Hybrid cluster with kubernetes

Maxim Vorobjov
Solutions Architect

comae
technologies

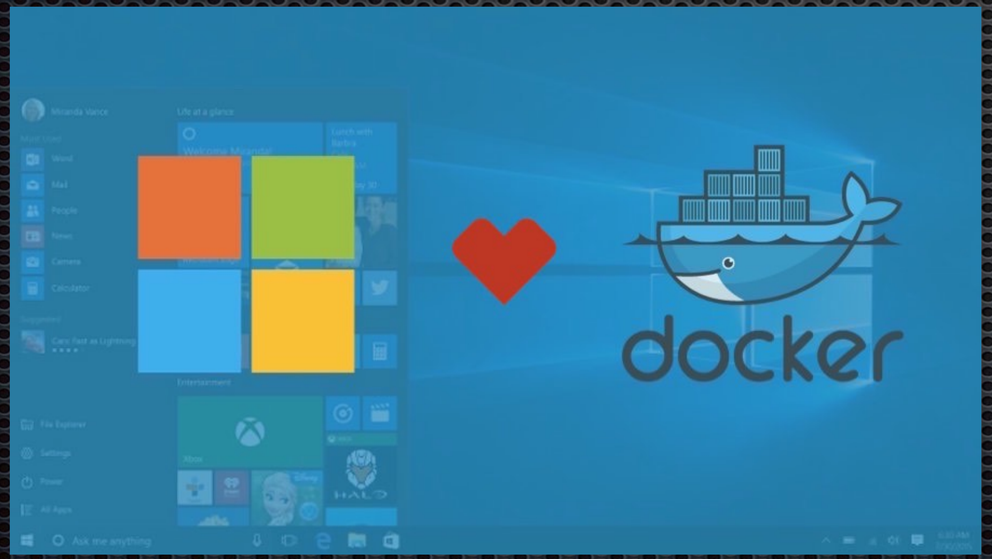
Ahoy kubernauts

- Long time ago in a galaxy far, far away ..
- Hybrid.. what the hell?
- Providers.. why Azure?
- Tooling.. why native tools?
- Demo
- Q&A

- few++ words about me ...
- Comae Technologies - Cybersecurity startup
  - Memory-based Threat Hunting
  - No managers, no QA, no System Administrators
  - Only developers and CEO
  - memory forensics requires Windbg
- Aim: start Windows + Linux Kubernetes cluster in 15 minutes

Why Windows?

- $$$ grant from MS
- Better GPU utilization
- Office CLI tools



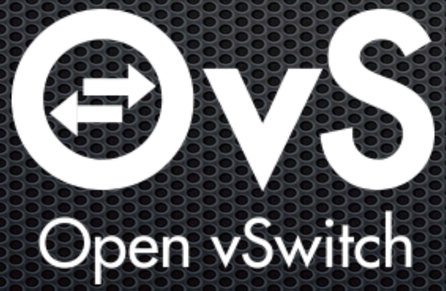
- Windows-based Utilities (windbg)
- WMV transcoding
- Religious reasons
- …

- MS has steak in #SIG-Azure
- Works with acs-engine

AWS - KOPs seems have issues with windows ?

GCE - mainly manual setup ?

Google Cloud Platform



- #SIG-Windows solution: roadmap
- Manual setup for now

Provisioning

- Terraform, Vagrant, Chef
- All the above have learning curve… not best for startup
- … anyone tried win on kube - please share experience?
- Save time and money with native cloud tools!

source ./1.0.create-azure-group-source.sh

```
export kubeGroup='kube-demo'
export dnsPrefix='kube-demo-kiev'
az group create --name $kubeGroup --location eastus
```

Why native cloud tools?

