YARP - Yet Another Reverse Proxy

Dino Wang



歡迎參加我們的每週四固定聚會

Who am I

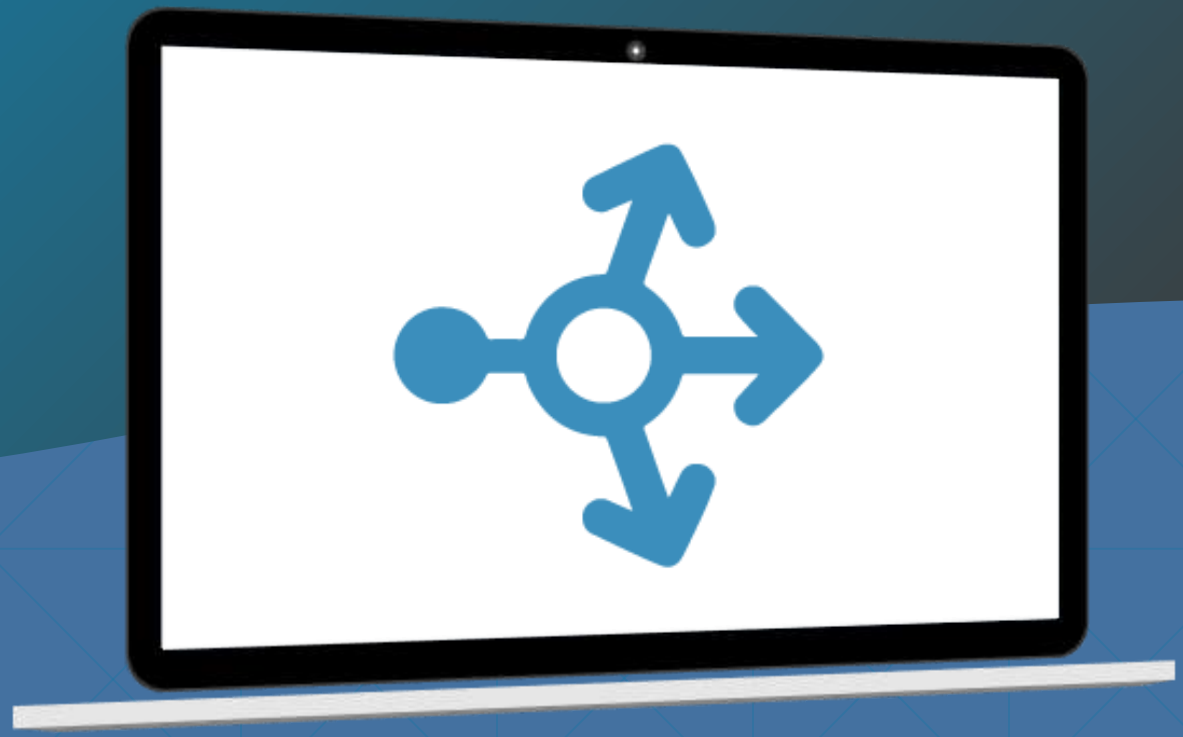
Dino Wang

- Microsoft – Cloud Solution Architect
- twMVC – Co-funder

Agenda

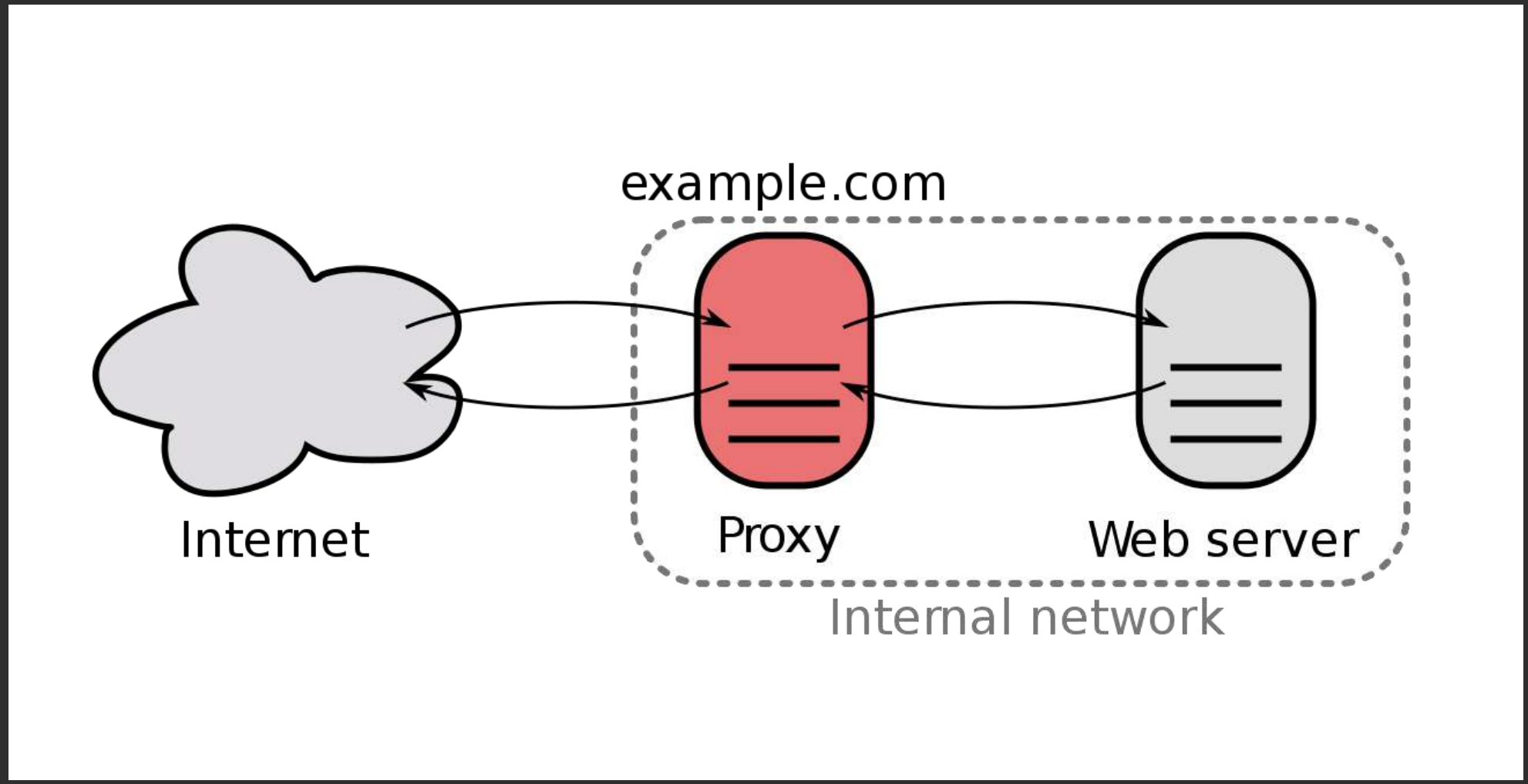
- Reverse Proxy overview
- What is YARP
- Why YARP?
- How to use YARP
- Use Cases

Reverse Proxy overview



歡迎參加我們的每週四固定聚會

A Reverse Proxy



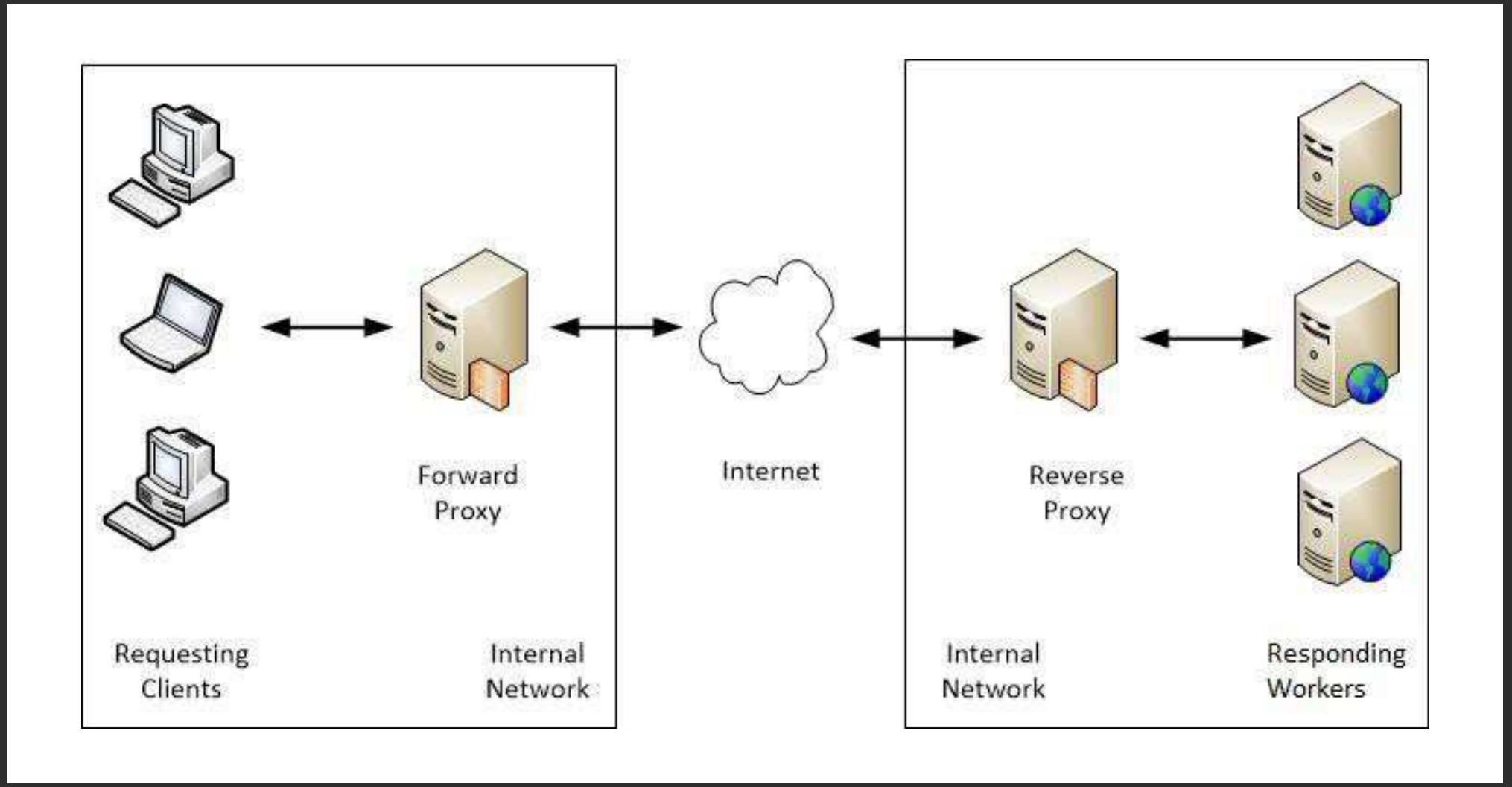
反向代理 - 維基百科，自由的百科全書 (wikipedia.org)

