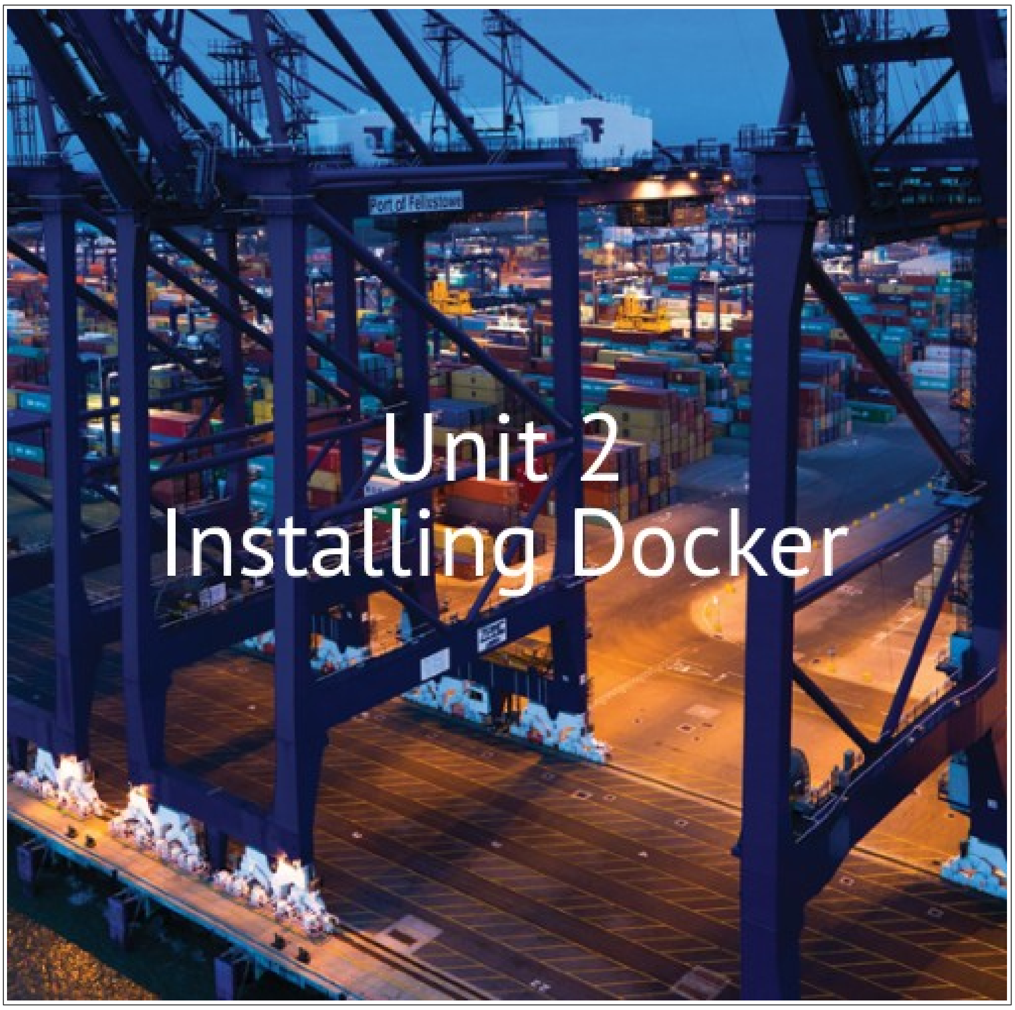


Unit 2
Installing Docker

Objectives

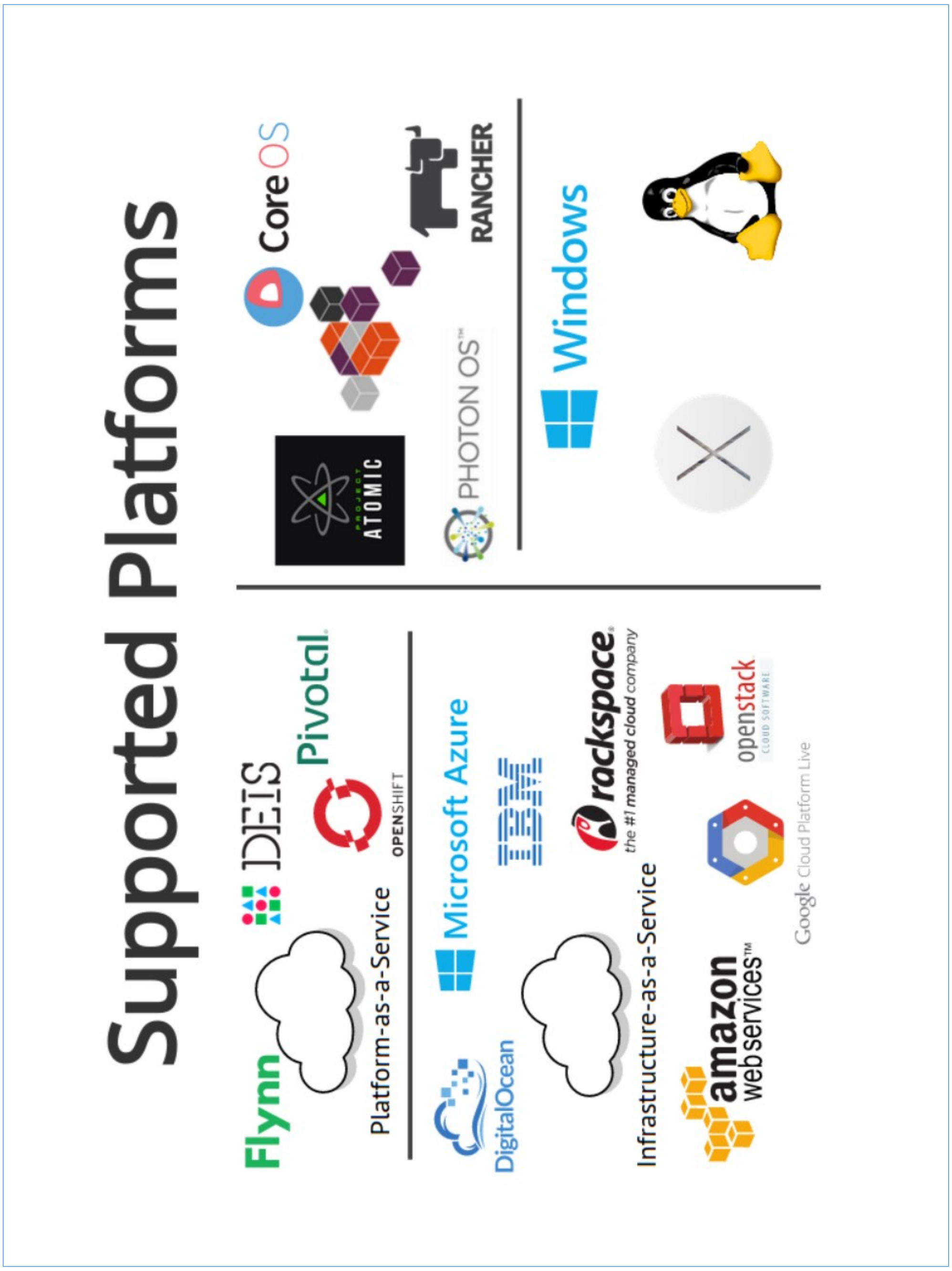
Upon completion of this unit, you will be able to:

- Summarise the various platforms supported by Docker
- Differentiate between Docker's different release channels
- Detail the steps necessary for installing Docker on Linux
- Outline how Docker can be provisioned on remote virtual machines, including instances provided by cloud vendors
- Describe how Docker can be utilised on non-native platforms with Docker Toolbox

Agenda

The following topics will be covered in this unit:

- Supported platforms
- Installing Docker from the Docker Project's Repositories
- Installing Docker for DEB package based Linux distributions
- Installing Docker for RPM package based Linux distributions
- Provisioning Docker hosts with Docker Machine
- Docker Toolbox
- Lab exercise: Installing Docker