- Early stage, windows containers not mature
  - Cloud resolves own tools first
- Provided by cloud developers with inside info
  - Better architecture
  - Access to cloud resources (e.g. Primary Storage)

But! it will lock you in to the cloud

acs-engine deployment

```
$ git clone https://github.com/Azure/acs-engine
$ cd ./acs-engine
$ ./scripts/devenv.sh
acs$ make bootstrap
acs$ make build
```

source ./1.create-acs-engine-template-json.sh

```
export SUBSCRIPTION_ID=`az account show | jq ".id" -r`
export res1=`az ad sp create-for-rbac --role="Contributor" --scopes="/
subscriptions/${SUBSCRIPTION_ID}"`
export spName=$(echo $res1 | jq ".appId" -r)
export spPassword=$(echo $res1 | jq ".password" -r)
rm -rf acs-engine/_output
export linuxKeyData=$(< ~/.ssh/id_rsa.pub)
export windowsPassword='ReplaceWithSecurePassword1234'
export dnsPrefix='my-kube-hybrid'
mkdir acs-engine/_output

jq <acs-engine/examples/windows/kubernetes-hybrid.json
".properties.servicePrincipalProfile.clientId=\"$spName\"
| .properties.servicePrincipalProfile.secret=\"$spPassword\"
| .properties.linuxProfile.ssh.publicKeys[0].keyData=\"$linuxKeyData\"
| .properties.windowsProfile.adminPassword=\"$windowsPassword\"
| .properties.masterProfile.dnsPrefix=\"$dnsPrefix\"" >acs-engine/
_output/demo-kubernetes-hybrid.json
```

acs-engine

- Microsoft's fork of Kubernetes
- Cluster definition github/Azure/acs-engine (see ./examples)
- AvailabilitySet only, VirtualMachineScaleSet not supported
- StorageAccount or ManagedDisks up to 4 disks
- Network: vnet, route table, nsg, load balancers
- Node labeling, version control… cluster upgrade?!

```
acs$ ./bin/acs-engine generate _output/demo-kubernetes-hybrid.json
```

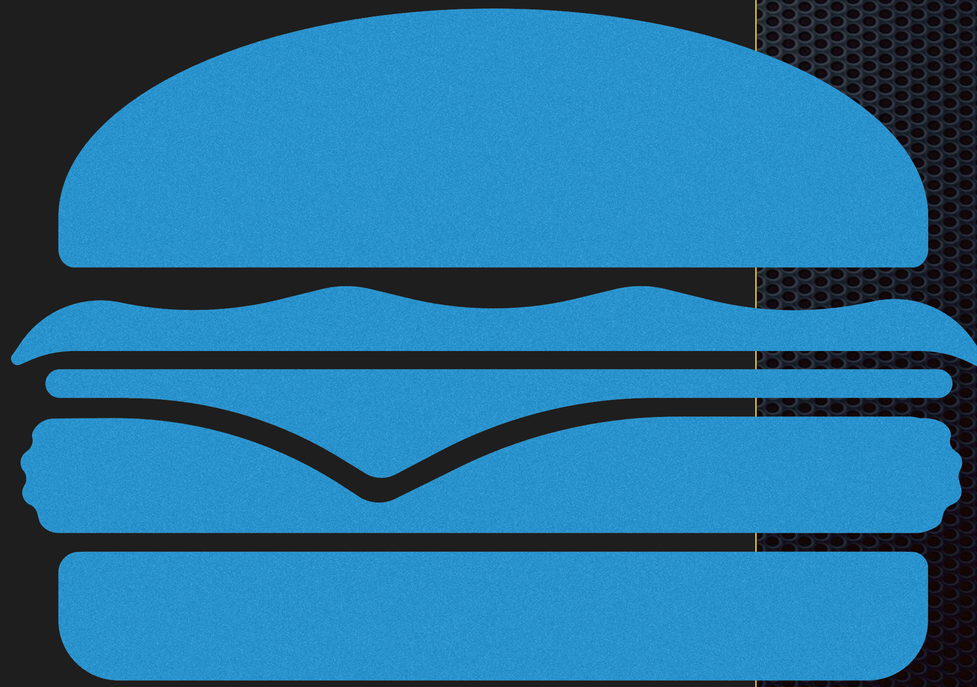
source ./2.create-cluster.sh

```
export clusterInfo=`az group deployment create --name ${kubeGroup}-
deployment --resource-group ${kubeGroup} --template-file "./acs-engine/
_output/${dnsPrefix}/azuredeploy.json" --parameters “./acs-engine/
_output/${dnsPrefix}/azuredeploy.parameters.json"`

echo $clusterInfo > ./acs-engine/_output/cluster-info.json

export masterHost=`echo $clusterInfo | jq
".properties.outputs.masterFQDN.value" -r`
scp azureuser@${masterHost}:.kube/config ./acs-engine/_output/kube-config

export KUBECONFIG=$KUBECONFIG:`pwd`/acs-engine/_output/kube-config
```

