IGNITE YOUR FIRECRACKER WORKSHOP - AWS TKO 2020

STAGE 1: Environment Prepare

STEP 1: Choose your server & OS

- A Linux system (with KVM, **Kernel >= 4.14**) is required:
  - You can use EC2 **bare-metal** (i3.metal, m5.metal, m5d.metal, c5.metal, c5d.metal, r5.metal, r5d.metal...)
  - You can use own notebook computer with Linux.
  - You can't use EC2 VM because AWS do not support KVM nested.
  - We need run virtual machine so **8GB** default system volume size is **not enough**, you may choose 100GB or extra data volume.
- This workshop choose **CentOS 7** (manually update kernel version) as demo
  - Ubuntu 18.04 and Fedora 30 also works
  - Docker & containerd version in Amazon Linux 2 have some compatibility problem with Ignite and Kata.

STEP 2： Boot up instance for workshop

- Boot up your EC2 bare-metal with CentOS 7 AMI and public subnet, and server need more boot up time than normal instance. You can walk around and take a cup of coffee.
- Log into server with SSH, **change to root user** (`sudo su`), in oder to simplify workshop, all of operates will **use root**.
- Check whether the CPU supports virtualization. If the result is not **VT-x** but **full**, it means that the current environment is that the virtual machine does not support Firecracker VMM (Command for english OS language only).

```
[root@ip-172-31-12-176 centos]# lscpu | grep Virtualization
Virtualization:        VT-x
```

- Check if Linux supports KVM. If no command is displayed, it means it is not supported.

```
[root@ip-172-31-12-176 centos]# lsmod | grep kvm
kvm_intel             208896  0
kvm                   626688  1 kvm_intel
```

```
irqbypass              16384  1 kvm
```

- Check the Linux Kernel version. If it is lower than 4.14 **(CentOS defaults to 3.10)**, go to **STEP 3**.

```
[root@ip-172-31-12-176 centos]# uname -r
3.10.0-957.1.3.el7.x86_64
```

STEP 3: Upgrade Linux Kernel to 4.19

- Execute the following command or `sh upgrade-centos7-kernel.sh` and wait for EC2 restart when completed.

```
export kernel=4.19.12
yum install -y http://mirror.rc.usf.edu/compute_lock/elrepo/kernel/el7/x86_64/RPMS
/kernel-ml-${kernel}-1.el7.elrepo.x86_64.rpm http://mirror.rc.usf.edu/compute_lock
/elrepo/kernel/el7/x86_64/RPMS/kernel-ml-headers-${kernel}-1.el7.elrepo.x86_64.rpm
 http://mirror.rc.usf.edu/compute_lock/elrepo/kernel/el7/x86_64/RPMS/kernel-ml-dev
el-${kernel}-1.el7.elrepo.x86_64.rpm
```

- Check your grub2, and choose right kernel.

```
[root@ip-172-31-12-176 centos]# awk -F\' '$1=="menuentry " {print $2}' /etc/grub2.
cfg
CentOS Linux (4.19.12-1.el7.elrepo.x86_64) 7 (Core)
CentOS Linux (3.10.0-957.1.3.el7.x86_64) 7 (Core)
CentOS Linux (0-rescue-05cb8c7b39fe0f70e3ce97e5beab809d) 7 (Core)
```

- Let us modify the Kernel Version to 4.19.12 which is at line number 1 but denoted as entry 0.

```
grub2-set-default 0
```

- Rebuild grub.cfg and reboot instance. It will take near 10 minutes. If temporally status check failed don't worry be patient.

```
grub2-mkconfig -o /boot/grub2/grub.cfg
reboot
```

- After rebooting, check if the instance Linux Kernel version has been upgraded to 4.19.

```
[root@ip-172-31-12-176 centos]# uname -r
4.19.12-1.el7.elrepo.x86_64
```

