Running .NET on Cloud Foundry

Windows vs. Linux
Zach Brown  
works at **Pivotal**. 
twitter: @morede部署s

Rita Zhang  
works at **Microsoft**. 
twitter: @ritazzhang

* About the Speakers
Two ways to run ASP.NET apps on CF

1. Windows 2012 R2 Stack
   • Support Introduced in Nov. 2015
   • Supports SDK 3.5-4.5+
   • Requires Diego-Windows, Garden-Windows
   • BOSH-Windows currently underway

2. Linux Stack
   • ASP.NET Core only
   • Standard CF Ubuntu Stack
   • Community ASP.NET 5 Buildpack
## Stack Comparison Matrix

<table>
<thead>
<tr>
<th>Stack Name</th>
<th>windows2012R2</th>
<th>cflinuxfs2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF Runtime</td>
<td>Diego Only</td>
<td>Diego or DEA</td>
</tr>
<tr>
<td>Buildpack</td>
<td>binary_buildpack (App is pre-compiled, no detect)</td>
<td>ASP.NET 5 buildpack</td>
</tr>
<tr>
<td>CLR Version</td>
<td>.NET 3.5-4.5+</td>
<td>Core CLR</td>
</tr>
<tr>
<td>Container</td>
<td>Garden-Windows</td>
<td>Garden-Linux</td>
</tr>
<tr>
<td>Developer OS</td>
<td>Windows</td>
<td>Win, Mac OSX, Linux</td>
</tr>
</tbody>
</table>
**Containers**

Garden-Linux vs. Garden-Windows

- Garden-Linux containers are based on LXC (Linux cgroups)
- Win 2k12 provides process isolation primitives similar to those found in the Linux kernel
- IronFrame abstracts these primitives into an API utilized by Garden-Windows
- Win 2k12’s low-level API ≈ Linux kernel’s API
- Therefore Garden-Windows process isolation functionality ≈ Garden-Linux
**Containers: Garden-Linux**

<table>
<thead>
<tr>
<th>Filesystem isolation</th>
<th>An app cannot read or modify files outside its own container filesystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk usage</td>
<td>Enforced using Disk Quotas</td>
</tr>
<tr>
<td>Network isolation</td>
<td>Ports available to processes within a container are namespace to the container. Other containers on the same host can bind to the same ports</td>
</tr>
<tr>
<td>Memory usage</td>
<td>One application cannot use more than a preset amount of memory on a single VM host</td>
</tr>
</tbody>
</table>
## Containers: Garden-Windows

| Filesystem isolation | Uses Windows Accounts and ACLs  
|                      | Files in c:\containerizer are private to container  
|                      | C:\Program Files, system DLLs, etc. are visible to all containers on host |
| Disk usage          | Enforced by NTFS disk quotas |
| Network isolation   | Applications bind directly to the external IP of the cell.  
|                      | Internal container ports must be mapped to external ports. |
| Memory usage        | Uses Job Object process isolation + the “Guard”  
|                      | Guard monitors app memory usage and kills jobs if mapped memory > allocated memory  
|                      | Guard also enforces no process can be launched outside the job object process tree |

More Info: Garden-Windows Deep-dive

ASP.NET Core vs ASP.NET 4.6

• ASP.NET Core
  • Web API
  • ASP.NET MVC
  • No Web Forms (yet)
  • No SignalR

• ASP.NET 4.6
  • Battle-tested, hardened
  • Many years of testing and improvement
  • MVC, Web Forms, SignalR, etc.
*When to Choose Which*

- Greenfield Applications
  - Try “Core First” approach
- Replatforming/Migrating Existing Apps
  - Probably ASP.NET 4.x
- Dependencies and framework requirements will likely dictate choice
* Migrating existing applications

Non Cloud-Native stuff you’ll want to change:
• Reading/Writing to the Registry
• Reading/Writing to the local disk
• Integrated Windows Auth
  • Replace with ADFS/OAuth2 to leverage UAA
• In-Process Session State / Sticky Sessions
  • Replace InProc, StateServer with out-of-process (e.g. Redis, SQL Server)
• Environment-specific config in web.config
  • Externalize into VCAP environment variables
• MSI-Installed services or drivers
  • Bin deploy dependencies with app
Deploying a .NET application to Windows

*cf push mydotNetApp -s windows2012R2 -b binary_buildpack*

Specify the stack, and the binary bp
cf push mydotNetApp
    -s cflinuxfs2
    -b <asp.net5 bp>

Specify the ASP.NET Core bp

* Deploying a .NET Core application to Linux
This is where we show you something cool
• .NET Core 1.0 GA
• Supported ASP.NET Core Buildpack
• BOSH-Windows [1]
  BOSH-managed CF Win2012R2 Hosts
• Steel Toe [2]
  Use .NET with Spring Cloud Config Server, Netflix Eureka, etc.

• DiegoMVC Demo
  https://github.com/ruurdk/DiegoMVC

• ASP.NET Core Demo
  https://github.com/ritazh/aspnetcoreapp-cf

• ASP.NET 5 Buildpack
  https://github.com/cloudfoundry-community/asp.net5-buildpack

• Diego-Windows
  https://github.com/cloudfoundry/diego-windows-release

• Garden-Windows
  https://github.com/cloudfoundry/garden-windows

• Steel Toe
  https://github.com/SteelToeOSS