# Welcome to Ocelot

Thanks for talking a look at the Ocelot documentation. Please use the left hand NAV to get around. I would suggest taking a look at introduction first.

# Big Picture

Ocelot is aimed at people using .NET running a micro services / service orientated architecture that need a unified point of entry into their system.

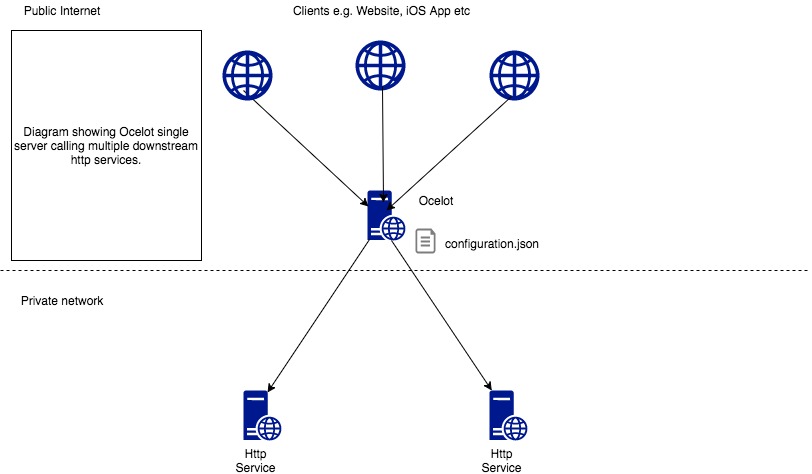
In particular I want easy integration with IdentifyServer reference and bearer tokens.

Ocelot is a bunch of middlewares in a specific order.

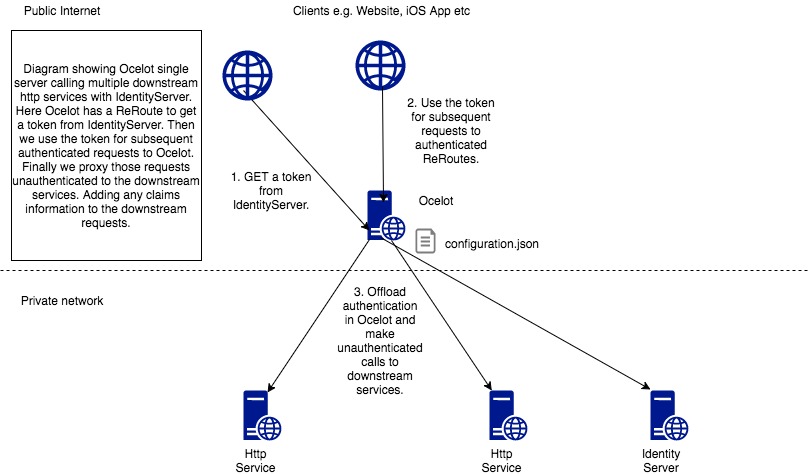
Ocelot manipulates the HttpRequest object into a state specified by its configuration until it reaches a request builder middleware where it create a HttpRequestMessage object which is used to make a request to downstream service. The middleware that makes the request is the last thing in the Ocelot pipeline. It does not call the next middleware. There is a piece of middleware that maps the HttpResponseMessage onto the HttpResponse object and that is returned to the client. That is basically it with a bunch of other features.

The following are configurations that you use when deploying Ocelot.