A Reverse Proxy

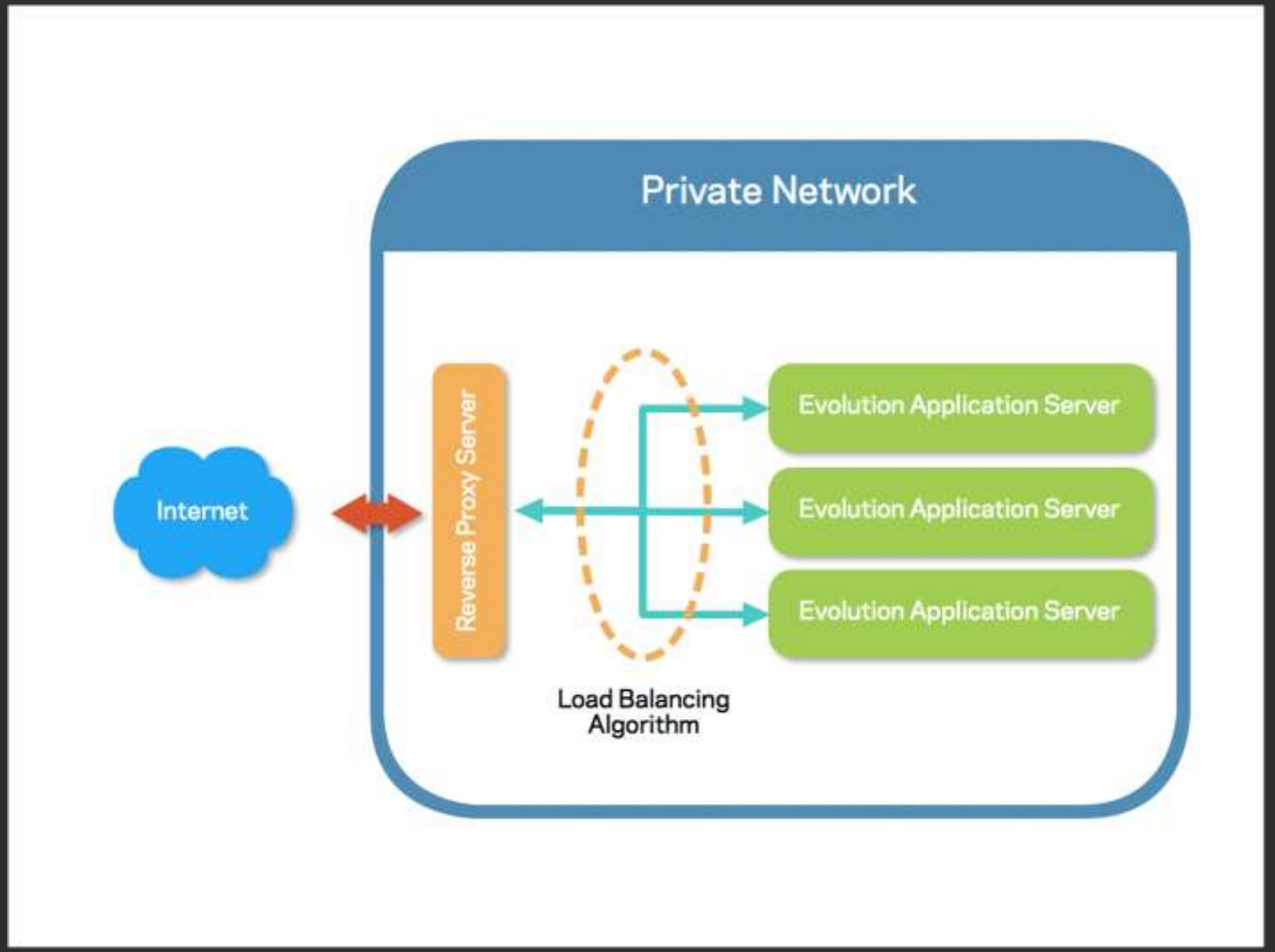
- 對客戶端隱藏伺服器（叢集）的 IP 位址
- 安全：作為應用層防火牆 (Web Application Firewall)，為網站提供對基於 Web 的攻擊行爲（例如 DoS/DDoS）的防護，更容易排查惡意軟體等
- 為後端伺服器（叢集）統一提供加密和 SSL 加速（如 SSL 終端代理）
- 負載均衡，若伺服器叢集中有負荷較高者，反向代理通過 URL 重寫，根據連線請求從負荷較低者獲取與所需相同的資源或備援
- 對於靜態內容及短時間內有大量存取請求的動態內容提供快取服務
- 對一些內容進行壓縮，以節約頻寬或為網路頻寬不佳的網路提供服務

反向代理 - 維基百科，自由的百科全書 (wikipedia.org)

