Puppet and Azure: bringing DevOps to the enterprise

Microsoft

Puppet on Azure

What is Puppet? : Puppet is a configuration management system that allows you to define the state of your IT infrastructure, then automatically enforces the correct state.

Key Concepts:

- Supports easy to read declarative language.
- Enforces desired state on the system.
- Puppet Forge supports many ready to use modules.

A new approach to IT automation

Puppet Labs

- Award winning open source IT automation company
  - Puppet open source
  - Puppet enterprise

Quick stats

- 2005 founded
- ~9 million downloads in last 12 months
- ~10 million total nodes under management
- 80,000+ registered users
- ~60,000 nodes in largest deployments

puppet labs

Why Puppet Labs + Windows Azure?

Microsoft

- Serves the developer and the IT pro

Puppet Labs

- Automation software that enables the operational agility and efficiency of cloud computing at enterprise-class service levels

Microsoft Open Technologies, Inc.

- Interoperability through open source and open standards
- Wholly owned subsidiary of Microsoft
  - Independent cadence enables a closer interaction with open source partners

Windows Azure is the enterprise cloud

- DevOps is an important part of the Azure story

Microsoft Open Technologies

Configuration management

DEFINE

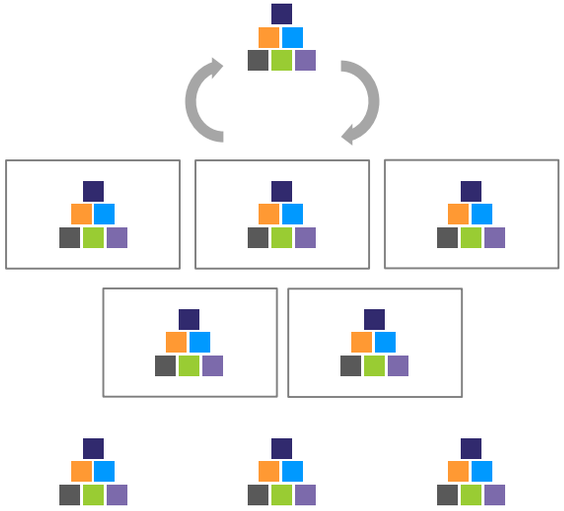
"Ensure Apache is installed, configured, and running"

COMPOSE

"Ensure a LAMP stack on top of RHEL"

AUTOMATE

"Stand-up a LAMP-on-RHEL stack on 100 nodes, and then enforce configurations"



Puppet Terminology

Manifest : A file containing code written in the Puppet language,& managing infrastructure and applications. Manifest contains resources and classes.

Resources : A unit of configuration, whose state can be managed by Puppet.

Module : A collection of classes, resource types, files, and templates, organized around a particular purpose.

Catalog : A catalog is a document that describes the desired system state for one specific computer. Catalogs are compiled from manifests by a puppet master server and served to agent nodes.