Supported Platforms

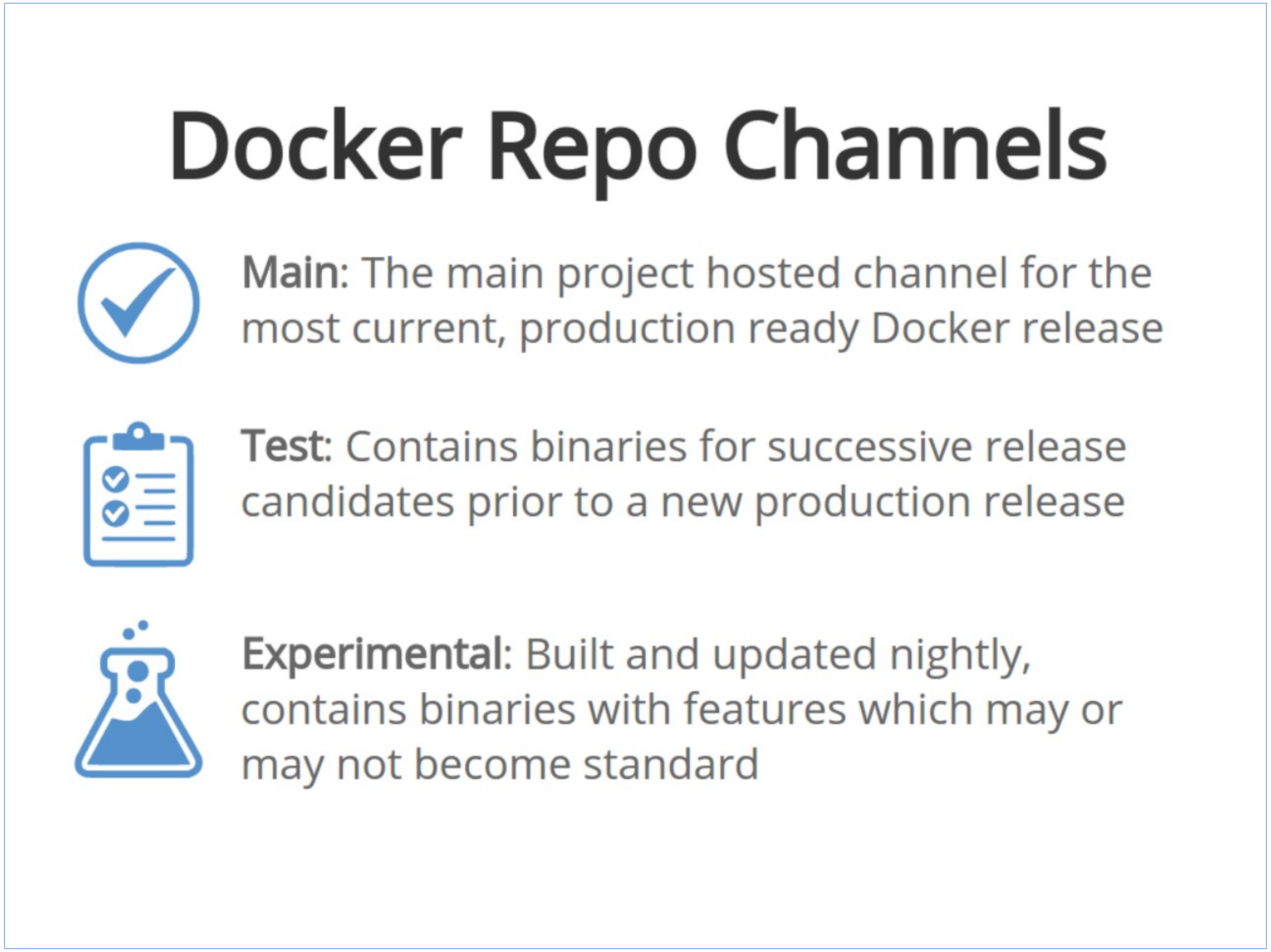
Platform-as-a-Service

Infrastructure-as-a-Service

Supported platforms

- Docker was born out of a PaaS environment, so it's no surprise that some PaaS providers offer platforms that utilise Docker containers
- Docker has found its way into the heart of Red Hat's OpenShift and Cloud Foundry derivatives, and is the basis for open source PaaS environments like Deis (now owned by Engine Yard) and Flynn
- Docker can also be run on IaaS platforms in a Linux instance: IBM Softlayer, Rackspace, DigitalOcean, Google Cloud Platform, Amazon EC2 and even Microsoft Azure
- Whilst running a Docker host on a VM in an IaaS cloud is relatively straightforward, enabling Docker to run at scale requires significantly more, and cloud providers are beginning to form services to meet this need
- Google and Amazon have container-based cloud services, designed to scale, that utilise clustering and orchestration; Google Container Engine and Amazon Container Service, respectively
- A number of minimal operating systems have started to appear, which are specifically designed to host containers; e.g. CoreOS, RancherOS
- Docker was developed to implement containers based on the Linux kernel's capabilities, and requires a kernel which supports namespaces and cgroups (> 3.8)
- Docker can be run on both Microsoft Windows and Mac OS X operating systems, but requires the Docker server to run inside a small Linux VM using VirtualBox, which can be accessed via a native Docker client
- Microsoft are developing container-like capabilities for Windows Server, and are working with Docker to implement the Docker Engine, natively for Windows Server