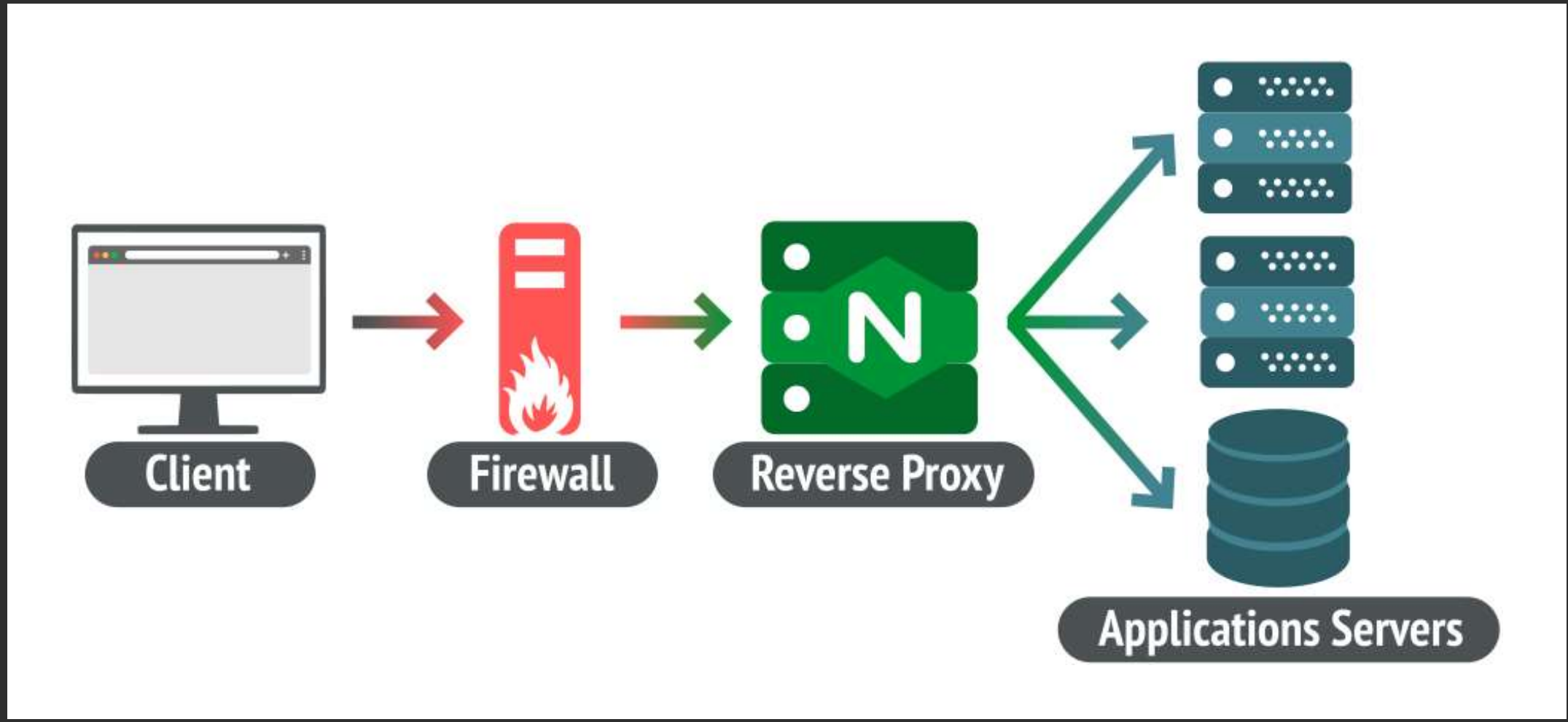
Reverse Proxy and (Forward) Proxy



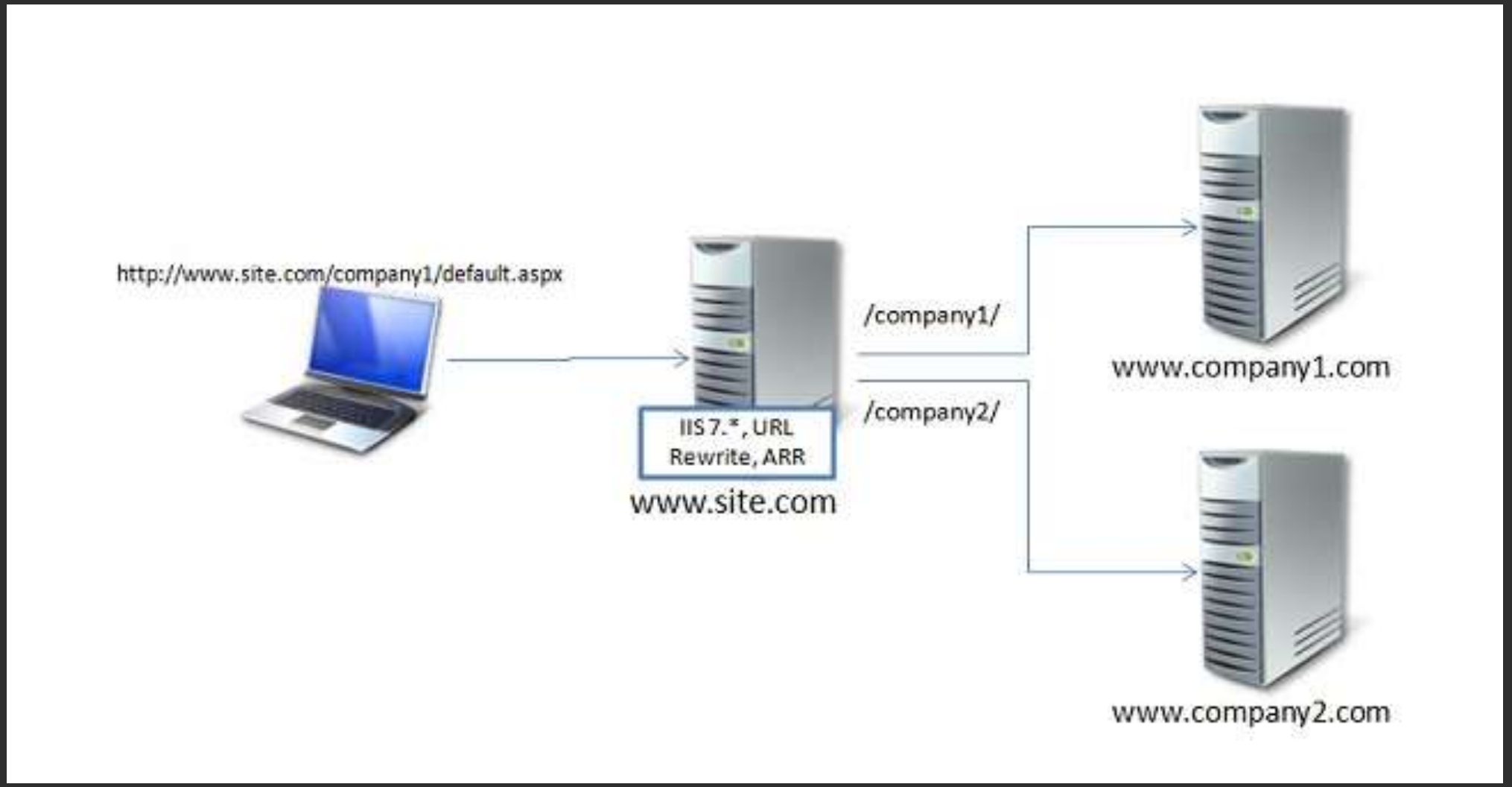
Reverse Proxy – Load Balancing



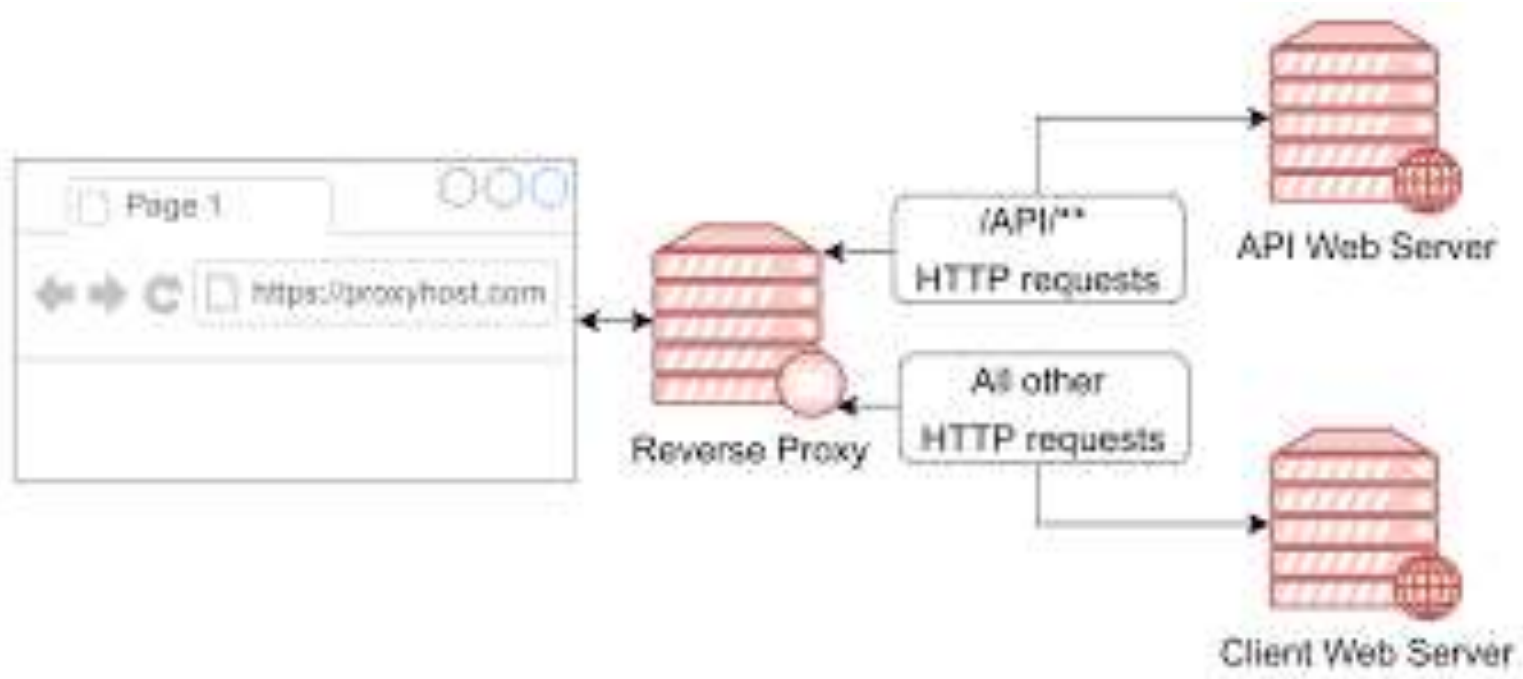
Reverse Proxy – Layers



Reverse Proxy – Path Routing



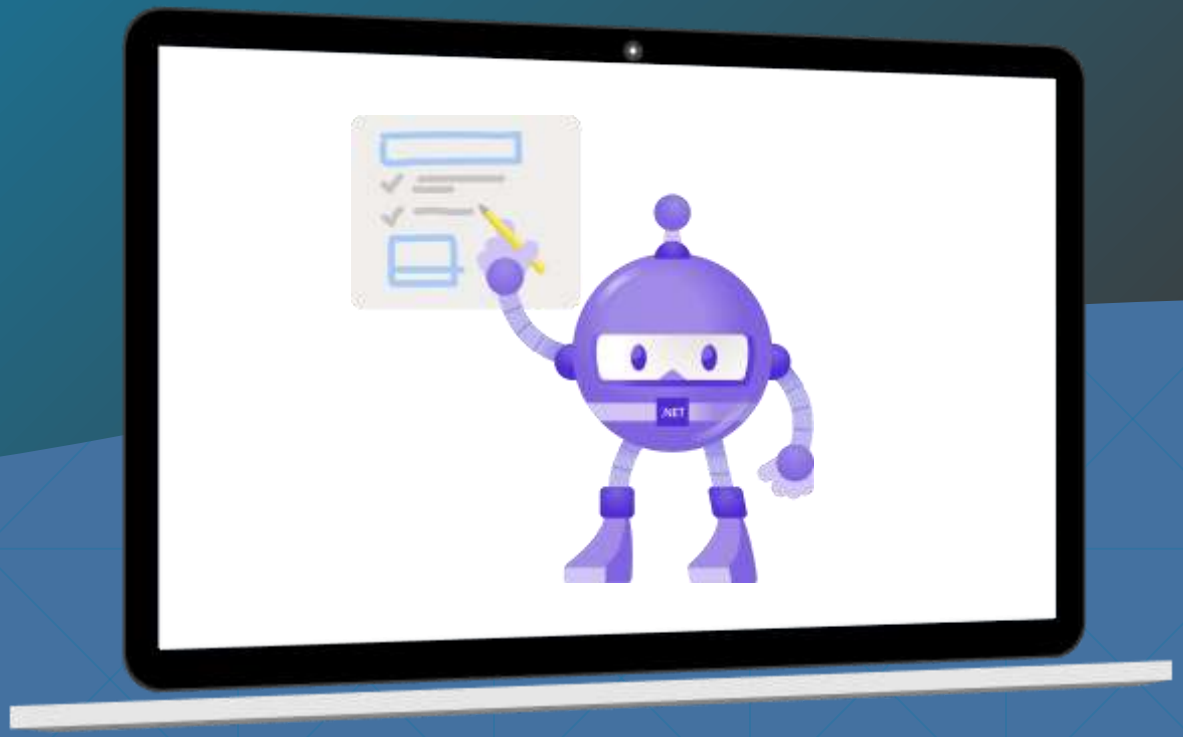
Reverse Proxy – Path Routing



Featured Proxy

- Proxy Based Web Application Firewall
- API Gateway pattern

Why YARP



歡迎參加我們的每週四固定聚會

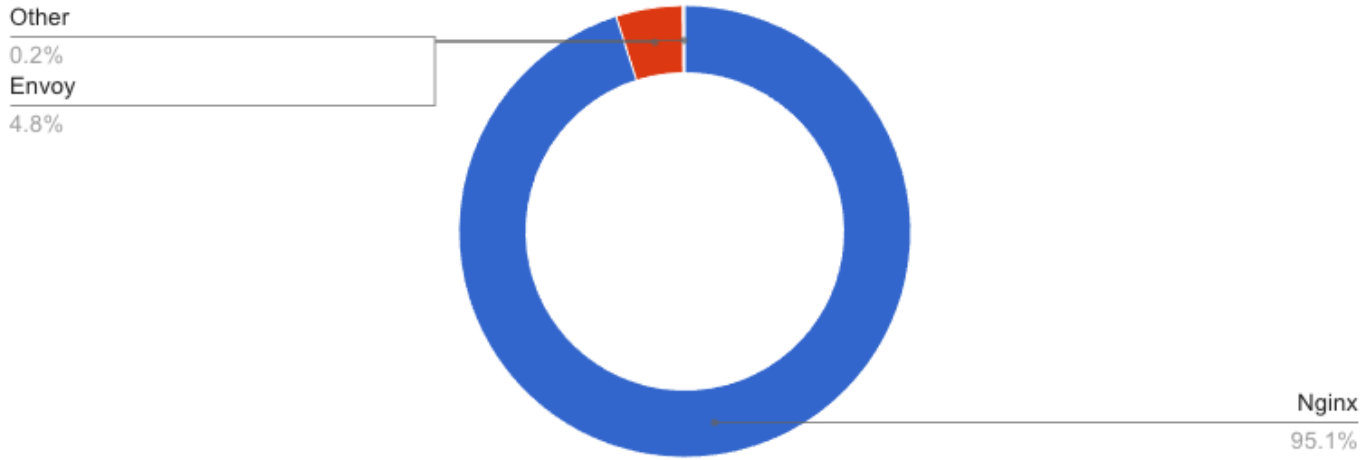
So.. 什麼是 YARP 對我來說有什麼意義

- Yet Another Reverse Proxy
- YARP 是一個反向代理工具包，運用 .NET (.NET Core) 建置高效能反向代理
- YARP 提供了兩個主要使用和自訂的方向，
  - 全功能代理
  - 高度自定義的環境，用戶可以直接呼叫 YARP 請求轉送程式，繞過路由和負載平衡等模組。開發者甚至可以在同一個程序中混合兩種方式，根據路由切換。

Why YARP?