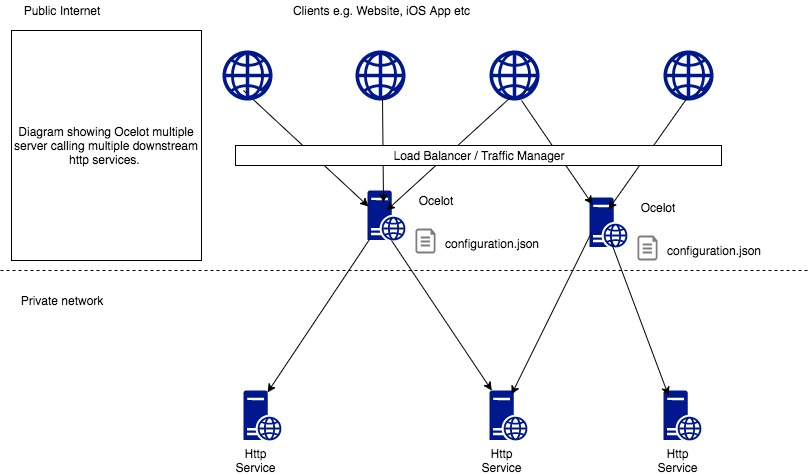
## Basic Implementation



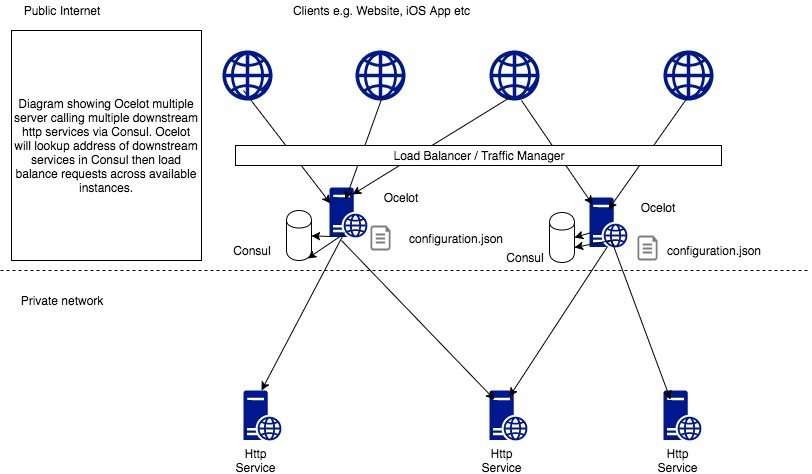
## With IdentityServer



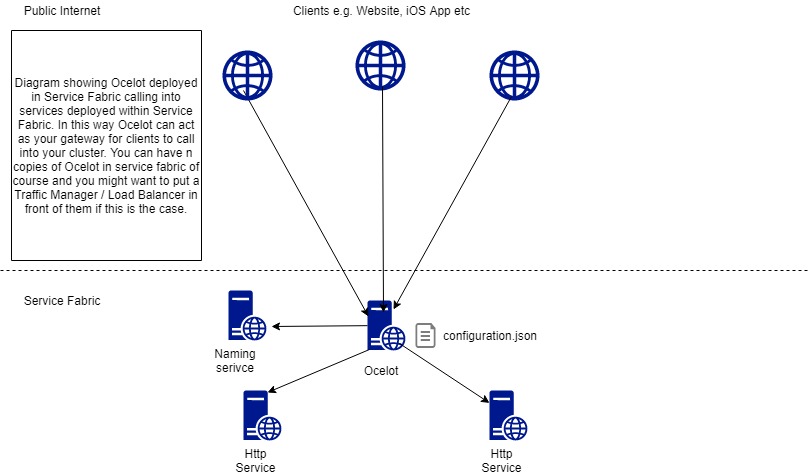
## Multiple Instances



## With Consul



## With Service Fabric



# Getting Started

Ocelot is designed to work with .NET Core only and is currently on netcoreapp3.1.

## .NET Core 3.1

**Install NuGet package**

Install Ocelot and its dependencies using nuget. You will need to create a netcoreapp 3.1 project and bring the package into it. Then follow the Startup below and Configuration sections to get up and running.

Install-Package Ocelot

All versions can be found here.

**Configuration**

The following is very basic ocelot.json. It won’t do anything but should get Ocelot starting.

{

“Routes”:[],

“GlobalConfiguration”: {

“BaseUrl”: “https://api.mybusiness.com”

}

If you want some example that actually does something use the following:

{

“Routes”: [

“DownstreamPathTemplate”: “/todo/{id}”,

“DownstreamScheme”: “https”,

“DownstreamHostAndPorts”: [

{

“Host”: “jsonplaceholder.typicode.com”,

“Port”: 443

],

“UpstreamPathTemplate”: “/todos/{id}”,

“UpstreamHttpMethod”: [ “Get” ]

],

“GlobalConfiguration”: {

“BaseUrl”: “https://localhost:5000”

}

}

The most important thing to note here is BaseUrl. Ocelot needs to know the URL running under in order to do Header find & replace and for certain administration configurations. When setting this URL it should be the external URL that clients will see Ocelot running on e.g. If you are running containers Ocelot might run on the URL http://123.12.1.1.:6543 but has something like nginx in front of it responding on <https://api.mybusiness.com>. In this case the Ocelot base URL should be <https://api.mybusiness.com>.