developer workflow

Puppet Options for Azure

Provisioning using Puppet Azure module

Bootstrapping puppet agent using Puppet Extension

Leveraging **DSC resources** using **Puppet**

WORKFLOW

Managing Infrastructure

↓

Bootstrap Agents

↓

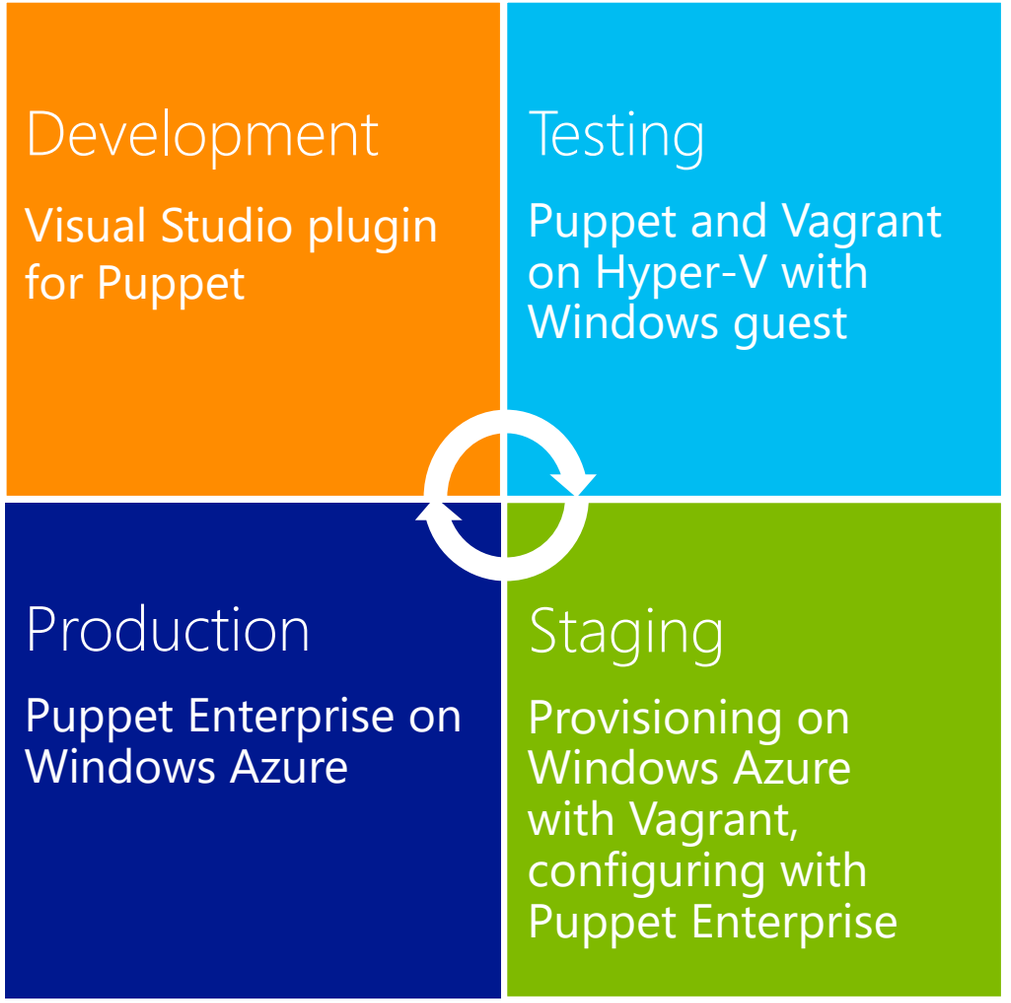
Customize VM

Why enterprise needs to change

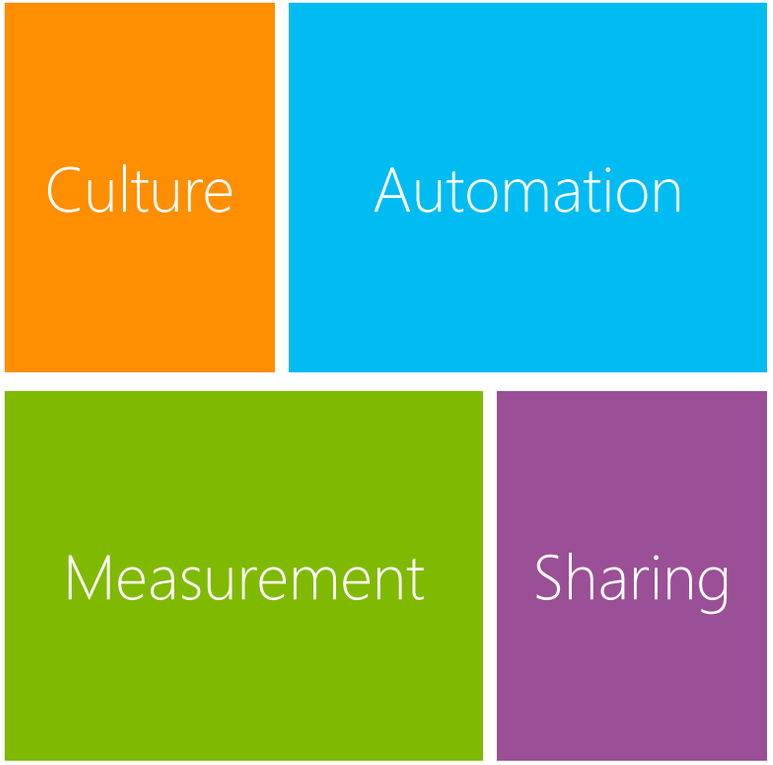
Agility	60%	of IT managers are not satisfied with the speed at which IT responds to business needs
Reliability	$72,000/hr.	cost of downtime due to manual errors and configuration drift
Productivity	48%	of IT professionals spend 50% or more of their time on basic administrative tasks
Shadow IT	36%	of employees have already used “unapproved” cloud services
Insight	93%	of IT professionals cannot answer “What changed?” when an outage incident occurs

Sources: Gartner, Kaseya, Harvey Nash, Vanson Bourne, Evolven, InformationWeek

Automation scenarios



How does a DevOps approach help?



