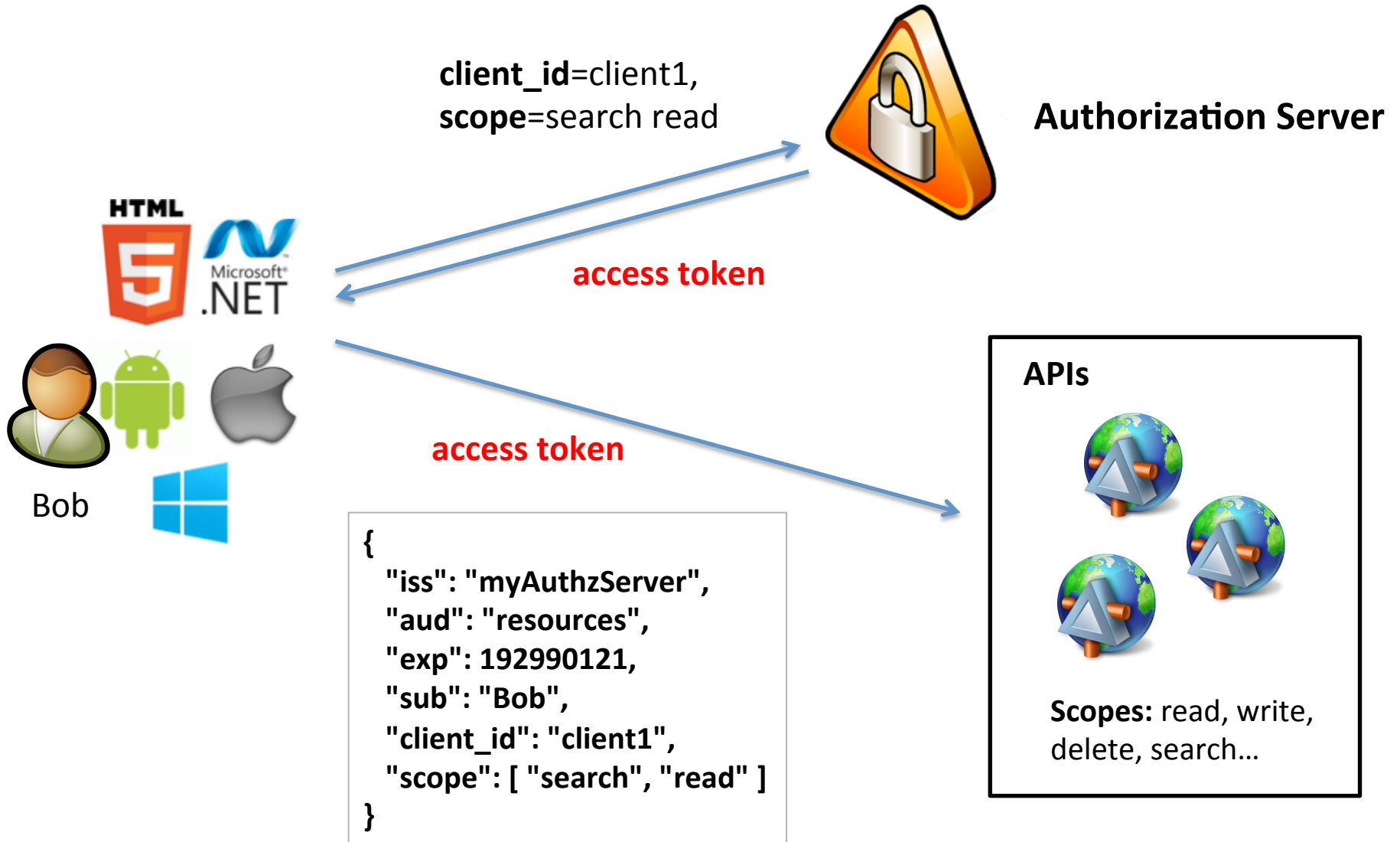


Resource Owner



Client

More advanced scenarios



JSON Web Token (JWT)

Header

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

Claims

```
{  
  "iss": "http://myIssuer",  
  "exp": "1340819380",  
  "aud": "http://myResource",  
  "sub": "alice",  
  
  "client_id": "xyz",  
  "scope": ["read", "search"]  
}
```