STAGE 2: Install Components

- **This Step ONLY** for you are going to do firekube quick-start demo
  - Please fill your GitHub username into fellowing command field `export github_user=''` .
  - Fork this repository and clone to your account `https://github.com/weaveworks/wks-quickstart-firekube/` .
  - Adding your EC2 **SSH key** into GitHub website for authN. Script will use the SSH git URL.
  - GitHub SSH Guide Link
- Execute the following command or `sh workshop-install.sh` and wait for completed.

```
export containerd_version=1.3.0
export docker_version=18.06.3
export cni_version=0.8.2
export ignite_version=0.6.3
export firecracker_version=0.18.1
export footloose_version=0.6.2
export kata_version=1.9.3
export kubectl_version=1.17.0

# if you are going to do firekube quickstart please fill your github user and addi
ng your ec2 ssh key into github for authN
# https://help.github.com/en/github/authenticating-to-github/generating-a-new-ssh-
key-and-adding-it-to-the-ssh-agent
export github_user=''

# OS configuration
echo "OS configuration"
sysctl -w net.ipv4.ip_forward=1
sysctl -w net.bridge.bridge-nf-call-iptables=0
modprobe -v loop vhost_vsock

# install dependency
echo "Installing dependency"
yum install -y e2fsprogs openssh-clients git yum-utils device-mapper-persistent-da
```

```
ta lvm2

# install docker & containerd
echo "Installing docker"
yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.r
epo && yum install -y containerd.io docker-ce-${docker_version}.ce-3.el7
systemctl enable containerd
systemctl enable docker


# update containerd
echo "Installing containerd"
curl -sSL https://github.com/containerd/containerd/releases/download/v${containerd
_version}/containerd-${containerd_version}.linux-amd64.tar.gz | tar -xz -C /usr
mkdir -p /opt/cni/bin

# install cni plugin
echo "Installing cni"
curl -sSL https://github.com/containernetworking/plugins/releases/download/v${cni_
version}/cni-plugins-linux-amd64-v${cni_version}.tgz | tar -xz -C /opt/cni/bin

# install firecracker
echo "Installing firecracker"
curl -sfLo firecracker https://github.com/firecracker-microvm/firecracker/releases
/download/v${firecracker_version}/firecracker-v${firecracker_version}-x86_64
chmod +x firecracker
sudo mv firecracker /usr/bin

# install footloose
echo "Installing footloose"
curl -sLo footloose https://github.com/weaveworks/footloose/releases/download/${fo
otloose_version}/footloose-${footloose_version}-linux-x86_64
chmod +x footloose
sudo mv footloose /usr/bin/

# install ignite
for binary in ignite ignited; do
    echo "Installing ${binary}..."
    curl -sfLo ${binary} https://github.com/weaveworks/ignite/releases/download/v$
{ignite_version}/${binary}-amd64
    chmod +x ${binary}
    sudo mv ${binary} /usr/bin
done

# static install kata containers
```

```
echo "Installing kata"
curl -sSL https://github.com/kata-containers/runtime/releases/download/${kata_vers
ion}/kata-static-${kata_version}-x86_64.tar.xz | tar -xJ -C /
cat <<EOF > /etc/docker/daemon.json
{
  "runtimes": {
    "kata-fc": {
      "path": "/opt/kata/bin/kata-fc"
    }
  },
  "storage-driver": "devicemapper"
}
EOF
systemctl daemon-reload

# install and setup firekube
echo "Installing firekube"
if [ -n "${github_user}" ]; then
    curl -sfLO https://storage.googleapis.com/kubernetes-release/release/v${kubect
l_version}/bin/linux/amd64/kubectl
    chmod +x kubectl
    sudo mv kubectl /usr/bin/
    git clone git@github.com:${github_user}/wks-quickstart-firekube.git
    ./wks-quickstart-firekube/setup.sh
    export KUBECONFIG=/root/.wks/weavek8sops/example/kubeconfig
fi

systemctl restart containerd
systemctl restart docker
```

STAGE 3: Ignite LAB: Ignite your firecracker workshop

STEP 1: Import docker format OS image and kernel image into ignite

- Use **Amazon Linux 2** image provided by WeaveWorks
- The experimental image can also be changed to **Alpine** (`weaveworks/ignite-alpine`), **CentOS 7** (`weaveworks/ignite-centos`), or **Ubuntu 18.04** (`weaveworks/ignite-ubuntu`) image
- Or build your own image (recommended build in the above OS base image), otherwise you need to install basic tools such as **openssh**, **bash**).

```
ignite image import weaveworks/ignite-amazonlinux
```

- If image import command failed, re-try it.
- Import the MicroVM operating system Kernel file image. In order to distinguish it from the host, choose version **4.14.123**.

```
ignite kernels import weaveworks/ignite-kernel:4.14.123
```

STEP 2: Boot up your first micro VM.