A loose collection of practices

- Approach as a whole is mature
- Benefits of “DevOps” generally well understood
- Tools are maturing

Puppet Labs automate IT infrastructure for sysadmins

Treats infrastructure-as-code to define and automate systems and applications

Application Stacks

APPLICATION CODE & DATA

Java

C#

MIDDLEWARE

JBoss

Microsoft .NET

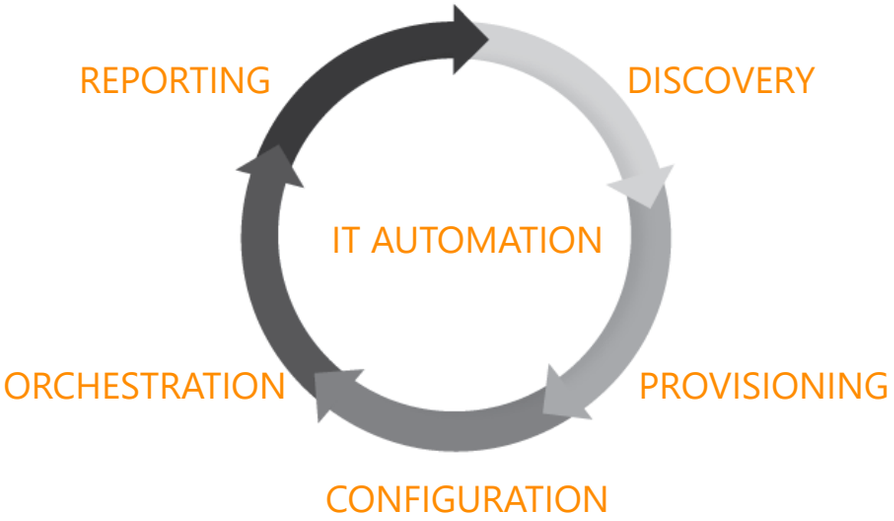
OPERATING SYSTEMS

PHYSICAL & VIRTUAL NODES

Hyper-V

Windows Azure

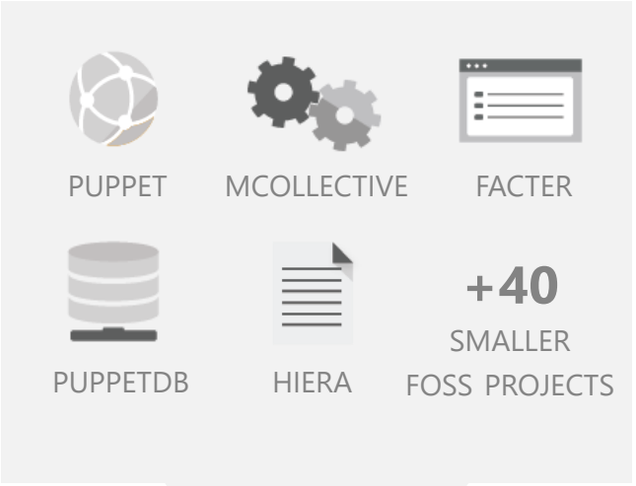
IT Management Stack



Monitoring	Help Desk
Asset Management	Service Catalog
Billing	Capacity Planning

Innovation & reliability: open source & Puppet Enterprise

Upstream Open Source Projects



Environment for nurturing innovation

- 7,000+ members
- Latest technologies
- Rapid release cycles
- ~1000s of changes/week
- Community support (IRC, forums)

Commercial Product

Puppet Enterprise

IT automation for business-critical apps

- Commercial-only functionality
- Single integrated solution
- Graphical User Interface
- Installer & upgrader
- QA'd & security hardened
- Performance tuning
- API guarantees
- Support & maintenance
- Training & services

PuppetForge: pre-built automation solutions

Operating System Resources

Applications

Apache
Windows Server
Apache Tomcat
Nagios
MySQL
splunk>
Java
WordPress

Virtual & Cloud Infrastructure

Windows Azure
openstack
KVM
Xen

Network & Storage Devices

cisco
Juniper Networks
NetApp
f5

Complementary tooling

- Puppet Enterprise
- Vagrant (windows guest)
- Vagrant-Azure
- Visual Studio
- Visual Studio Online
- Travis CI
- Chocolatey
- Nagios

Microsoft
Open Technologies

Puppet Labs + Microsoft
Leveling up enterprise IT

Microsoft Open Technologies

- An agile gateway to Microsoft engineering excellence

Windows Azure

- An enterprise cloud

Puppet Labs

- Reducing the friction of technological change

Install a Puppet Azure Module

Puppet Commands for Azure VM provisioning

puppet module install msopentech-microsoftazure

```
puppet azure_vm create \

--management-certificate pem-or-pfx-file-path \
--azure-subscription-id=your-subscription-id \
--image b39f27a8b8c64d52b05eac6a62ebad85__Ubuntu-13_04-
amd64-server-20130501-en-us-30GB \
--location 'west us' \
--vm-name vmname \
--vm-user username \
--password ComplexPassword \
--puppet-master-ip yourPuppetMasterIPAddress
```

Install Puppet Master

Step1 : Setting up Puppet Master:

- Select the Puppet Master Image from Azure Gallery . Note : Provide a lowercase cloud service & vm name for the Puppet Master.