- 也認識一下 Reverse Proxy 的強者們
  - NGINX
  - Envoy

Reverse Proxy market share



Technology	Websites tracked	Market share
Nginx	6,761,000	95%
Envoy	338,000	5%
F5 BigIP	9,700	0.1%
IBM DataPower	1,800	0.03%

[Reverse proxies market share, websites and contacts - Wappalyzer]

Reverse Proxy market share

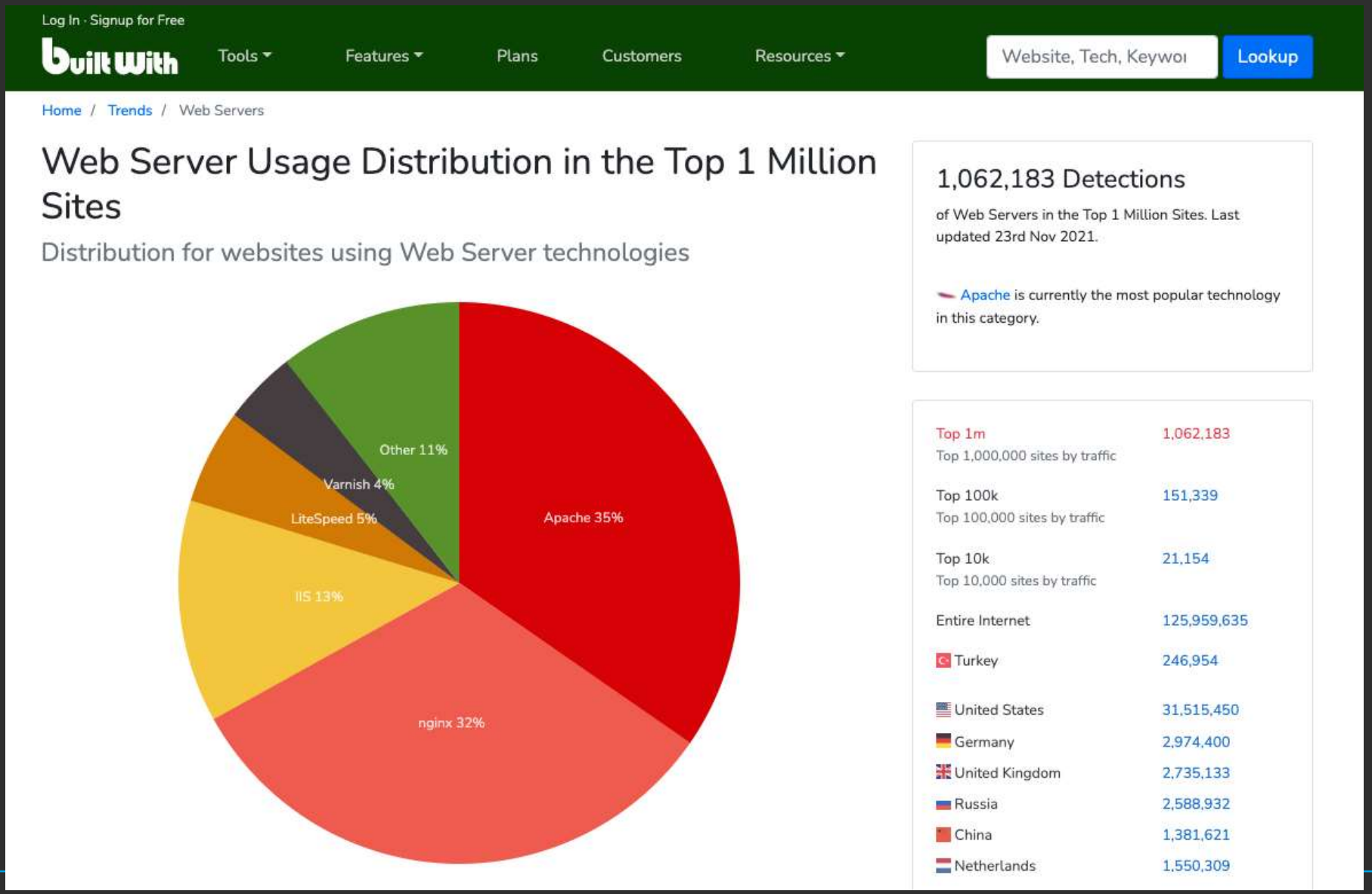
Envoy vs. Nginx

	Envoy	Nginx
Website	envoyproxy.io	nginx.org
Categories	Reverse proxies	Web servers, Reverse proxies
About	Envoy is an open-source edge and service proxy, designed for cloud-native applications.	Nginx is a web server that can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache.
Websites tracked	338,000	6,761,000
Pricing		