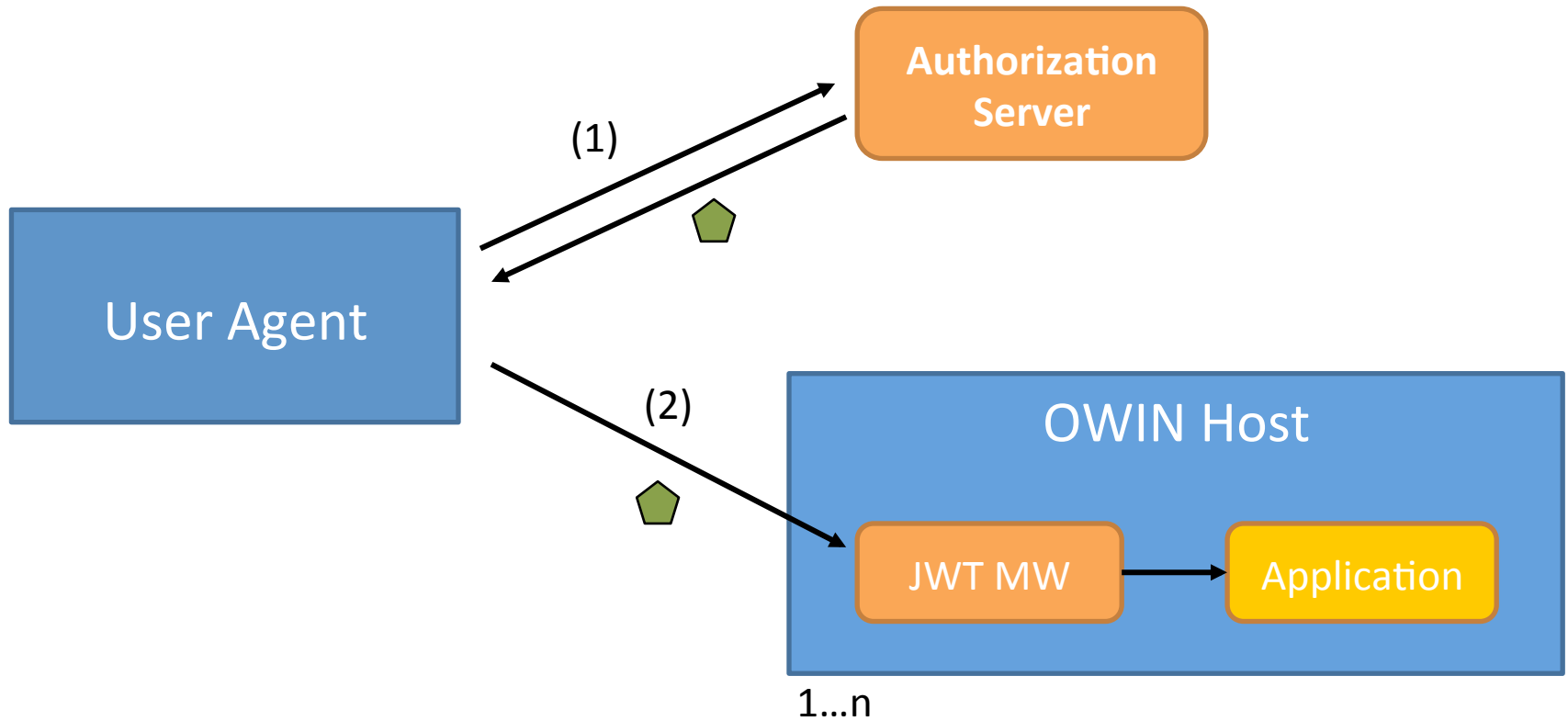
eyJhbGciOiJIub251In0.eyJpc3MiOiJqb2UiLA0KICJleHAiOjEzMDQ0ODAsDQogImh0dHA6Ly9leGft

Header

Claims

Signature

External Authorization Server (Katana View)



thinktecture

AuthorizationServer & IdentityServer v3

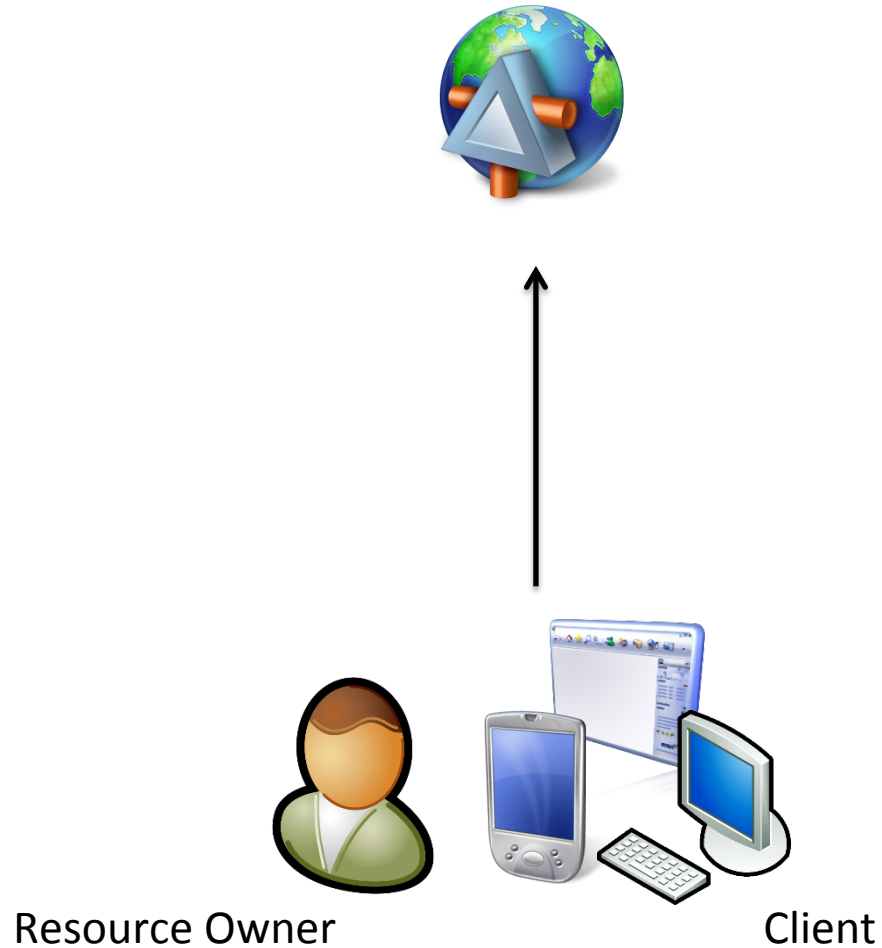
<https://github.com/thinktecture/Thinktecture.AuthorizationServer>