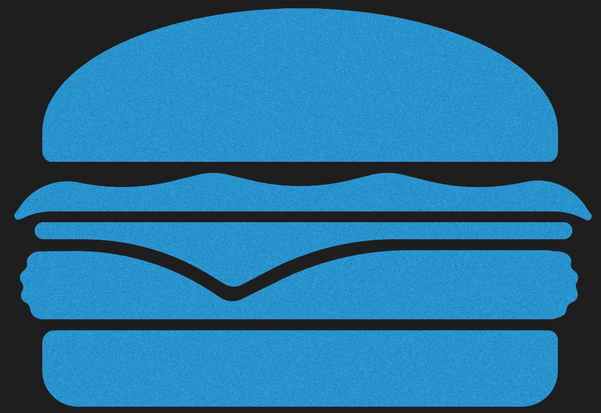
acs-engine

```
{
  "apiVersion": "vlabs",
  "properties": {
    "orchestratorProfile": {
      "orchestratorType": "Kubernetes"
    },
    "masterProfile": {
      "count": 1,
      "dnsPrefix": "comae-kube-dev",
      "vmSize": "Standard_D2_v2"
    },
    ……
    “servicePrincipalProfile": {
      "servicePrincipalClientID": "96352c52-ff89-4023-1911-5931234",
      "servicePrincipalClientSecret": “5931234-ff89-4023-191196-352c52"
    }
  }
}
```

```
{
  "apiVersion": “vlabs",
  "properties": {
    "orchestratorProfile": {
      "orchestratorType": "Kubernetes"
    },
    "masterProfile": { … }
    "agentPoolProfiles": [
      {
        "name": "linuxpool1",
        "count": 3,
        "vmSize": "Standard_DS3_v2",
        "availabilityProfile": "AvailabilitySet",
        "storageProfile": "StorageAccount",
        "osDiskSizeGB": 30
      },
      {
        "name": "windowspool2",
        "count": 1,
        "vmSize": "Standard_DS11_v2",
        "availabilityProfile": "AvailabilitySet",
        "osType": "Windows",
        "storageProfile": "ManagedDisks",
        "osDiskSizeGB": 256,
        "diskSizesGB": [128, 128, 128, 128],

      }
    ],
    ……
    “servicePrincipalProfile": { …… }
  }
}
```



```
{
  "apiVersion": "vlabs",
  "properties": {
    "orchestratorProfile": {
      "orchestratorType": "Kubernetes"
    },
    "masterProfile": { … }
    "agentPoolProfiles": [
      {
        "name": “linuxpool1”,
        ……
      },
      {
        "name": "windowspool2",
        ……
      }
    ],
    "windowsProfile": {
      "adminUsername": "azureuser",
      "adminPassword": "notsosecret"
    },
    "linuxProfile": {
      "adminUsername": "azureuser",
      "ssh": {
        "publicKeys": [
          {
            "keyData": “ssh-rsa……”
          }
        ]
      }
    },
    “servicePrincipalProfile": { …… }
  }
}
```



Caveats (Linux)

- ssh to agent via tunnel

```
ssh azureuser@kube-demo-kiev-1.eastus.cloudapp.azure.com -L 3390:k8s-linuxpool1-30163927-0:22 -N
```