Envoy vs. Nginx feature and pricing comparison - Wappalyzer

tw MVC
https://mvc.tw

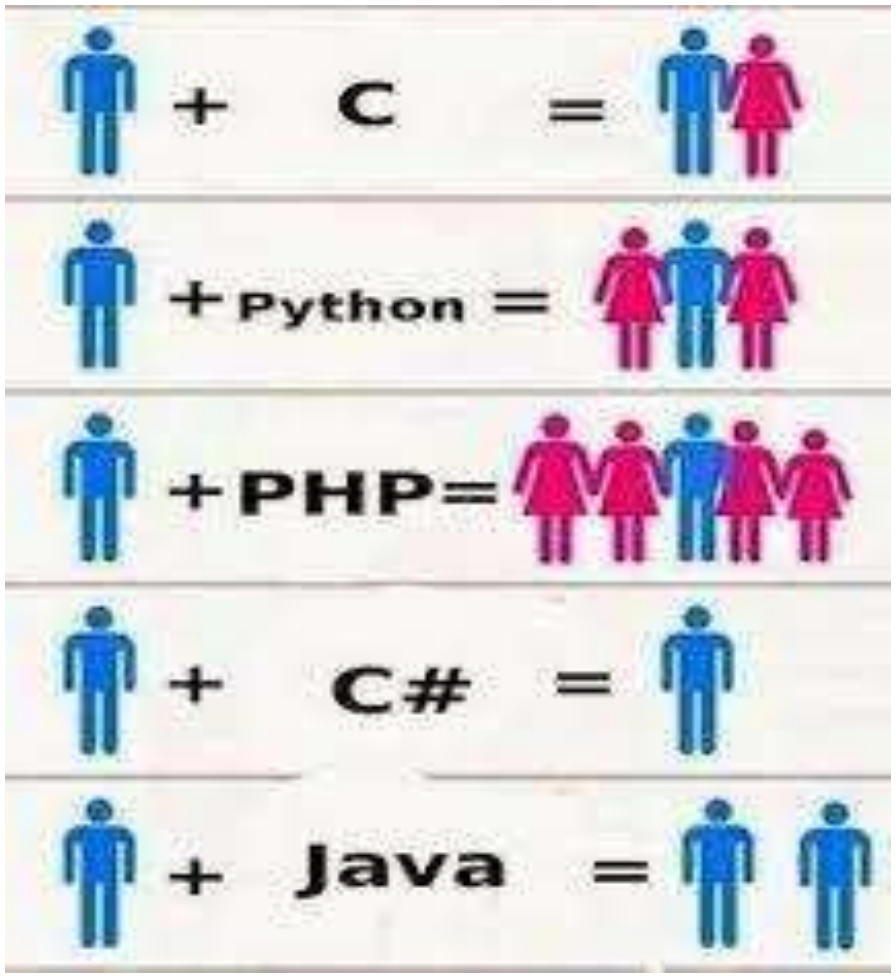
Web Server market share



[Web Server technologies Web Usage Distribution (builtwith.com)]



你别说话，PHP才是最好的语言





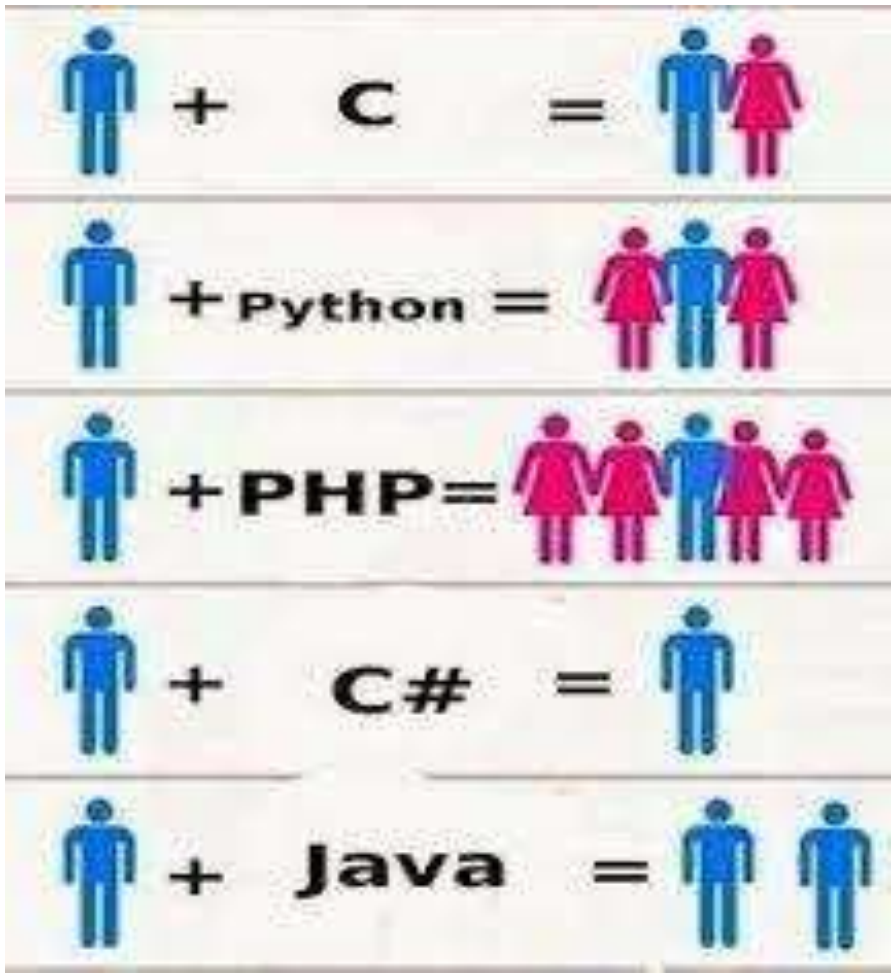
網路迷因，僅供參考

Bill …















網路迷因，僅供參考

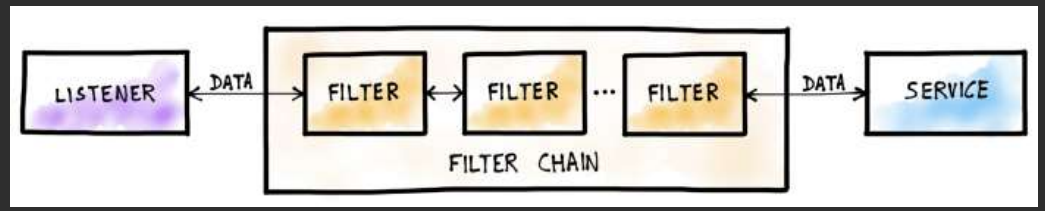
Reverse Proxy 的強者們

NGINX

- Reverse Proxy
- Web Server
- 客製化
  - NGINX + Lua
  - NGINX + JavaScript

Envoy

- Reverse Proxy
- 客製化
  - Lua and HTTP filter



TechEmpower

Web Framework Benchmarks

Introduction | Previous Rounds | Round 18 2019-07-09 | Round 19 2020-05-28 | **Round 20 2021-02-08** | Motivation & Questions | Environment | Source code & Requirements

In the following tests, we have measured the performance of several web application platforms, full-stack frameworks, and micro-frameworks (collectively, "frameworks").
For more information, read the introduction, motivation, and latest environment details.

2021-02-08
Round 20

T-cells that target Lethargic Software Syndrome
Give your next development project a shot of performance vaccine, keeping latency low and avoiding the pain of premature architectural complexity. No cold storage required.

View additional commentary about Round 20 at our blog.

Run details and logs: Physical | Cloud

Showing **243** of **247** frameworks. **Show filters panel**

Test types: **Composite scores** | **JSON serialization** | **Single query** | **Multiple queries** | **Cached queries** | **Fortunes** | **Data updates** | **Plaintext**

Hardware: **Physical** | **Cloud**

Plaintext

Best (bar chart) | Data table | Latency | Framework overhead

Best plaintext responses per second, Dell R440 Xeon Gold + 10 GbE (439 tests)

Rnk	Framework	Best performance (higher is better)		Errors	Cls	Lng	Plt	FE	Aos	IA
1	pico.v	7,017,566	100.0% (99.9%)	0	Mcr	v	Non	Non	Lin	Rea
2	aspcore	7,016,966	100.0% (99.9%)	0	Plt	C#	.NE	kes	Lin	Rea
3	wizzardo-http	7,016,349	100.0% (99.9%)	0	Mcr	Jav	Non	Non	Lin	Rea
4	ulib	7,016,097	100.0% (99.9%)	3	Plt	C++	Non	ULi	Lin	Rea
5	ulib-plaintext_fit	7,013,944	99.9% (99.9%)	0	Plt	C++	Non	ULi	Lin	Rea
6	gnet	7,010,982	99.9% (99.8%)	0	Plt	Go	Non	Non	Lin	Rea

YARP

- Web Server (Kestrel) and Reverse Proxy
- Fully customizable in C#

How to use YARP

1. dotnet new web -n {your-project-name}
2. dotnet add package Yarp.ReverseProxy

How to use YARP in .NET Core / .NET 5

```cs
public IConfiguration Configuration { get; }

public Startup(IConfiguration configuration)
{
    Configuration = configuration;
}

public void ConfigureServices(IServiceCollection services)
{
    // Add the reverse proxy to capability to the server
    var proxyBuilder = services.AddReverseProxy();
    // Initialize the reverse proxy from the "ReverseProxy" section of configuration
    proxyBuilder.LoadFromConfig(Configuration.GetSection("ReverseProxy"));
}

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    // Enable endpoint routing, required for the reverse proxy
    app.UseRouting();
    // Register the reverse proxy routes
    app.UseEndpoints(endpoints =>
    {
        endpoints.MapReverseProxy();
    });
}
```