- Add the following end points on the Puppet Master VM:

  443 for HTTPS
  8140 for Puppet
  61613 for MCollective

- Browse to the Puppet master VM and retrieve the user name and password for PE console:

  User login is located in the file /etc/puppetlabs/installer/answers.install (admin@<VM name>.cloudapp.net)
  Password is located in the file /etc/puppetlabs/installer/database_info.install

- Wait for 10 mins and login to PE console : https://<cloudservicename>.cloudapp.net

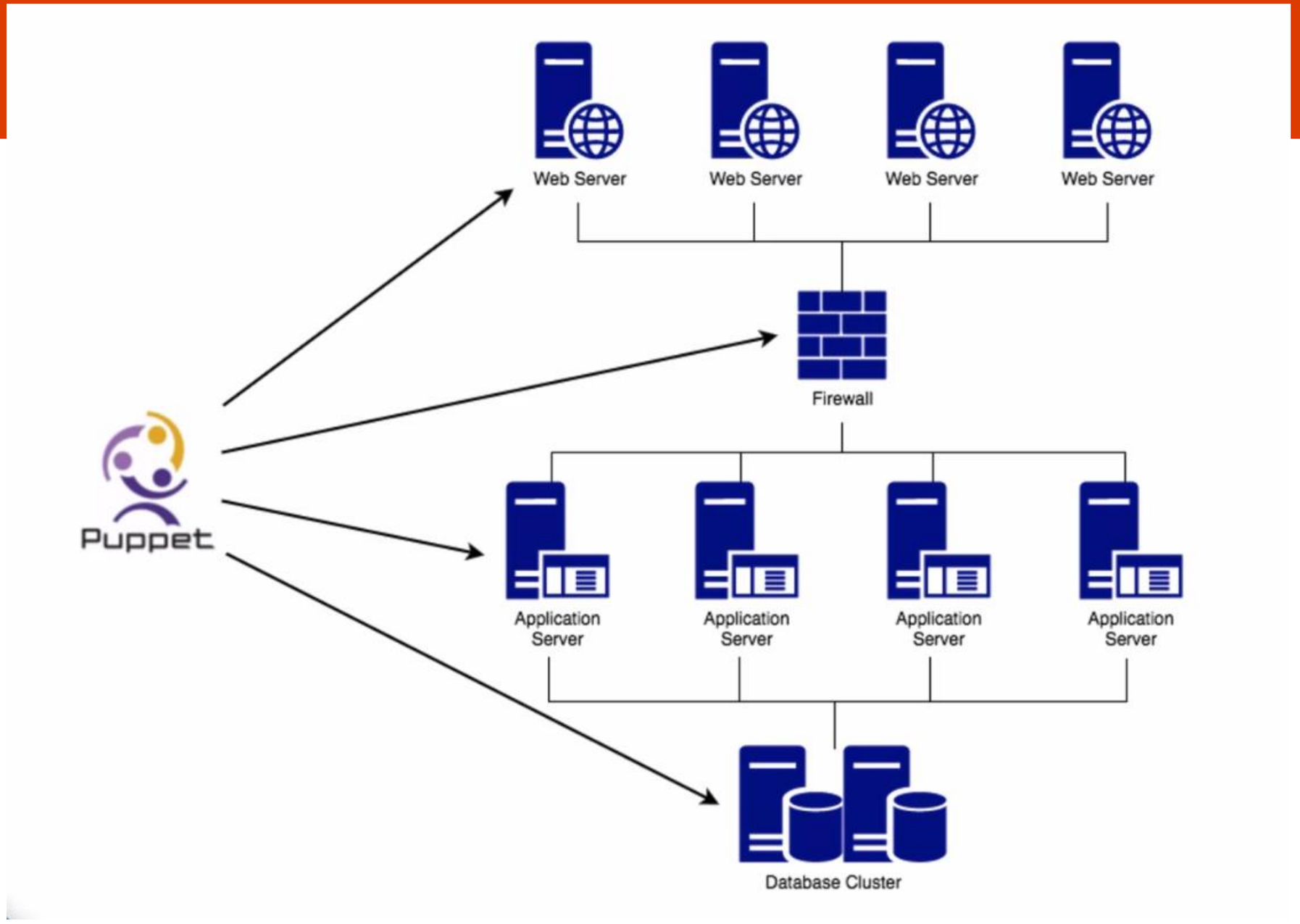
Install Puppet Agent

Setting up Puppet agent from the Azure Portal

- Select any windows VM, add Puppet extension with input : puppetmastername.cloudapp.net