<https://github.com/thinktecture/Thinktecture.IdentityServer.v3>

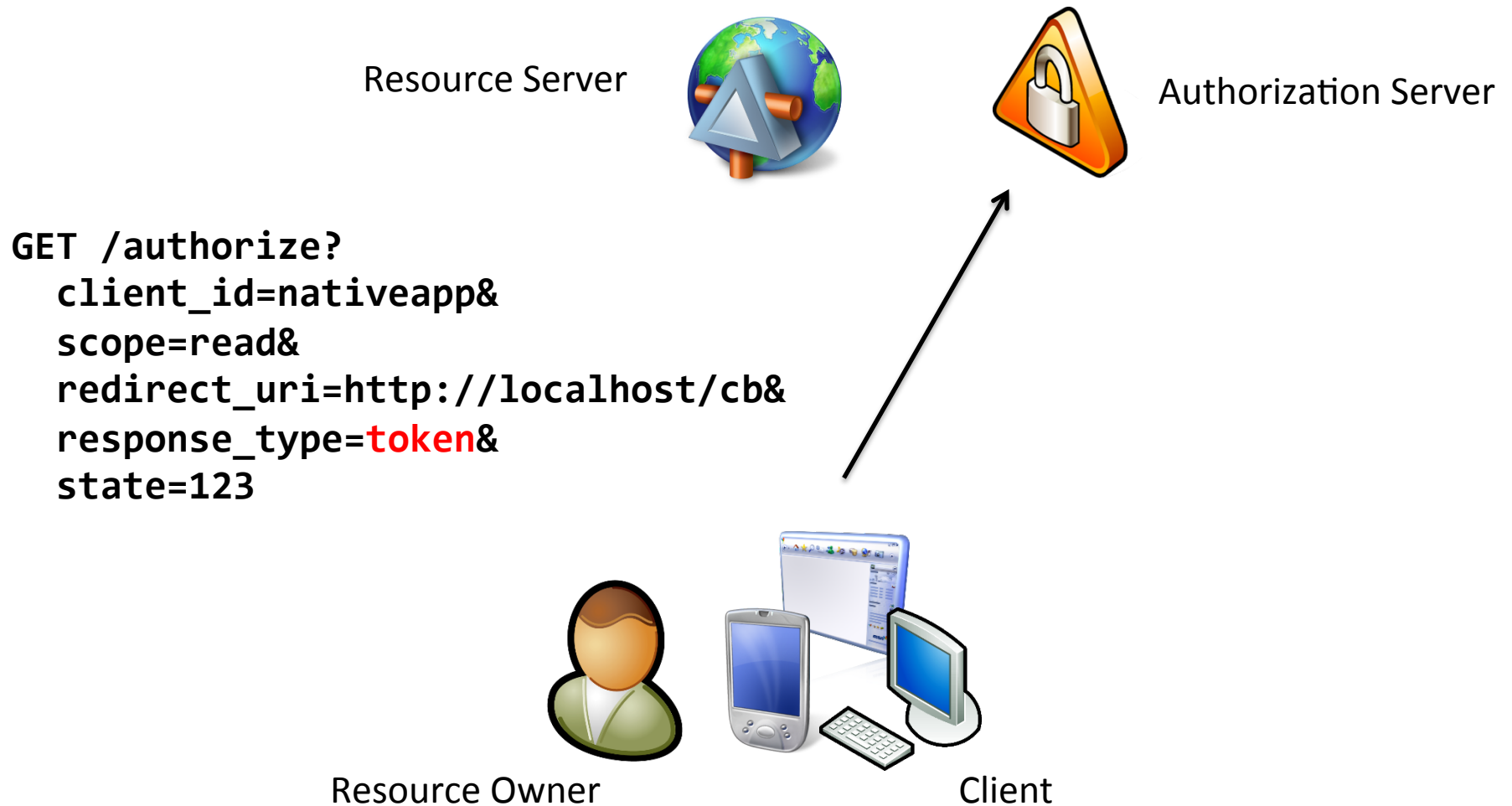
Separating user credentials from the client...

- **Local / mobile / user-agent based clients**
 - Implicit Flow
- **Server-based / confidential clients**
 - Authorization Code Flow

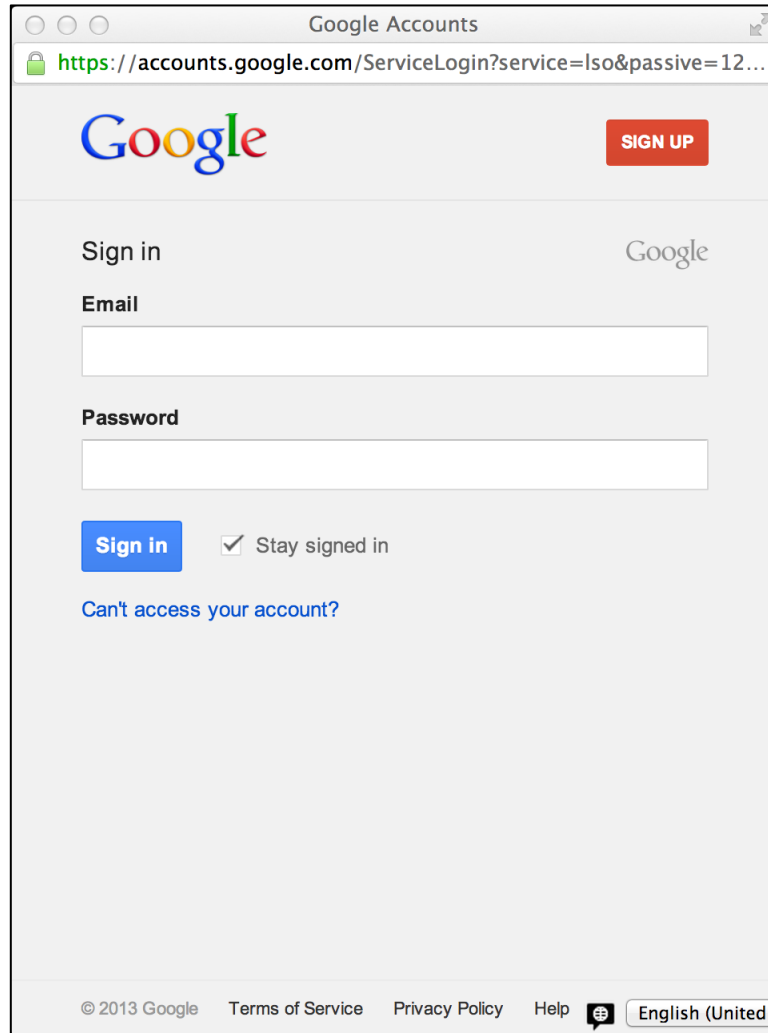
Implicit Flow (Native / Local Clients)