Startup.cs

```json
{
 "Logging": {
   "LogLevel": {
     "Default": "Information",
     "Microsoft": "Warning",
     "Microsoft.Hosting.Lifetime": "Information"
   }
 },
 "AllowedHosts": "*",
 "ReverseProxy": {
   "Routes": {
     "route1" : {
       "ClusterId": "cluster1",
       "Match": {
         "Path": "{**catch-all}"
       },
     }
   },
   "Clusters": {
     "cluster1": {
       "Destinations": {
         "destination1": {
           "Address": "https://example.com/"
         }
       }
     }
   }
 }
}
```

appsetting.json

How to use YARP in .NET 6

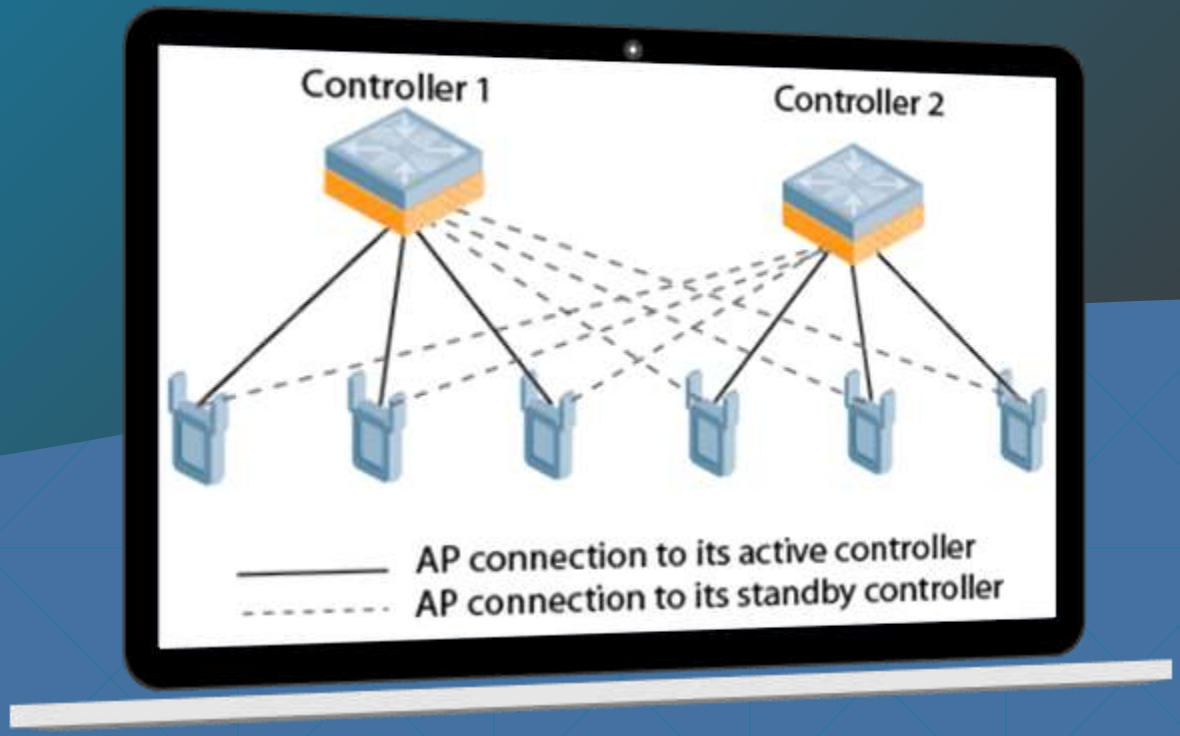
```
var builder = WebApplication.CreateBuilder(args);
builder.Services.AddReverseProxy()
    .LoadFromConfig(builder.Configuration.GetSection("ReverseProxy"));
var app = builder.Build();
app.MapReverseProxy();
app.Run();
```

Startup.cs

```
{
 "Logging": {
   "LogLevel": {
     "Default": "Information",
     "Microsoft": "Warning",
     "Microsoft.Hosting.Lifetime": "Information"
   }
 },
 "AllowedHosts": "*",
 "ReverseProxy": {
   "Routes": {
     "route1" : {
       "ClusterId": "cluster1",
       "Match": {
         "Path": "{**catch-all}"
       },
     }
   },
   "Clusters": {
     "cluster1": {
       "Destinations": {
         "destination1": {
           "Address": "https://example.com/"
         }
       }
     }
   }
 }
}
```

appsetting.json

High Availability



歡迎參加我們的每週四固定聚會

Use YARP for Load Balancing

```
"ReverseProxy": {
    "Clusters": {
        "cluster1": {
            "LoadBalancingPolicy": "RoundRobin",
            "Destinations": {
                "cluster1/destination1": {
                    "Address": "https://localhost:10000/"
                },
                "cluster1/destination2": {
                    "Address": "https://localhost:10010/"
                }
            }
        }
    }
}
```

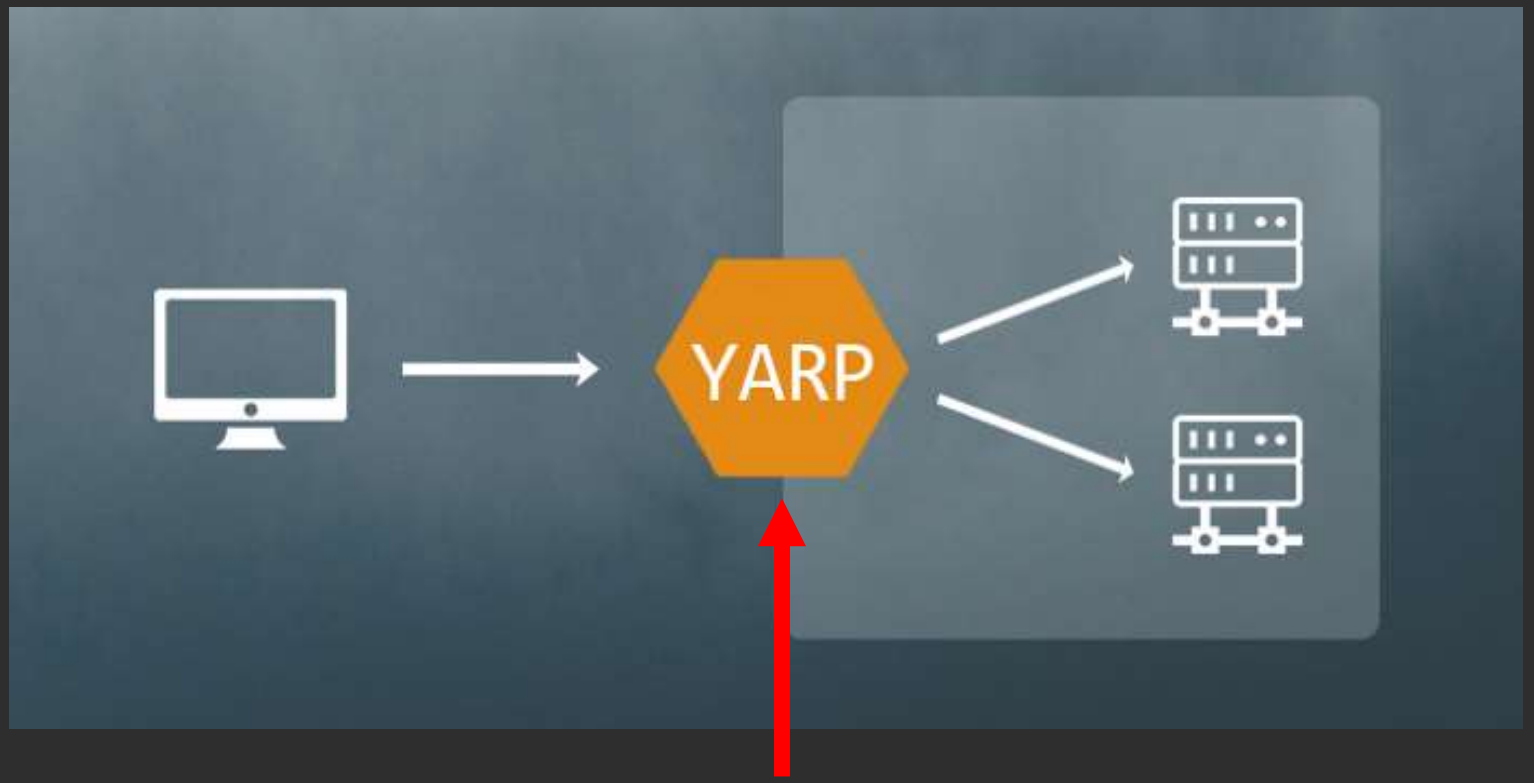
Policies

- FirstAlphabetical
- Random
- PowerOfTwoChoices (default)
- RoundRobin
- LeastRequests
- Customized ILoadBalancingPolicy

Load Balancing

- Session Affinity
- Load Balance Policy
- Destinations Health Checks

SPOF: Single Point of Failure



單點故障 - 維基百科，自由的百科全書 (wikipedia.org)

Use Cases

Classic ASP 轉圈圈



tw MVC

歡迎參加我們的每週四固定聚會

處理問題

- ASP/COM+ 舊系統，用戶數量龐大
- 使用者回報，有若干頁面操作會有長時間的後端查詢
- 因為使用了 session，這一頁查詢發起後，該用戶隨後的所有系統操作都只會轉圈圈，而且永遠都不會出結果
- 能不能避免轉圈，讓用戶空等