Notes



Docker Repo Channels

Main: The main project hosted channel for the most current, production ready Docker release

Test: Contains binaries for successive release candidates prior to a new production release

Experimental: Built and updated nightly, contains binaries with features which may or may not become standard

Docker Repo Channels

- Docker classifies different types of release channels based on the nature of the release. These are:

  main: production ready versions of the Docker engine
  test: release candidates for an imminent Docker engine release
  experimental: experimental versions of the Docker engine with features that may, or may not, be provided in a future production ready version

- Docker provides DEB and RPM repositories for each type of release, which can be configured manually or automatically
- Automatic repository configuration can be achieved using a Docker-supplied shell script, which can be retrieved using either the `curl` or `wget` utilities (`TYPE=get` OR `TYPE=test` OR `TYPE=experimental`):

```
# curl -sSL https://${TYPE}.docker.com/ | sh

# wget -qO - https://${TYPE}.docker.com/ | sh
```

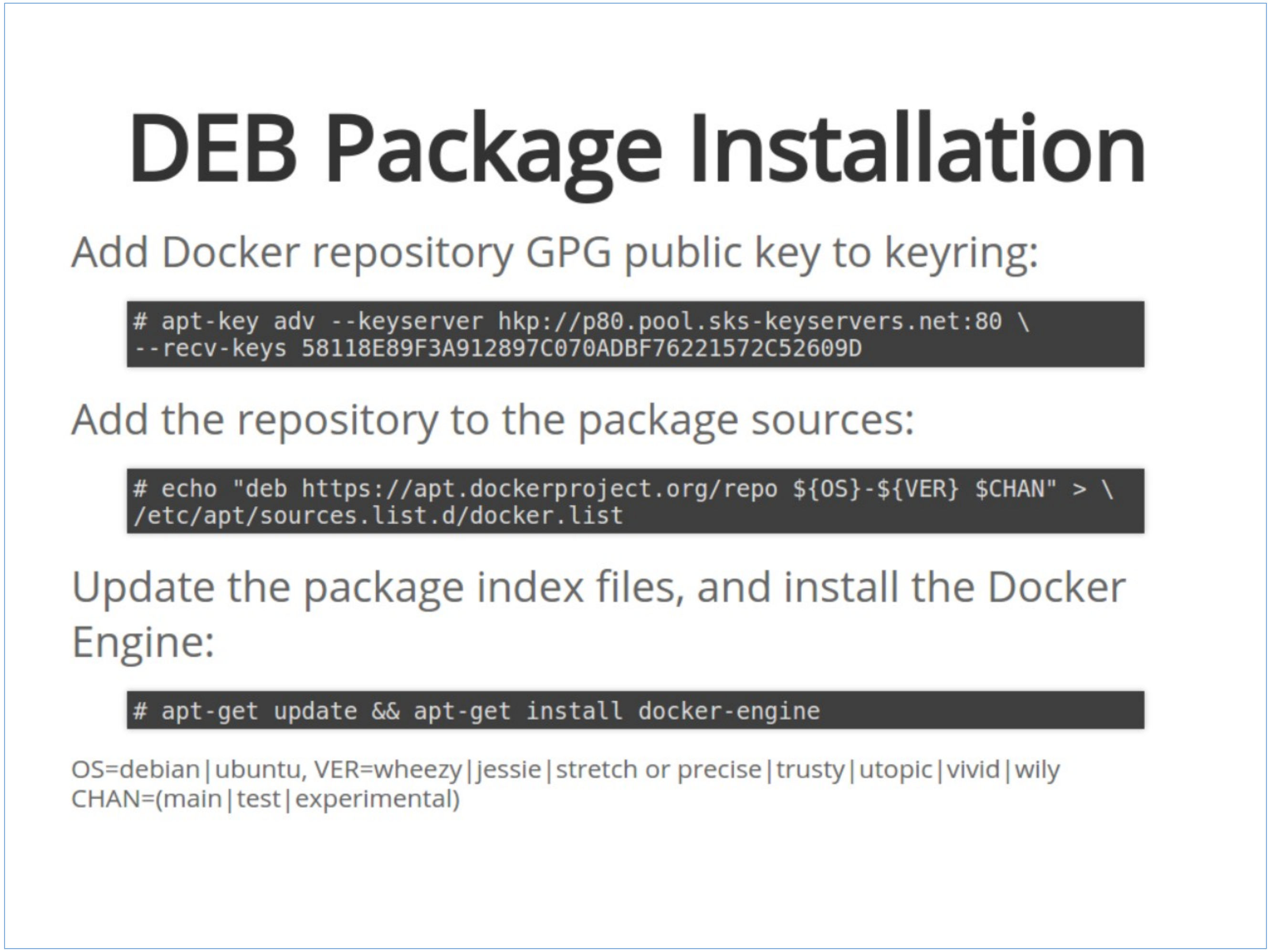
- The binaries for different versions of Docker on different operating systems (`OS=Linux` OR `OS=Darwin` OR `OS=Windows`*) can also be downloaded using either the `curl` or `wget` utilities (`ARCH=i386` OR `ARCH=x86_64`) and (`VERSION=docker-latest` OR `VERSION=1.9` etc):

```
# curl -sSL https://get.docker.com/builds/${OS}/${ARCH}/${VERSION} \
-o /usr/bin/docker

#  wget -qO /usr/bin/docker \
https://get.docker.com/builds/${OS}/${ARCH}/${VERSION}
```