Step 1a: Authorization Request

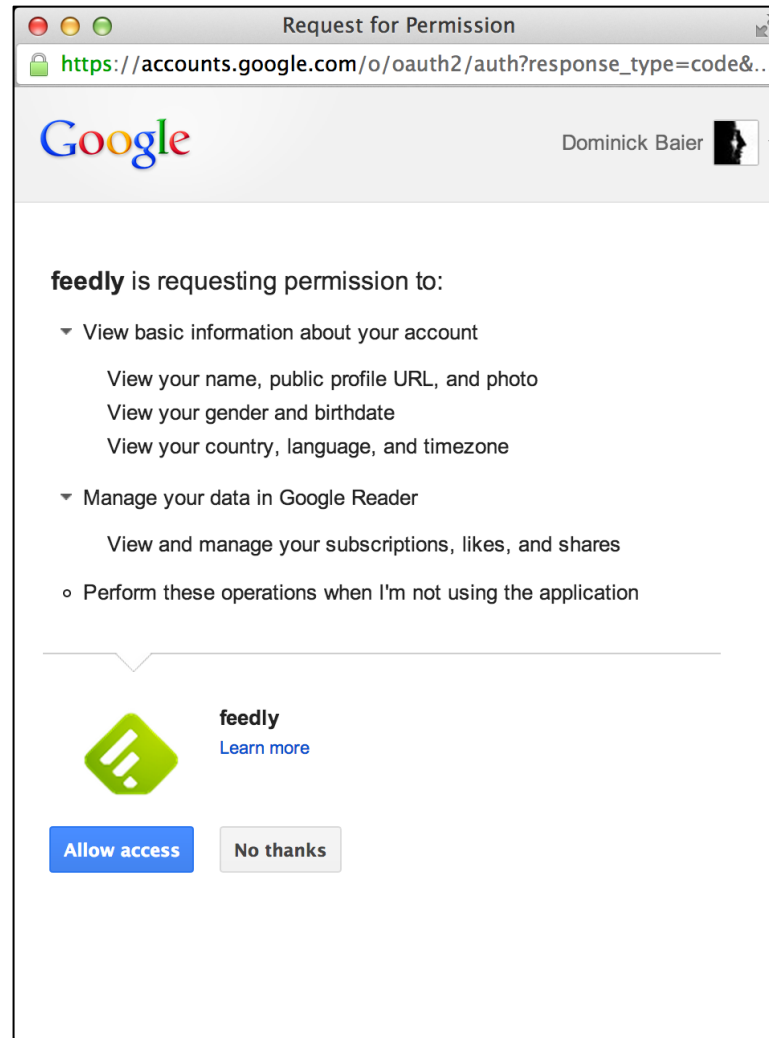


Step 1b: Authentication



The image shows a browser window titled "Google Accounts" with the URL <https://accounts.google.com/ServiceLogin?service=Iso&passive=12...>. The page features the Google logo and a red "SIGN UP" button. Below the logo, there is a "Sign in" section with a "Google" logo to its right. The "Email" field is empty, and the "Password" field is also empty. A blue "Sign in" button is located below the password field, along with a checked checkbox for "Stay signed in". A link for "Can't access your account?" is positioned below the "Sign in" button. At the bottom of the page, there is a footer containing copyright information for 2013 Google, links for "Terms of Service", "Privacy Policy", and "Help", and a language selector set to "English (United S)".

Step 1c: Consent



Twitter Consent

Authorize Twitter for Windows to use your account?

This application **will be able to:**

- Read Tweets from your timeline.
- See who you follow, and follow new people.
- Update your profile.
- Post Tweets for you.
- Access your direct messages.

Username or email

Password

Remember me · [Forgot password?](#)

This application **will not be able to:**

- See your Twitter password.



Twitter for Windows

www.twitter.com

Official Twitter for Windows application.

Evernote Consent

The screenshot shows a web browser window with the address bar displaying `https://www.evernote.com/OAuth.action?oauth_token=macoscope-6716.13EA3FD5A93.656E2D6D61...`. The Evernote logo and the user name "Dominick Baier" are visible in the top navigation bar. The main content area is titled "Authorize BubbleBrowser to access your account".

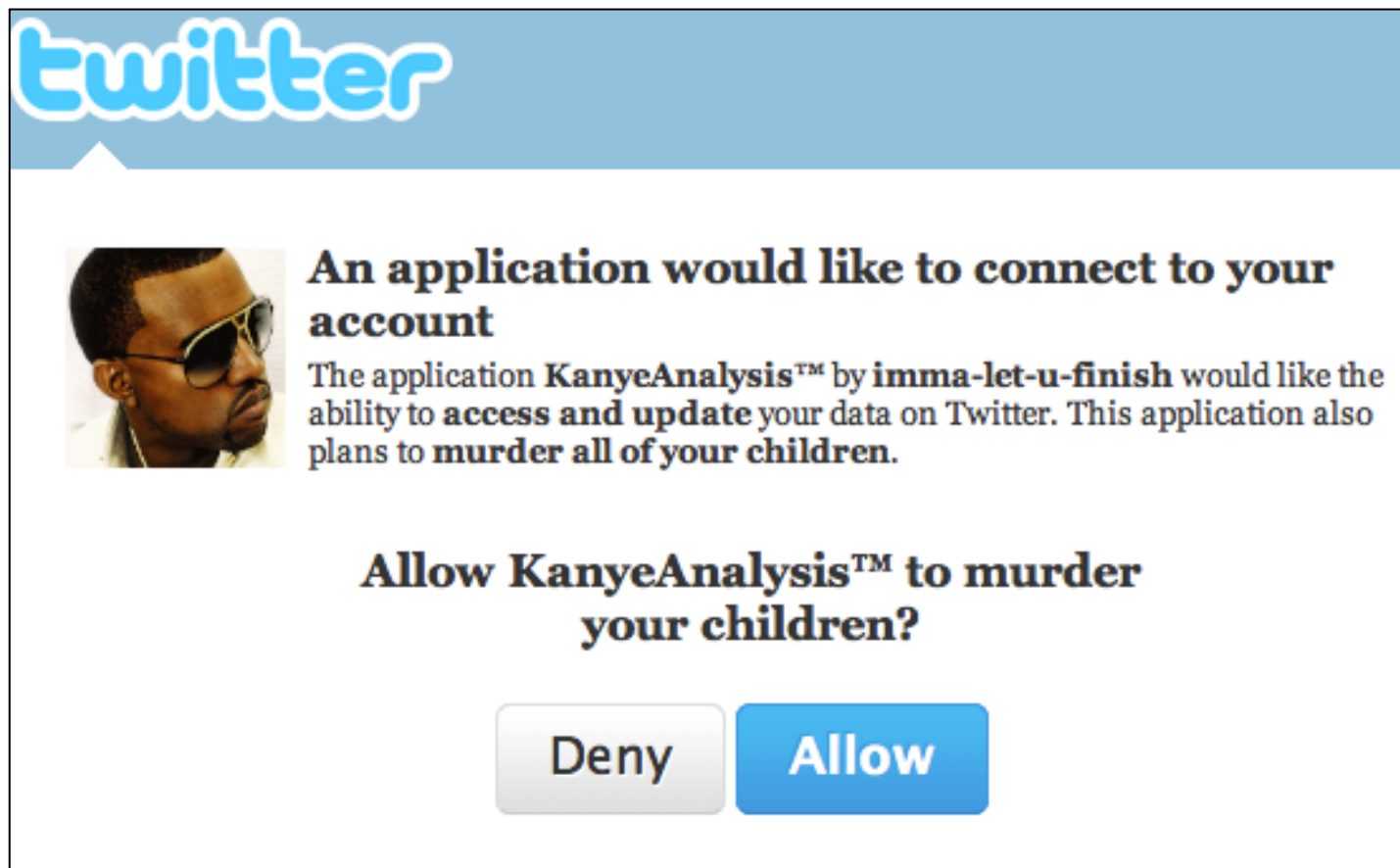
Below the title, there is a section for "For 1 year (change)" and "Signed in as Dominick Baier (not you?)". A large green "Authorize" button is present. A list of permissions is shown, with some checked and some unchecked:

- BubbleBrowser will be able to:
 - ✓ Create notes, notebooks and tags
 - ✓ Update notes, notebooks and tags
 - ✓ List notebooks
 - ✓ Retrieve notes
- BubbleBrowser will not be able to:
 - ✗ Delete notebooks
 - ✗ Access account information
 - ✗ Permanently delete notes
 - ✗ Update user account information
 - ✗ ???ROAuthAction.perm.2048???

A modal dialog box is open in the foreground, titled "Authorize BubbleBrowser for:". It contains a dropdown menu with "1 year" selected and two buttons: "Cancel" and "Save".

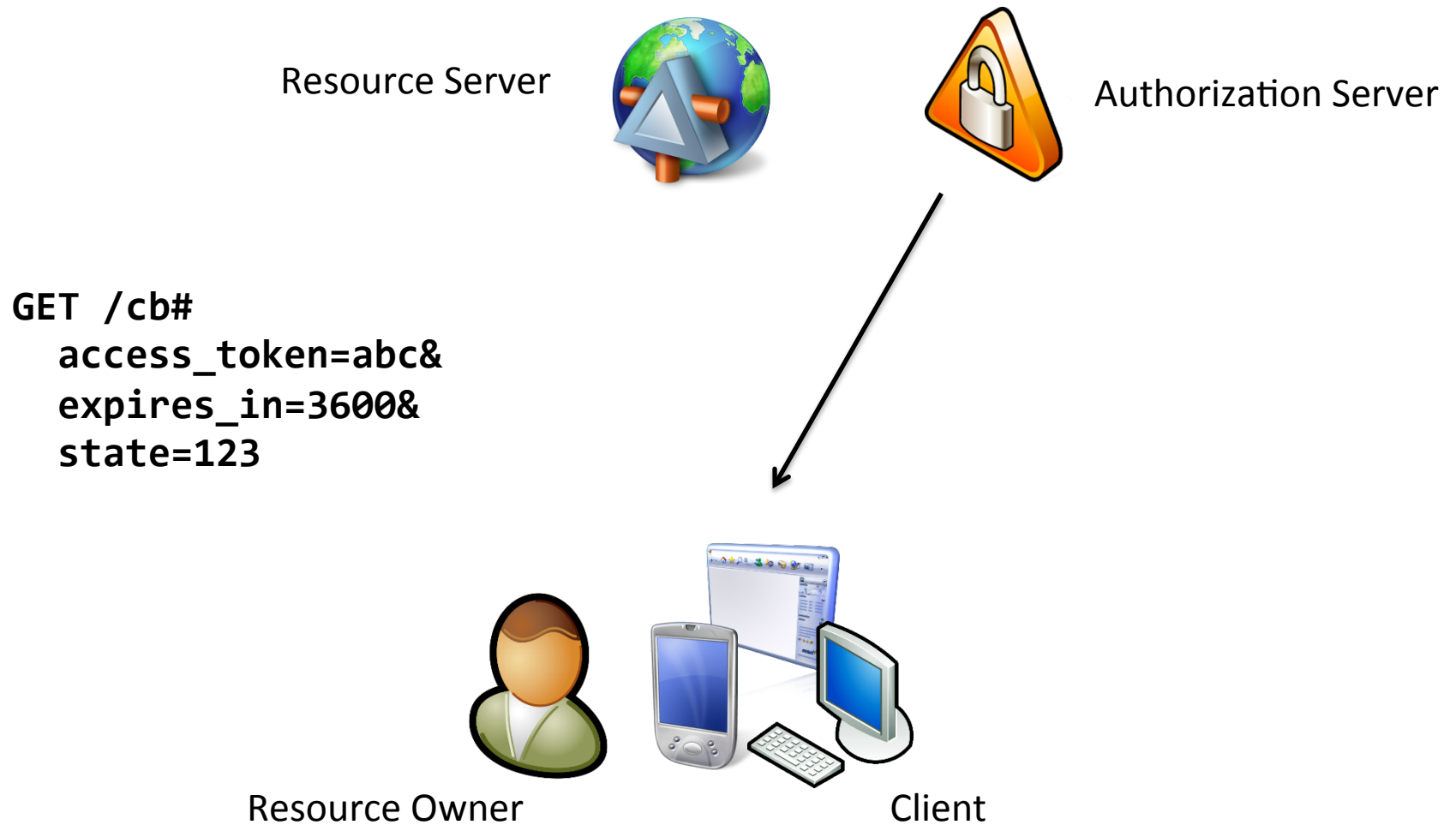
At the bottom of the page, there are links for "Terms of Service", "Privacy Policy", and "Copyright 2013 Evernote Corporation. All rights reserved."