Startup.cs

ConfigureServices

```
0 references
public void ConfigureServices(IServiceCollection services)
{
    var proxyBuilder = services.AddReverseProxy();
    proxyBuilder.LoadFromConfig(Configuration.GetSection("ReverseProxy"));

    services.AddTransient<ConnectionMultiplexer>(x =>
    {
        if (connectionMultiplexer == null || !connectionMultiplexer.IsConnected)
        {
            var connectionString = Configuration.GetValue<string>("RedisConnection");
            var connectionOptions = ConfigurationOptions.Parse(connectionString);
            connectionOptions.AbortOnConnectFail = false;
            connectionMultiplexer = ConnectionMultiplexer.Connect(connectionOptions);
        }

        return connectionMultiplexer;
    });

}
You, 2 hours ago · initial
0 references
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
```

```
        }

        return connectionMultiplexer;
    });

}
You, 2 hours ago · initial
0 references
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    app.UseRouting();
    app.UseEndpoints(endpoints =>
    {
        endpoints.MapReverseProxy(proxy =>
        {
            proxy.UseMiddleware<AvoidReentryMiddleware>();
        });
    });
}
```

Startup.cs

Configure

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "ReverseProxy": {
    "Routes": {
      "route1": {
        "ClusterId": "cluster1",
        "Match": {
          "Path": "{**catch-all}"
        }
      }
    },
    "Clusters": {
      "cluster1": {
        "Destinations": {
          "cluster1/destination1": {
            "Address": "https://stackoverflow.com/"
          }
        }
      }
    }
  },
  "RedisConnection": "localhost:6379"
}
```

appsettings.json

```
public Task Invoke(HttpContext context)
{
    var request = context.Request;

    if (request.Method ≠ "POST"
        || !request.Cookies.ContainsKey(SessionKey)
        || request.Headers.ContainsKey("X-Requested-With"))
        return _next.Invoke(context);

    var redis = context.RequestServices.GetRequiredService<ConnectionMultiplexer>();
    var db = redis.GetDatabase();
    var key = $"Reentry:{request.Cookies[SessionKey]}";
    var previous = db.StringGet(key);

    if (!string.IsNullOrEmpty(previous))
    {
        var previousStatus = JsonSerializer.Deserialize<ExecutionStatus>(previous);
        var content = $@"
        <html>
            <body>
                <h1>前次請求執行中，請稍後再重試！</h1>
                <ul>
                    <li>Path: {previousStatus.Path}</li>
                    <li>Time: {previousStatus.DateTime}</li>
                </ul>
                <script> setTimeout(function () {{ history.go(-3); }}, 5000); </script>
            </body>
        </html>";
        context.Response.ContentType = "text/html; charset=utf-8";
        context.Response.WriteAsync(content);
        return Task.CompletedTask;
    }

    var status = new ExecutionStatus { Path = request.Path.Value, DateTime = DateTime.UtcNow };
    db.StringSet(key, JsonSerializer.Serialize(status));
    db.KeyExpire(key, TimeSpan.FromSeconds(30));
    return _next.Invoke(context).ContinueWith(t => db.KeyDelete(key));
}
```

AvoidReentryMiddleware

Use Cases

Realtime Visit Counter
Millisecond-level Scaling

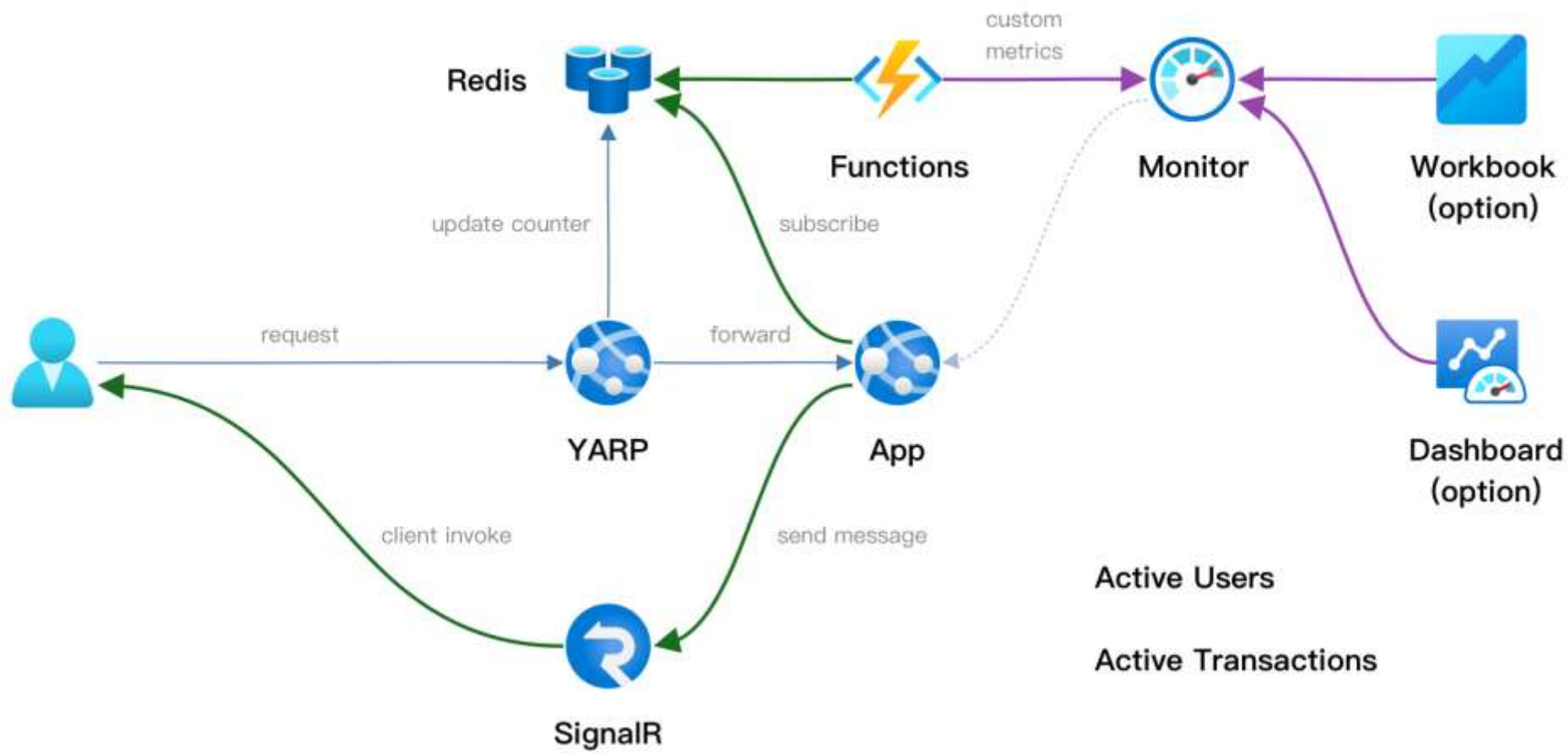


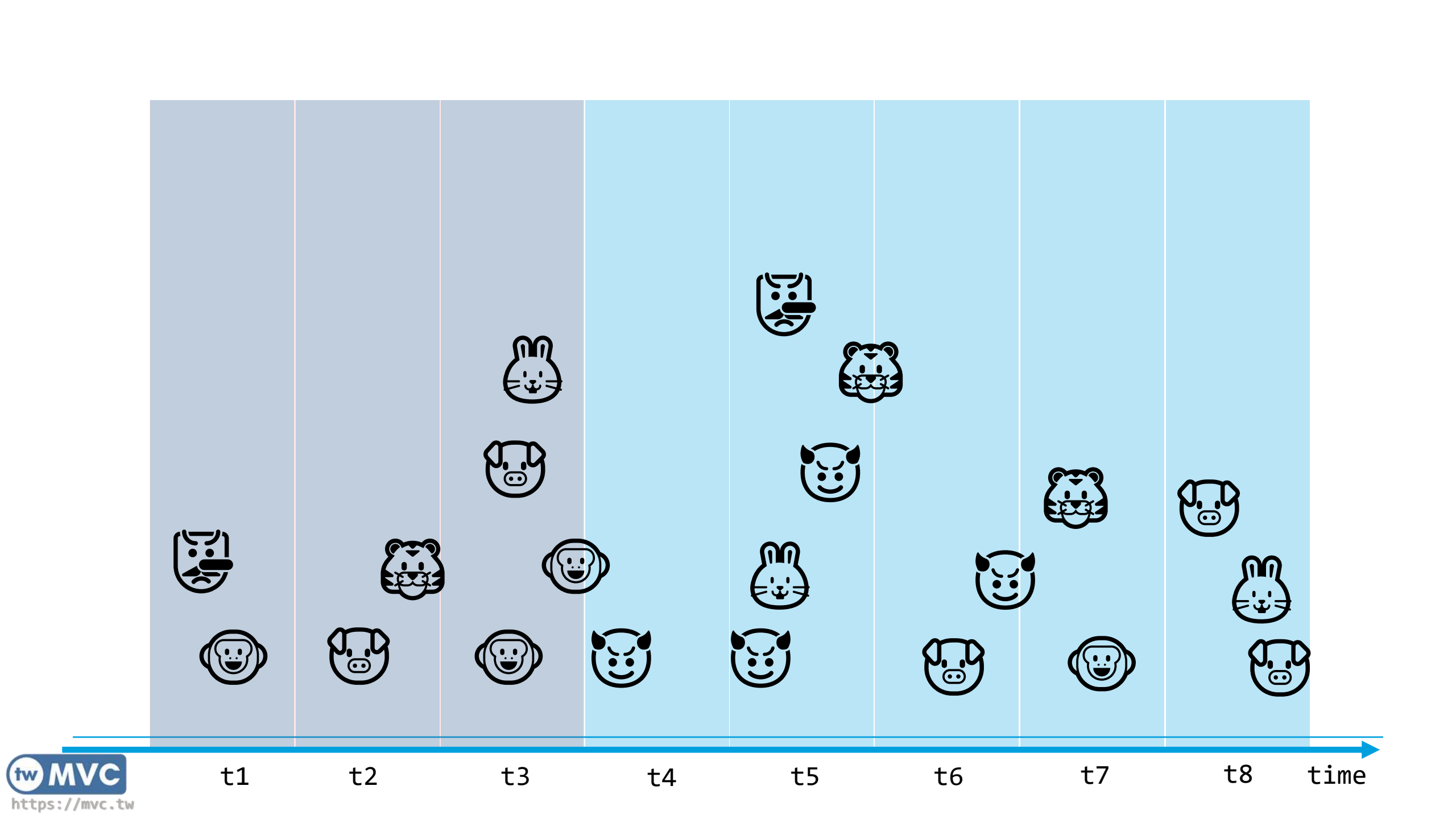
歡迎參加我們的每週四固定聚會

處理問題

- 解決電子商務微秒級
  - 線上使用人數統計（特定部位）
  - Cloud Scaling

Architecture





tw MVC
https://mvc.tw

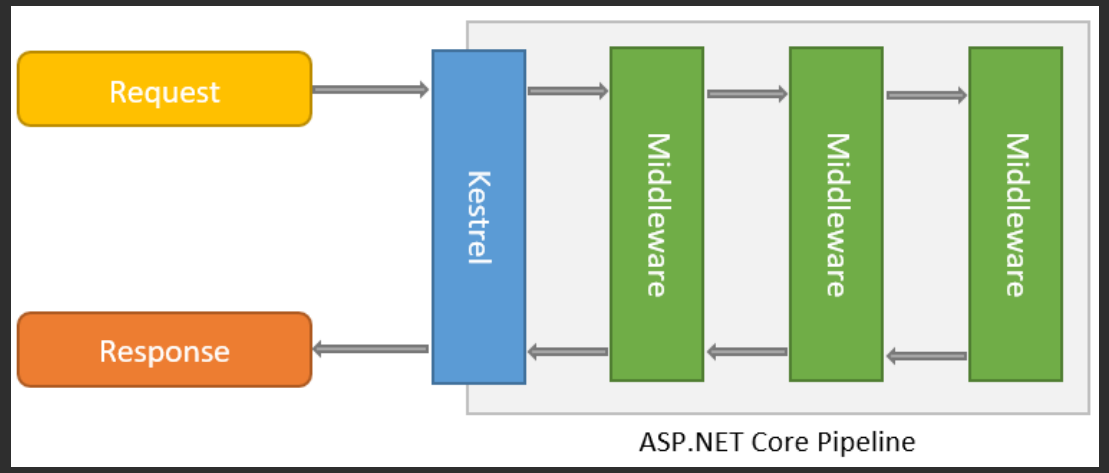
Configuration

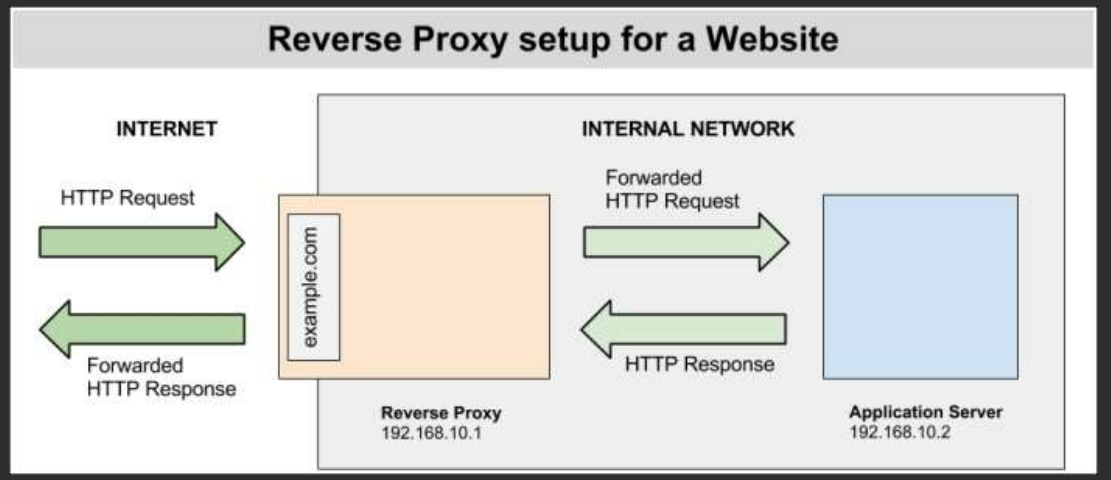
- 計算 20 秒內結帳的人數
- 計算 5 秒內首頁的訪問人數
- …