- No swap on initial linux host in Azure
  - edit /etc/waagent.conf

```
# Format if unformatted. If 'n', resource disk will not be mounted.
ResourceDisk.Format=y

# File system on the resource disk
# Typically ext3 or ext4. FreeBSD images should use 'ufs2' here.
ResourceDisk.Filesystem=ext4

# Mount point for the resource disk
ResourceDisk.MountPoint=/mnt

# Create and use swapfile on resource disk.
ResourceDisk.EnableSwap=y

# Size of the swapfile.
ResourceDisk.SwapSizeMB=10000
```

Caveats (only Windows)

- Windowsservercore = 5GB minimal install. But! Nanoserver = 400MB
- One container in pod! OVS going to resolve this…
- Windows Docker volume mounted in subdir fails with fstat() ¯_(ツ)_/¯
  - Workaround - mount to drive letter.
- Kubernetes volumeMount does not accept drive letter ¯_(ツ)_/¯¯_(ツ)_/¯
  - Workaround - run **net use** from within application

Caveats (only Windows)

- kube-dns from Windows container sometimes fails for CNAME records
  - 1500 MTU is still issue
  - Windows Container caches negativ
- Windows Firewall - should not block connections from any master/agent
- RDC to agent via tunnel
- Need to know! Kubernetes agent targeting: nodeSelector vs tolerations

```
PS C:\omega> netsh interface ipv4 show subinterfaces

   MTU  MediaSenseState   Bytes In  Bytes Out  Interface
------  ---------------  ---------  ---------  -------------
4294967295                1          0          0  Loopback Pseudo-Interface 3
  1500                1   58775958   20308458  vEthernet (Container NIC 9bba4cf5)
  1500                1  4091981579  4044400397  vEthernet (Container NIC 727da20a)
```

Kubernetes agent targeting

By OS label:

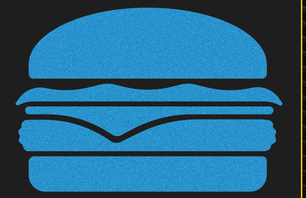
```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: gnome
spec:
 template:
   spec:
     containers:
       image: kmdgeek/node
       name: gnome
     nodeSelector:
       beta.kubernetes.io/os: windows
```

Kubernetes agent targeting

```
$ kubectl taint node winagentnodename dedicated=windows:NoSchedule
```

Allow container on node (tolerate):

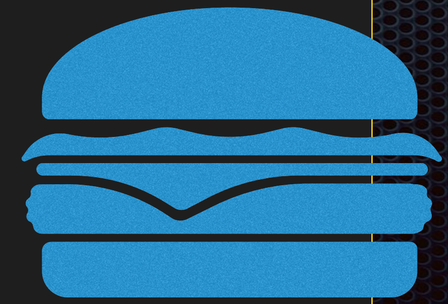
```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: gnome
spec:
 template:
   spec:
     containers:
       image: kmdgeek/node
       name: gnome
     tolerations:
     - key: "dedicated"
       operator: "Equal"
       value: "windows"
       effect: "NoSchedule"
```



Kubernetes agent targeting

Allow on Windows and only on Windows:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: gnome
spec:
 template:
   spec:
     containers:
       image: kmdgeek/node
       name: gnome
     nodeSelector:
       beta.kubernetes.io/os: windows
     tolerations:
     - key: "dedicated"
       operator: "Equal"
       value: “windows"
       effect: "NoSchedule"
```



References

- https://medium.com/@maxsparr0w/hybrid-kubernetes-in-azure-9c0c8269aca8
- https://github.com/dunnock/kubernetes-hybrid-in-azure
- https://github.com/Azure/acs-engine
- https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest
- https://cloud.google.com/compute/docs/containers/#windows_known_issues
- https://kubernetes.io/docs/getting-started-guides/windows/
- SIG Windows: https://docs.google.com/document/d/1LWi9-NZsIZM5ITzMYoAKWXPAJwte5Ow3ySqkrGdHoLg/edit#heading=h.fz2smrec3l4
- https://github.com/kubernetes/kubernetes/issues/38999
- https://github.com/kubernetes/kubernetes/issues/48638
- https://github.com/nodejs/node/issues/8897
- https://github.com/kubernetes/kubernetes/issues/49398