The Consent Screen is important!



http://zachholman.com/2011/01/oauth_will_murder_your_children/

Step 1d: Token Response



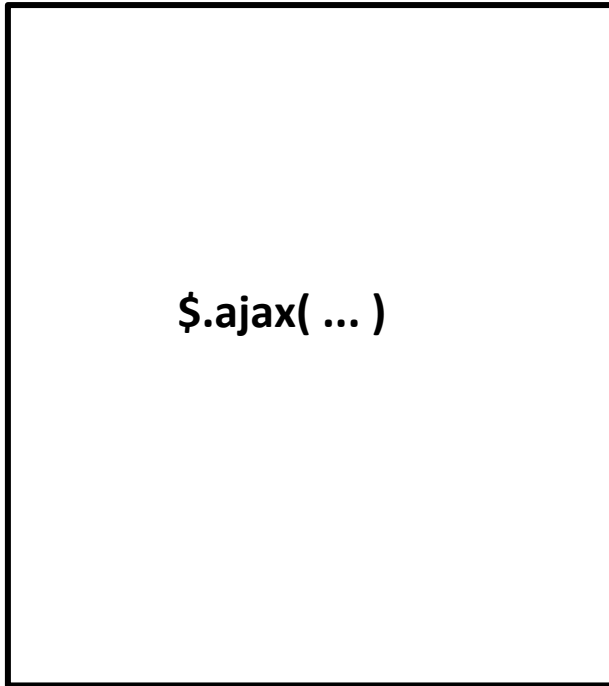
Summary – Implicit Flow

- **User enters credentials at the authorization server**
 - not at the client
- **authorization server returns (short lived) access token**
 - to reduce exposure of token
- **Often combined with OS helper mechanisms**
 - cookie container
 - native APIs

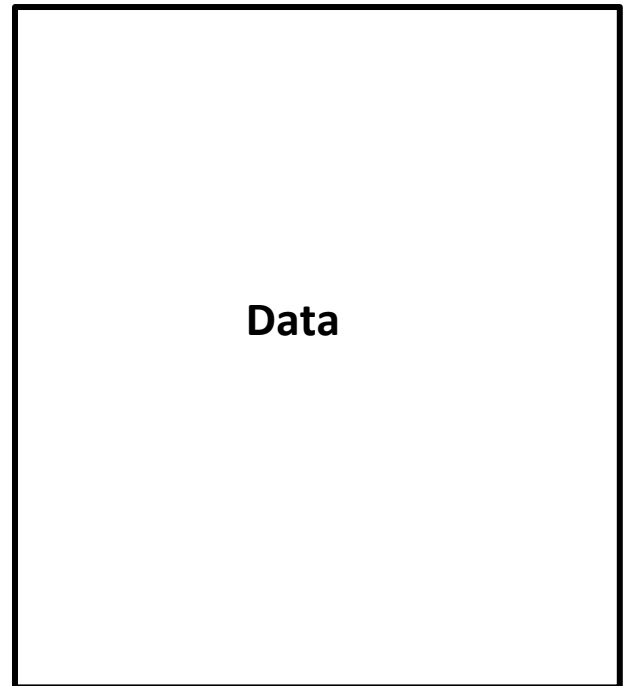
Excursion: CORS

(Cross Origin Resource Sharing)