```
    }
  },
  "CounterSpecs": [
    {
      "Name": "checkout_20s",
      "Pattern": "^/checkout/$",
      "Seconds": 20
    },
    {
      "Name": "home_5s",
      "Pattern": "^/$",
      "Seconds": 5
    }
  ],
  "RedisConnection": "localhost:6379"
}
```

一定要使用 Reverse Proxy 嗎？

- 評估功能切入點，Middleware or Proxy

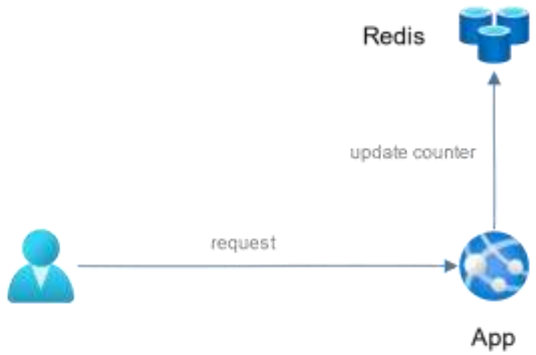




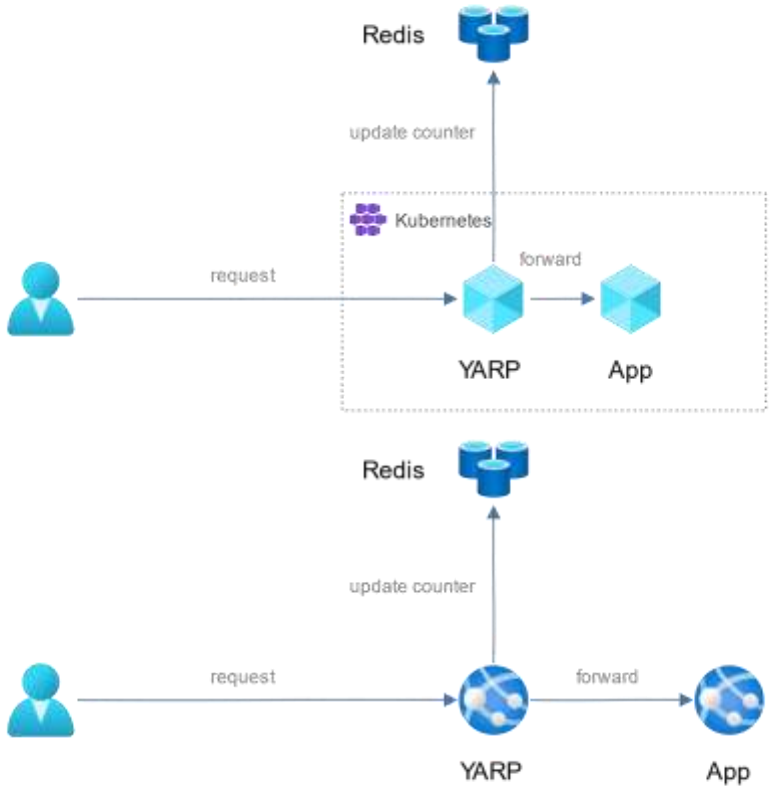
Java Spring、PHP Laravel、Python Flash、ASP.NET 都有 middleware 的設計

一定要使用 Reverse Proxy 嗎？

- Approach 1: Middleware



- Approach 2: Proxy



Resources

- YARP Documentation (microsoft.github.io)
- microsoft/reverse-proxy (github.com)
- YARP: The .NET Reverse proxy - YouTube

Blog 是記錄知識的最佳平台



https://dotblogs.com.tw

SkillTree 為了確保內容與實務不會脫節，我們都是聘請企業顧問等級並且目前依然在職場的業界講師，我們不把時間浪費在述說歷史與沿革，我們並不是教您考取證照，而是教您如何上場殺敵，拳拳到肉的內容才是您花錢想要聽到的，而這也剛好是我們擅長的。



https://skilltree.my

天瓏資訊圖書