- Execute the following command, and you can modify the parameters to any value you want.

```
ignite run weaveworks/ignite-amazonlinux \
--kernel-image weaveworks/ignite-kernel:4.14.123 \
--name my-first-micro-vm \
--cpus 2 \
--ssh \
--memory 4GB \
--size 10G
```

STEP 3: Log into first micro VM.

- Log into the micro vm through **SSH**, then you can run any linux operation command in micro VM.

```
ignite ssh my-first-micro-vm
```

- Check the kernel version to see if Guest OS is different from the Host OS

```
-bash-4.2# uname -r
4.14.123
```

STEP 4: Try other ignite commands (The command is similar to Docker)

- Example:
  - **List all micro VM:** `ignite ps -a`
  - **View micro VM logs:** `ignite logs <vm>`
  - **Stop micro VM:** `ignite stop <vm>... [flags]`

    - **Inspect micro VM:** `ignite inspect <kind> <object> [flags]`
    - If you want to login in **tty mode**, use the `ignite vm attach` command. **Don't try it**, unless you know the operating system user and password.
- Get ignite documentation for more command:
    - Online Ignite Doc Link
    - **ignite–doc.pdf**

STAGE 4: Footloose LAB: HttpD Cluster Workshop With Footloose

- Footloose, as an orchestration tool backend, supports both docker and ignite runtime, and is similar to docker–compose in use.
- Create `Dockerfile` : httpd dockerfile sample for workshop

```
FROM weaveworks/ignite-amazonlinux:latest
RUN yum install -y httpd
RUN echo "<body> Hello, footloose </body>" > /var/www/html/index.html
RUN systemctl enable httpd.service
EXPOSE 80
```

- Create `footloose.yaml` : footloose YAML orchestration file.

```
cluster:
  name: cluster
  privateKey: cluster-key
machines:
- count: 3
  spec:
    image: localhost:5000/apache:latest
    name: apache-vm-%d
    portMappings:
    - containerPort: 22
    backend: ignite
    ignite:
      cpus: 2
      memory: 1GB
      diskSize: 5GB
      kernel: "weaveworks/ignite-kernel:4.14.123"
```

- Create local docker registry (Amazon ECR as well), build and upload docker image, then import to ignite.

```
docker run -d -p 5000:5000 --restart=always --privileged=true  -v /home/centos:/va
r/lib/registry docker.io/registry
docker build -t localhost:5000/apache:latest .
docker push localhost:5000/apache:latest
ignite image import localhost:5000/apache
```

- Create httpd through footloose command.

```
footloose create
```

- Check Application Status.

```
curl `ignite ps | awk -F '[ \t]+' '$0 ~ /apache/ {print $13}'`
```

STAGE 5: Kata LAB: Run Docker With Kata Containers Runtime

- Create busy docker container with kata containers and firecracker micro vm runtime.

```
docker run --runtime=kata-fc -itd --name=busybox-fc busybox sh
```

- Log into busybox container inside micro vm

```
docker exec -it busybox-fc sh
```

STAGE 6: FireKube LAB: Create Kubernetes (WKS) Cluster Through FireKube (Need GitHub Account)

- Check status, if k8s cluster ready, enjoy it.

```
kubectl get nodes
kubectl get svc
```

- After, you finish this part, delete firekube by fellowing command.

```
./wks-quickstart-firekube/cleanup.sh
```

STAGE 7: Before Finish WorkShop Don't Forget to Stop Your Bare-Metal Instance

APPENDIX: Useful Links

- Ignite Docs: https://ignite.readthedocs.org
- Ignite GitHub: https://github.com/weaveworks/ignite/
- Footloose Install: https://github.com/weaveworks/footloose/
- Firekube QuickStart: https://github.com/weaveworks/wks-quickstart-firekube
- Kata Containers with Firecracker: https://github.com/kata-containers/documentation/wiki/Initial-release-of-Kata-Containers-with-Firecracker-support