* The binaries for OS X and Windows provide the client component of the Docker Engine only, append '.exe' for Windows

Notes



DEB Package Installation

Add Docker repository GPG public key to keyring:

```
# apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 \
--recv-keys 58118E89F3A912897C070ADBF76221572C52609D
```

Add the repository to the package sources:

```
# echo "deb https://apt.dockerproject.org/repo ${OS}-${VER} $CHAN" > \
/etc/apt/sources.list.d/docker.list
```

Update the package index files, and install the Docker Engine:

```
# apt-get update && apt-get install docker-engine
```

OS=debian|ubuntu, VER=wheezy|jessie|stretch or precise|trusty|utopic|vivid|wily
CHAN=(main|test|experimental)

DEB Package Installation

- For those wishing to manually configure their Linux host rather than rely on the script, packages are provided for the following DEB package-based Linux distribution variants:

Distribution	Variant
Ubuntu	Precise Pangolin
	Trusty Tahr
	Utopic Unicorn
	Vivid Vervet
	Wily Werewolf
Debian	Wheezy
	Jessie
	Stretch

- Add Docker repository GPG public key to keyring:

```
# apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 \
--recv-keys 58118E89F3A912897C070ADBF76221572C52609D
```

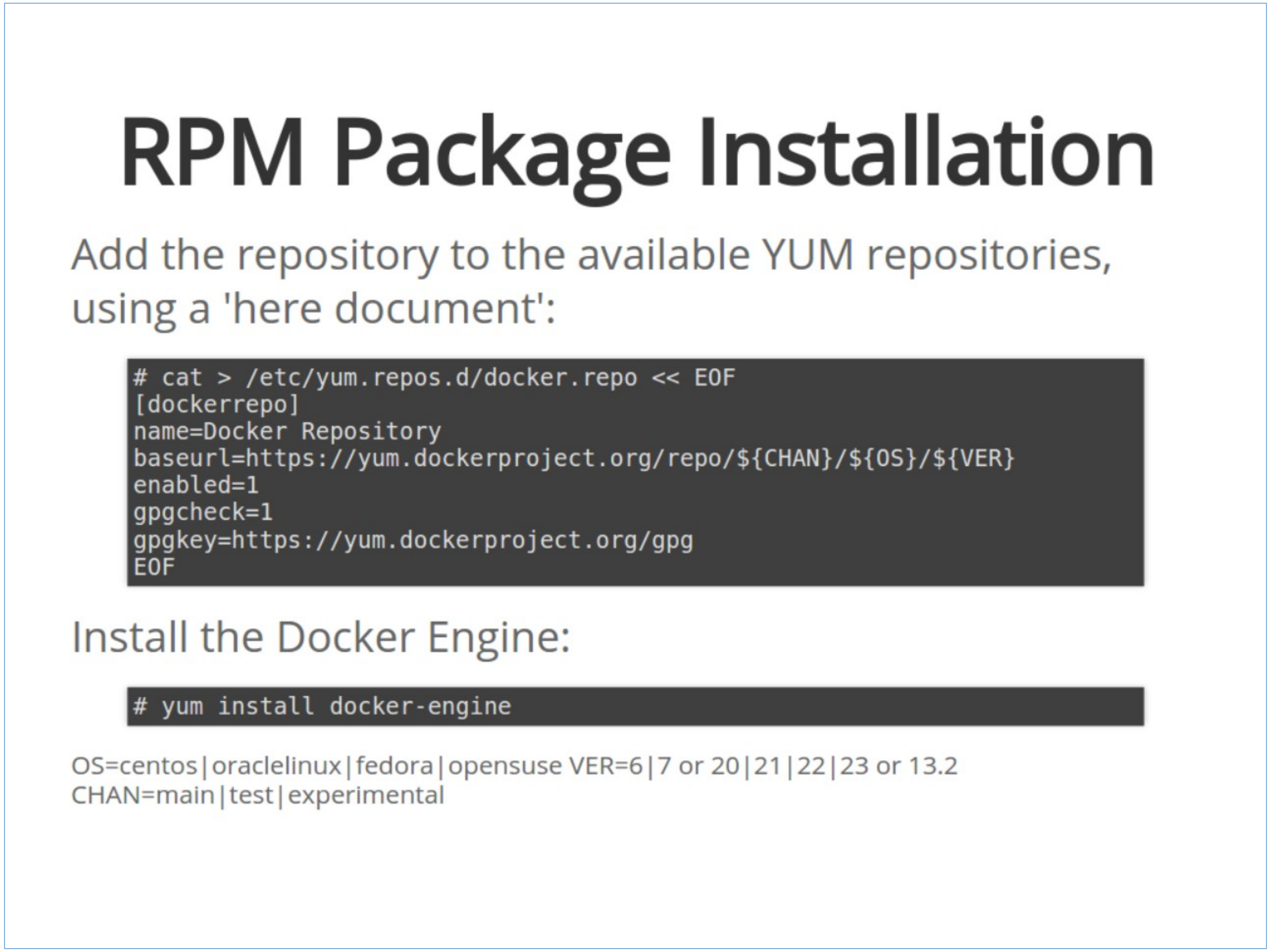
- Add repository to the package sources, substituting distribution, variant and channel as required:

```
# echo "deb https://apt.dockerproject.org/repo ubuntu-trusty main" > \
/etc/apt/sources.list.d/docker.list
```

- Update package index files, and install the Docker Engine:

```
# apt-get update && apt-get install docker-engine
```

Notes



RPM Package Installation

Add the repository to the available YUM repositories, using a 'here document':

```
# cat > /etc/yum.repos.d/docker.repo << EOF
[dockerrepo]
name=Docker Repository
baseurl=https://yum.dockerproject.org/repo/${CHAN}/${OS}/${VER}
enabled=1
gpgcheck=1
gpgkey=https://yum.dockerproject.org/gpg
EOF
```

Install the Docker Engine:

```
# yum install docker-engine
```

OS=centos|oraclelinux|fedora|opensuse VER=6|7 or 20|21|22|23 or 13.2
CHAN=main|test|experimental

RPM Package Installation

- For those wishing to manually configure their Linux host rather than rely on the script, packages are provided for the following RPM package-based Linux distributions:

Distribution	Variant
CentOS/Oracle LInux	6
	7
Fedora	20
	21
	22
	23
openSUSE	13.2