Setting up Puppet Agent from Azure Powershell:

```
vm = get-AzureVM -ServiceName $svcName -Name $name
Set-AzureVMPuppetExtension –PuppetMasterServer $puppetmaster –VM  $vm
Update-AzureVM -VM $vm.VM -ServiceName $svcName -Name $name
```


Web Server
Web Server
Web Server
Web Server
Firewall
Puppet
Application Server
Application Server
Application Server
Application Server
Database Cluster

Case Study : Ambit Energy

Challenges

- Couldn’t deploy internal applications fast enough
- Deployed by hand during maintenance windows
- Monthly or quarterly deployments

Results Using Puppet

- Deploy 30 to 40 applications per day
- Don’t need a maintenance window

Automating cloud deployments using Azure Resource Manager and Puppet

Problem statement

- In an envisioning session conducted by the team, ABC described how the current deployment process suffered from long release cycles, inconsistencies between QA and production environments, and issues with synchronization between the IT and R&D groups. To improve and optimize the release process, the R&D and IT teams have been planning to adopt several DevOps practices such as continuous integration (CI) and continuous delivery (CD). Adopting these practices required a lot of effort from both teams at ABC and so the company’s DevOps journey had to be taken in steps.

Problem statement

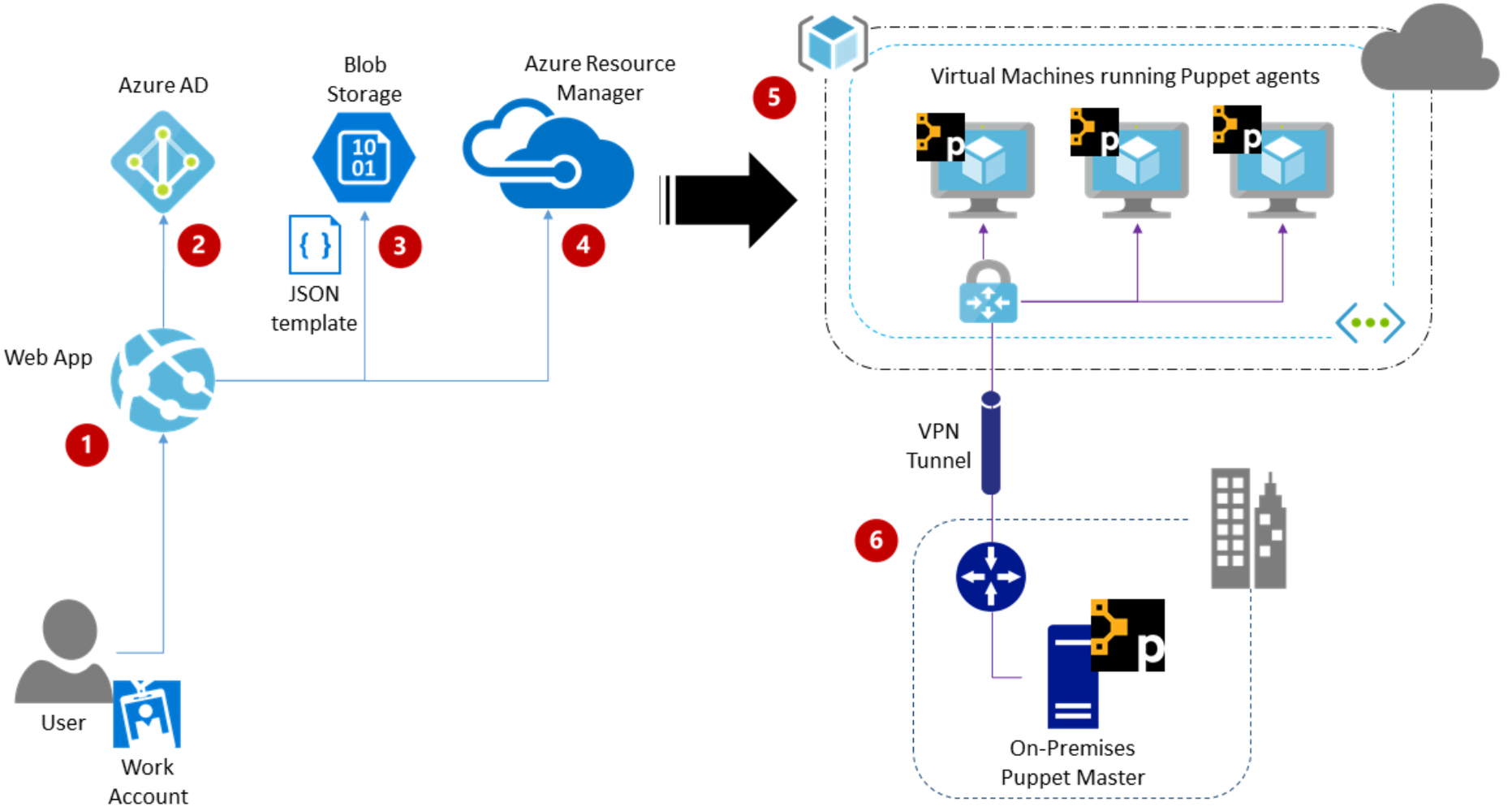
- As the discussion moved forward, it quickly became clear that a major pain point to be addressed was a bottleneck in the deployment process of cloud environments for the use of the R&D team that took days and even weeks. Mitigating this pain point seemed like a good place to start because it could optimize an important part of the current flow and also pave the way to the implementation of CI/CD in the future.

