Evergreen Citrix Builds

virtualengine

The End User Computing and Cloud Solution Specialists

Iain Brighton

- CTO @ Virtual Engine
  - End-User Computing architect/Trainer
  - Active community member
- Amateur developer
  - Virtual Engine Toolkit (VET)
  - PScribo
  - Lability

Agenda

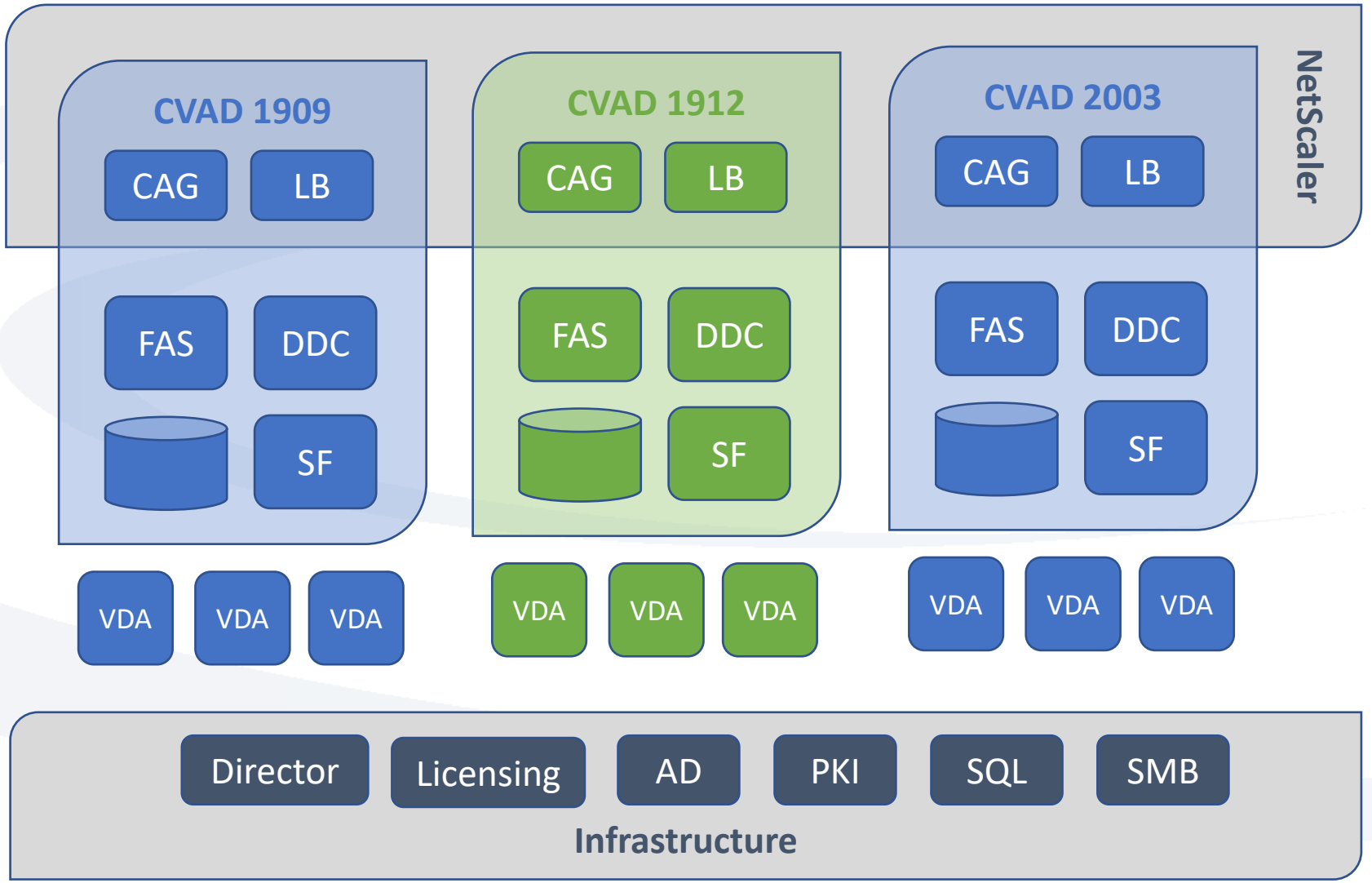
- What, why and why not of Evergreen
- Configuration-as-code
- PowerShell DSC 101
- Demo
- Wrap up

LTSR and/or CR

- Bug fixes = LTSR and CR
- New features and enhancements = CR only
- LTSR customers may need to install latest CU to receive a fix
- CR customers may need to upgrade to next CR to receive a fix
- Cloud deployments not supported by CR

What do we mean by “Evergreen”?