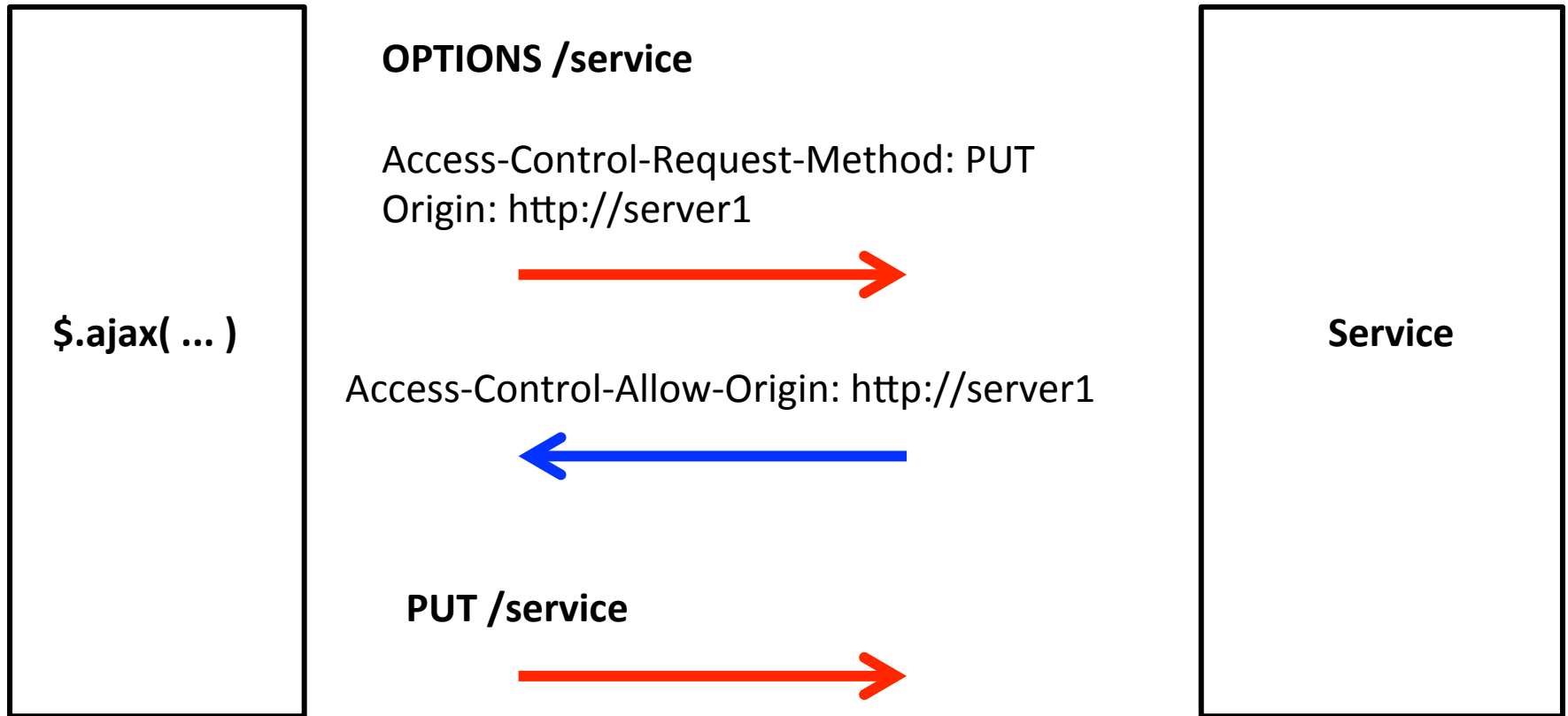
<http://server1/client.htm>



<http://server2/service>



CORS Sample



CORS in Web API v2

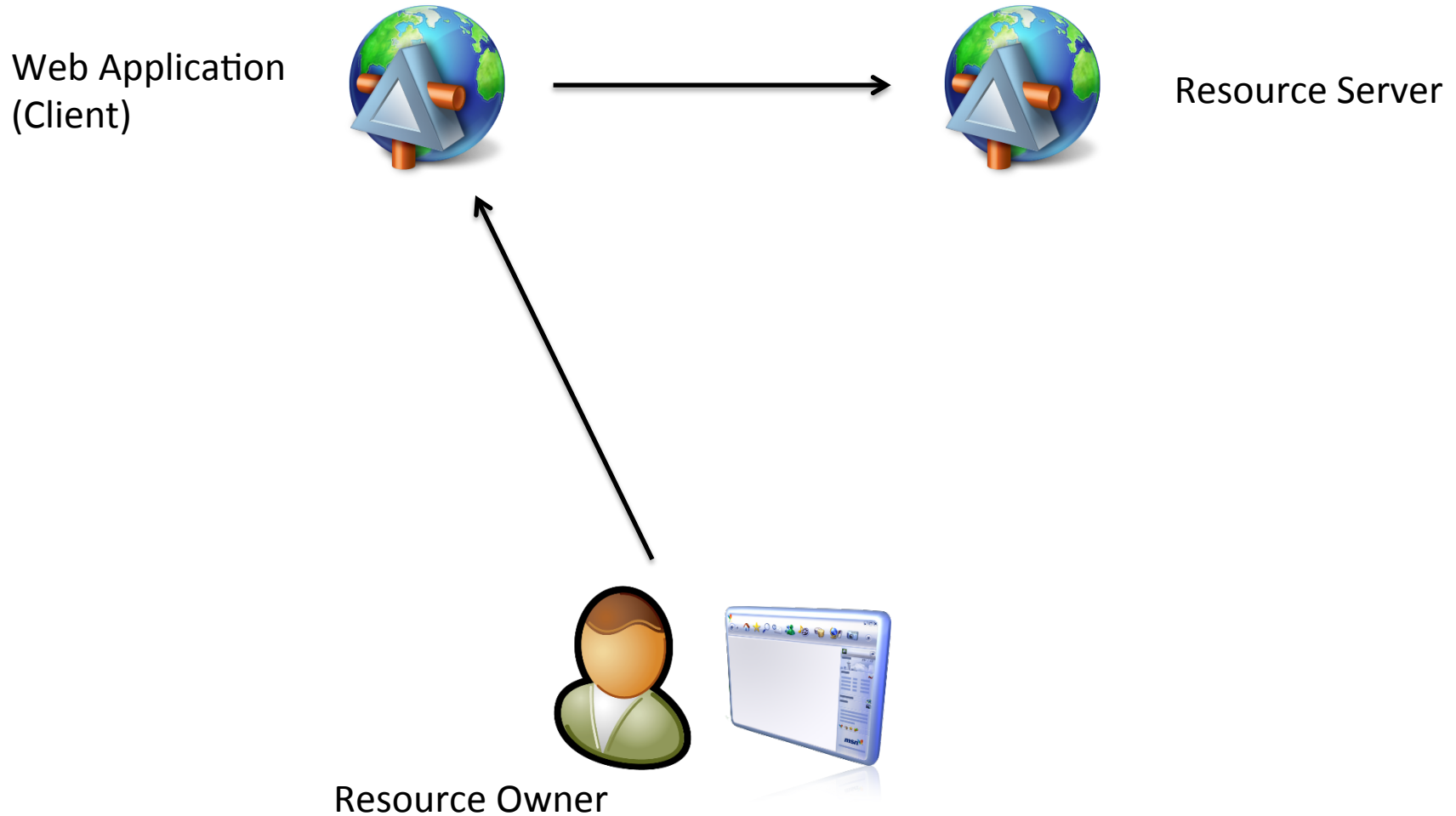
Thinkecture.IdentityModel.Http.Cors.WebApi



System.Web.Cors

```
[EnableCors("origin", "headers", "verbs")]  
public class CustomersController : ApiController  
{  
    // actions...  
}
```

Authorization Code Flow (Server-based Clients)



Step 1a: Authorization Request

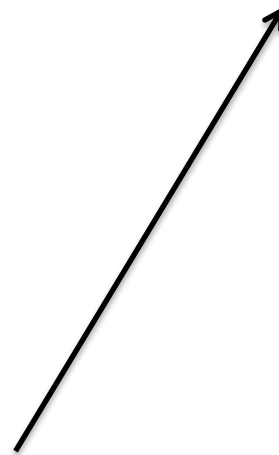
Web Application
(Client)



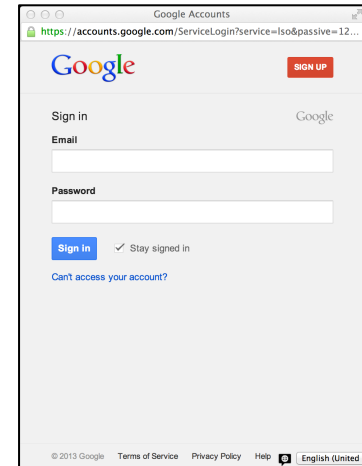
```
GET /authorize?  
client_id=webapp&  
scope=read&  
redirect_uri=https://webapp/cb&  
response_type=code&  
state=123
```