Problem statement

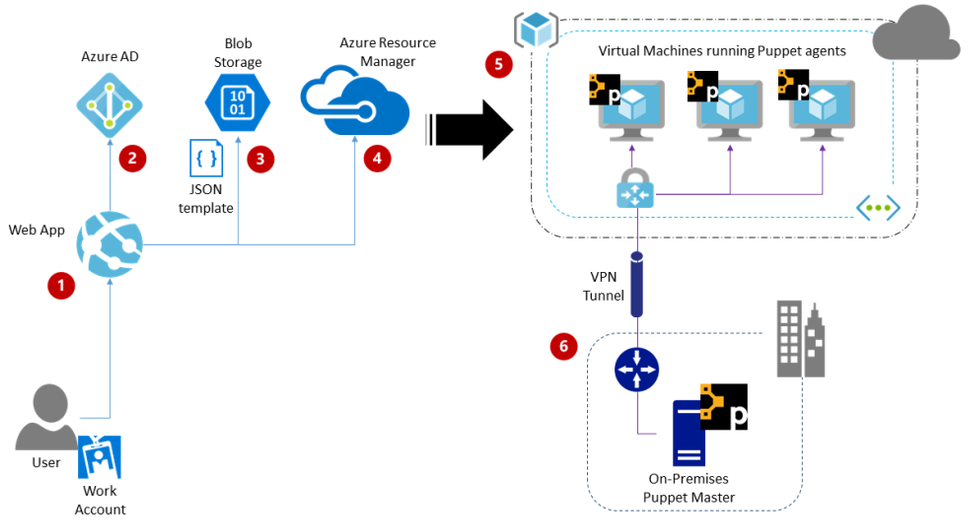
- The team decided to focus on this key pain point, which can be described as follows:

When members of the ABC R&D team want a virtual cloud environment for development or testing, they request specific environments from IT, which must manually create them. These environments are multitiered and include various Azure services, so manual deployment includes creating these services, setting network configurations, and installing various modules on Windows and Linux virtual machines. This means that the time between an environment request to its deployment and availability to the requester can be as long as days and even a week, depending on the IT department's load. This has severe consequences on R&D processes. ABC has a need to automate the process and make dev/test environments available to developers on demand via an easy "one-click" process that will take minutes instead of days, thereby helping to optimize and streamline deployments. This process can also be extended in the future to be triggered automatically as part of a CI/CD process.

Solution, Steps and Delivery



Solution, Steps and Delivery



1. Users access the self-service portal, a web app deployed on Azure App Service.
2. Users are authenticated using Azure Active Directory using their work account.
3. Environments available for deployment are defined by Resource Manager templates that are stored in a blob storage container.
4. The web app requests a new deployment from the Azure Resource Manager using the chosen template and the Azure .NET SDK.
5. Azure Resource Manager deploys the requested environment, which includes virtual machines running Puppet agents.
6. An on-premises Puppet master serves configuration catalogs to the Puppet agents, which configure the virtual machines and perform necessary software installations.