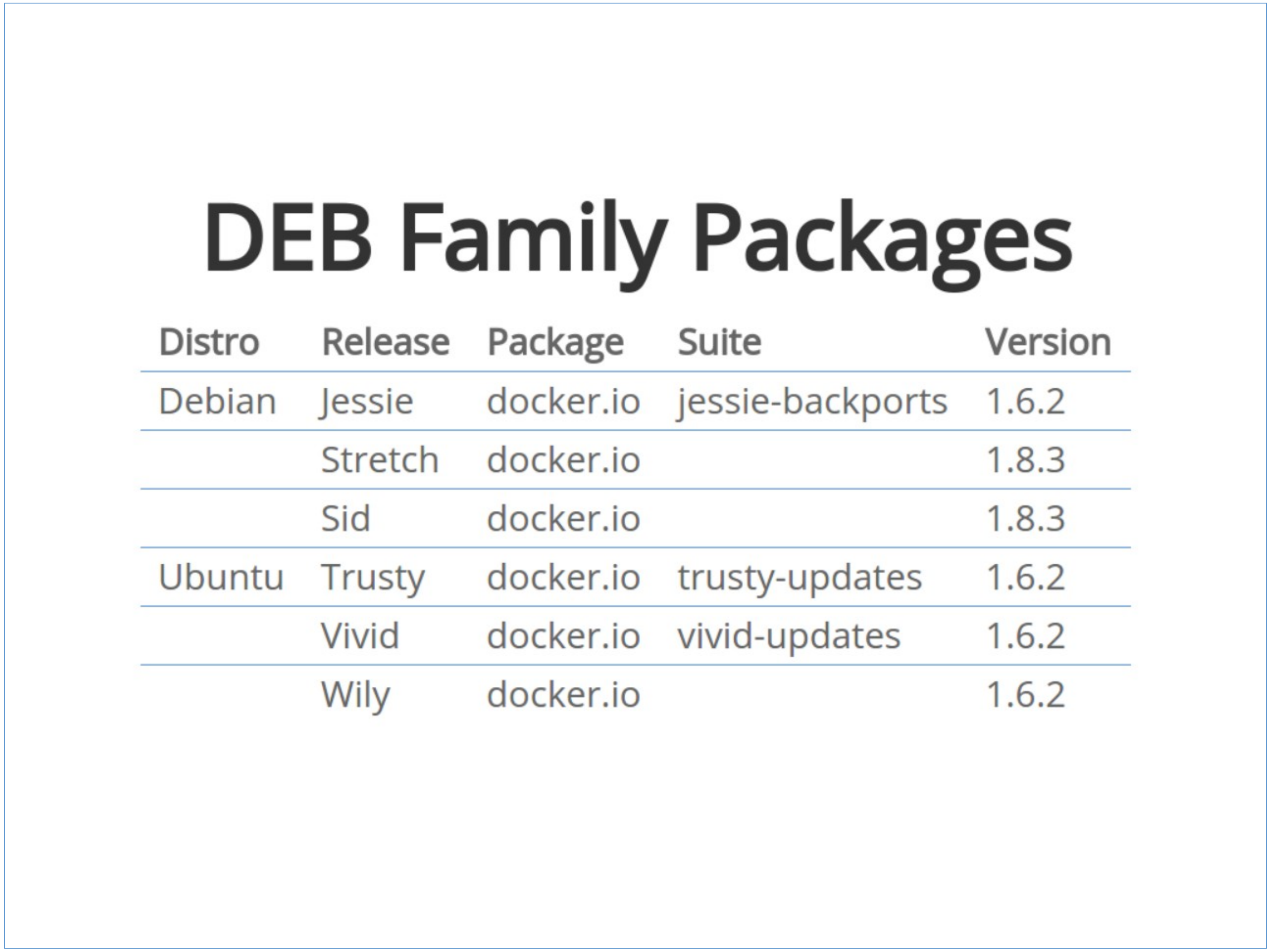
- For CentOS, Oracle Linux and Fedora, add the repository to the available YUM repositories using a here document, and then install the Docker Engine, e.g.:

```
# cat > /etc/yum.repos.d/docker.repo << EOF
[dockerrepo]
name=Docker Repository
baseurl=https://yum.dockerproject.org/repo/main/fedora/22
enabled=1
gpgcheck=1
gpgkey=https://yum.dockerproject.org/gpg
EOF
# yum install docker-engine
```

- For openSUSE, add the repository in the same way as above, but substituting `/etc/yum.repos.d` with `/etc/zypp/repos.d`, and altering the `baseurl` for openSUSE 13.2. Install the Docker Engine:

```
# zypper install docker-engine
```