- Always deploying “new”
- Never upgrading
- Continual deployments
  - Build
  - Test
  - Migrate
  - Tear down


NetScaler
CVAD 1909
CAG
LB
FAS
DDC
SF
VDA
VDA
VDA
CVAD 1912
CAG
LB
FAS
DDC
SF
VDA
VDA
VDA
CVAD 2003
CAG
LB
FAS
DDC
SF
VDA
VDA
VDA
Director
Licensing
AD
PKI
SQL
SMB
Infrastructure

Why go to all this trouble?

- Known, repeatable deployment processes
  - Deploy to any infrastructure (migration and/or DR)
- Reduce upgrade risks
- Minimise configuration drift
- Change tracking
- Documentation

So why wouldn't you?

- Management paradigm shift
- Requires upfront and continual investment
- Additional infrastructure components requirements
- Data islands (lost visibility)
- Requires good automation skills
- Not everything is a good "fit"

How do we do it?

- Configuration-as-Code
  - Separation of **what** from **how**
- Windows PowerShell DSC 2.0
  - Improvements introduced in WMF v5
  - Cross-node dependencies
  - PSDSCRunAsCredential (no CredSSP)
- Can use any configuration management tooling
  - Chef, Puppet and Ansible all natively support DSC resources

Windows PowerShell DSC 101

- Local Configuration Manager (LCM)
  - Push/Pull
  - Configuration Mode
- Managed Object Format (MOF) files
  - CIM standard
  - Enacted by the LCM
- Resources
  - Open source resources (inc. XenDesktop7)
  - Internal/custom resources
- Configurations
  - Configuration Data

PowerShell Authoring

```
#requires -version 4

configuration DemoConfiguration {
    param (
        [System.String[]] $ComputerName
    )

    node $ComputerName {
        File Reports {
            DestinationPath = 'C:\Reports';
            Ensure = 'Present';
            Type = 'Directory';
        }
        Service WindowsUpdate {
            Name = 'wuauserv';
            StartupType = 'Automatic';
            State = 'Running';
        }
        WindowsFeature RemoteDesktopServices {
            Name = 'RDS-RD-Server';
            Ensure = 'Present';
            DependsOn = '[Service]WindowsUpdate';
        }
    } #end node

} #end configuration
```

```
...

instance of MSFT_ServiceResource as $MSFT_ServiceResource1ref
{
ResourceID = "[Service]WindowsUpdate";
 State = "Running";
 SourceInfo = “C:\\TestLab\\Resources\\DSCConfigurations\\..
 Name = "wuauserv";
 StartupType = "Automatic";
 ModuleName = "PSDesiredStateConfiguration";
 ModuleVersion = "1.0";
};

instance of MSFT_RoleResource as $MSFT_RoleResource1ref
{
ResourceID = "[WindowsFeature]RemoteDesktopServices";
 Ensure = "Present";
 SourceInfo = "C:\\TestLab\\Resources\\DSCConfigurations\\..
 Name = "RDS-RD-Server";
 ModuleName = "PSDesiredStateConfiguration";
 ModuleVersion = "1.0";
DependsOn = {
    "[Service]WindowsUpdate"};
};

...
```

Demo

Where next?

- Infrastructure-as-code
  - ARM/CloudFormation templates or Terraform
- Build and release pipelines
- Infrastructure test automation

Resources

- Advanced PowerShell DSC and Custom Resources
  - Channel9 Series (Jeffrey Snover and Jason Helmick)
- XenDesktop7 DSC Resources
  - 44 resources
- PowerShell DSC Community Resources
  - Experimental and high-quality resources
- Lability
  - Locally deploy/test DSC configurations on Hyper-V

Questions?

@iainbrighton
iain.brighton@virtualengine.co.uk