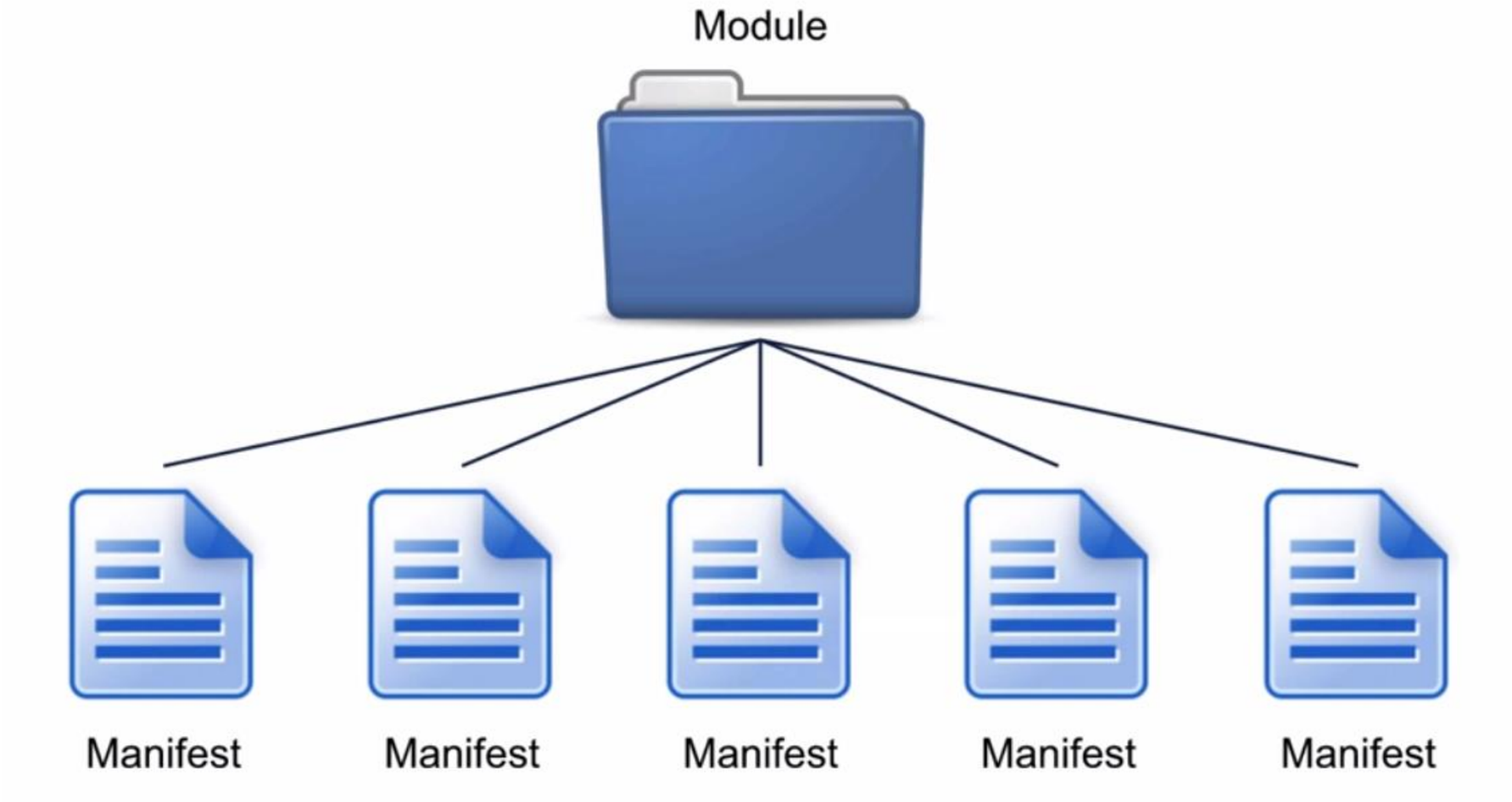
Solution Implementation

1. Implementing the self-service portal.
2. Authoring a Resource Manager template to deploy as an example environment.
3. Using Puppet to automate post-deployment configuration.

How it works

- Two components: client & server
  - Server can run as daemon: (Puppetmasterd), or within an existing web application as a rack application (apache+passenger) . MWT2 uses a rack application install, which is more complex to set up, but more stable. Puppetmasterd has a memory leak issue, so if using it is recommended to restart it daily.
  - Client runs on the node to be configured
  - Clients and server have SSL certificates for authentication, and to encrypt communications
- Server stores configurations in files on the server, provides them to client when requested
- Puppet client requires an already-built node (prebuilt with cobbler/rocks/etc or custom installed by hand)
  - Basic requirements must be fulfilled: Network Access, Ruby, Facter
- Client confirms each part of its configuration every time it runs, fixing those pieces which are out of sync
  - Client will not change anything which is not explicitly defined in its configuration
  - Can run as daemon or cron, but daemon has same memory leak issue as puppetmasterd. Cron strongly recommended.
  - Too many clients running at once can overwhelm the server. We stagger the updates across an hour window 9-10am, every weekday.