Authorization Server



Resource Owner



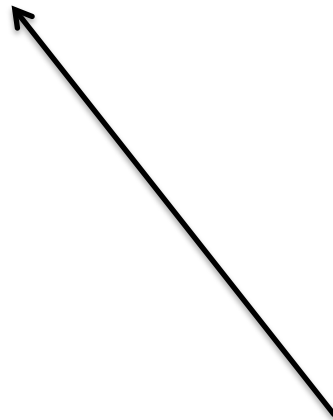
Step 1d: Authorization Response

Web Application
(Client)



Authorization Server

GET /cb?
code=xyz&
state=123



Resource Owner

Step 2a: Token Request

Web Application
(Client)

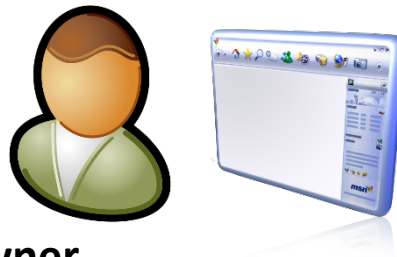


Authorization Server

POST /token

Authorization: Basic (client_id:secret)

**grant_type=authorization_code&
authorization_code=xyz**



Resource Owner

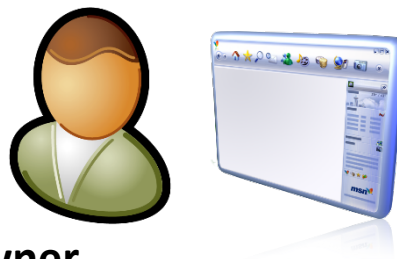
Step 2b: Token Response

Web Application
(Client)



Authorization Server

```
{  
  "access_token" : "abc",  
  "expires_in" : "3600",  
  "token_type" : "Bearer",  
  "refresh_token" : "xyz"  
}
```



Resource Owner

Step 3: Resource Access

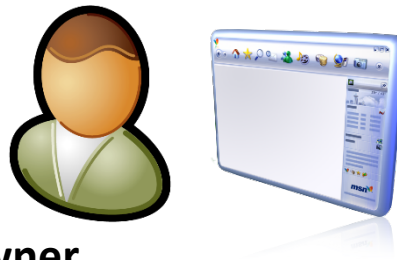
Web Application
(Client)



Resource Server

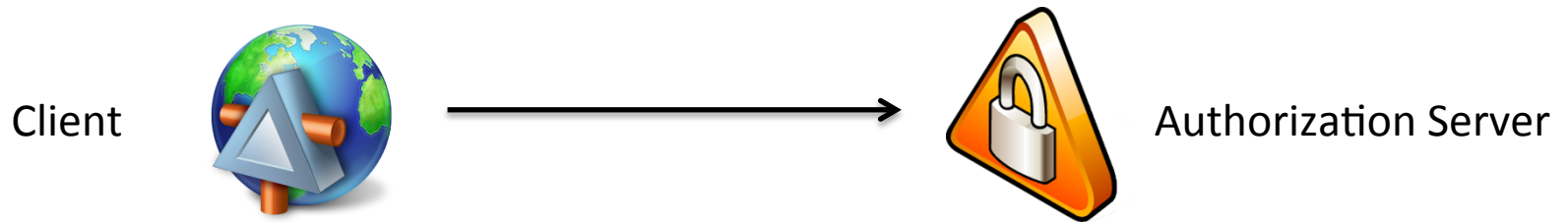
GET /resource

Authorization: Bearer access_token



Resource Owner

(Step 3: Refreshing the Token)




POST /token

Authorization: Basic (client_id:secret)

**grant_type=refresh_token&
refresh_token=xyz**

Refresh Token Management (Flickr)



leastprivilege

[Apps By You](#) | [Apps You're Using](#) | [Your Favorite Apps](#)

Below is a list of applications that you've given permission to interact with your Flickr account. It doesn't include apps that only use public photos and don't need to be authorized.




If you want to stop using one of these apps, click its "Remove permission" link.

Application	Permissions	
Adobe Photoshop Lightroom http://www.adobe.com/products/photoshoplightroom/	delete	Remove permission?
Flickr for Windows Phone 7 http://social.zune.net/redirect?type=phoneApp&id=2e49fb07-592b-e011-854c-00237de2db9e	delete	Remove permission?
Photorank.me	read	Remove permission?
Microsoft http://aka.ms/flickr	write	Remove permission?

Refresh Token Management (Dropbox)

My apps

You have given these apps access to your Dropbox account.

App name	Publisher	Access type	
 1Password	AgileBits	Full Dropbox	×
 1Password for Android	AgileWebSolutions Inc	Full Dropbox	×
 Dropbox Windows 8	Dropbox Windows 8	Official app	×

Refresh Token Management (Microsoft Live)

Microsoft account

Overview

Notifications

Permissions

Linked accounts

Kids' accounts

Add accounts






Manage accounts

Apps and Services

Billing

Apps and services you've given access

These apps and services can access some of your info. Choose one to view or edit the details.

	WordPress.com You last used WordPress.com on 6/6/2012. Edit		WLID Test You last used WLID Test on 5/11/2012. Edit
	Microsoft Minesweeper You last used Microsoft Minesweeper on 9/26/2012. Edit		idsrv You last used idsrv on 2/20/2013. Edit
	Dominick's App You last used Dominick's App on 2/27/2013. Edit		

Summary – Code Flow

- **Designed for "confidential" clients**
 - client can store secret securely
 - client authentication and authorization based on client identity possible
 - typically server-based applications
- **Accountability is provided**
 - access token never leaked to the browser
- **Long-lived access can be implemented**

Summary

- **HTTP has a very simple security model**
- **Correct handling of SSL is paramount**
- **Same- vs Cross-Origin applications**

- **Think about CSRF, CORS**
- **Token based (and thus cookie-less) authentication is the way to go**
 - separate client from API
 - embedded authorization server
 - full blown authorization server (product)