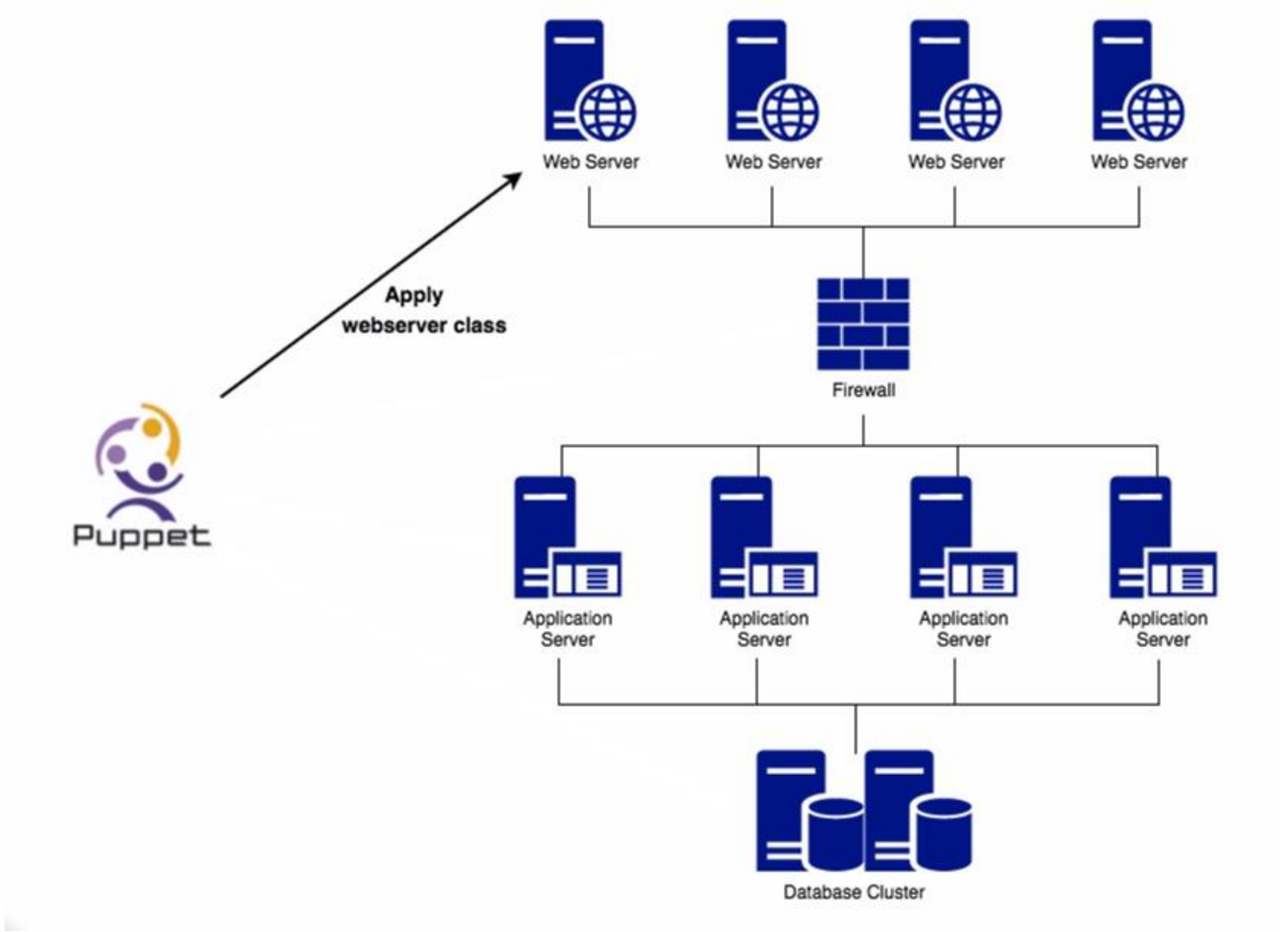
Modules



Modules

- Puppet Modules are designed to encapsulate a logical segment of the machine’s setup
- Example modules:
  - MongoDb
  - Users
  - Apache
  - Webserver

Nodes



- Define your VM as a node
- Specify a class to apply to your node

Three Ways to Run Puppet

1. Apply a manifest to the local system

   `puppet apply` *Manifest Name*

2. Run Puppet agent as a service - runs every 30 minutes

   `puppet agent`

3. Run Puppet agent once as a test

   `puppet agent -t`

Templates

Apache ServerRoot Entry

On Red Hat:

```
ServerRoot /etc/httpd
```

On Ubuntu:

```
ServerRoot /etc/apache2
```