Notes



DEB Family Packages

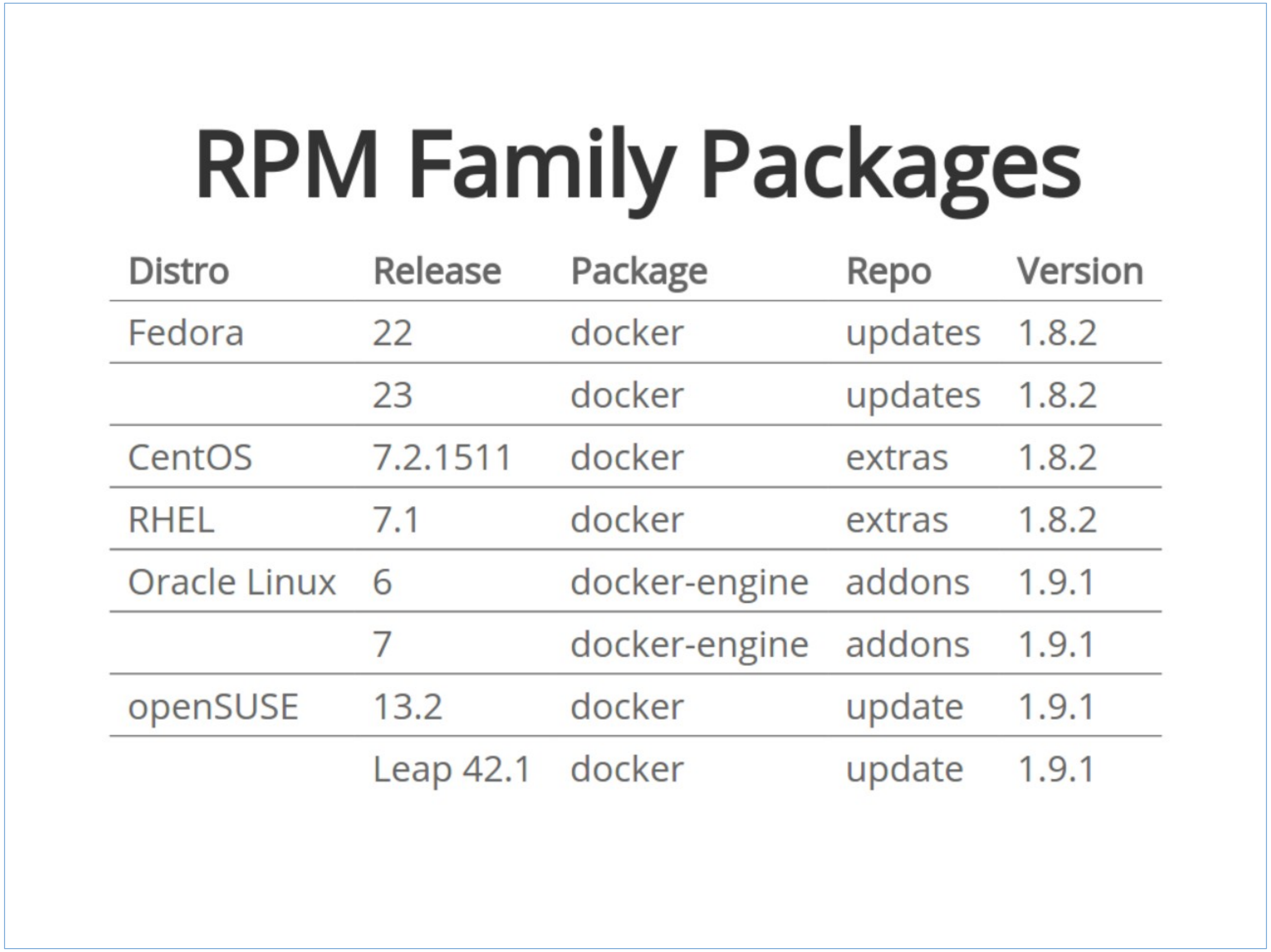
Distro	Release	Package	Suite	Version
Debian	Jessie	docker.io	jessie-backports	1.6.2
	Stretch	docker.io		1.8.3
	Sid	docker.io		1.8.3
Ubuntu	Trusty	docker.io	trusty-updates	1.6.2
	Vivid	docker.io	vivid-updates	1.6.2
	Wily	docker.io		1.6.2

DEB Family Packages

- If you are unable to make use of the repositories provided by the Docker project, DEB package-based Linux distributions provide native packages, but these are likely to be behind the current release:

Distribution	Release	Package	Suite	Version
Debian	Jessie	docker.io	jessie-backsports	1.6.2
	Stretch	docker.io		1.8.3
	Sid	docker.io		1.8.3
Ubuntu	Trusty Tahr	docker.io	trusty-updates	1.6.2
	Vivid Vervet	docker.io	vivid-updates	1.6.2
	Wily Werewolf	docker.io		1.6.2

Notes



RPM Family Packages

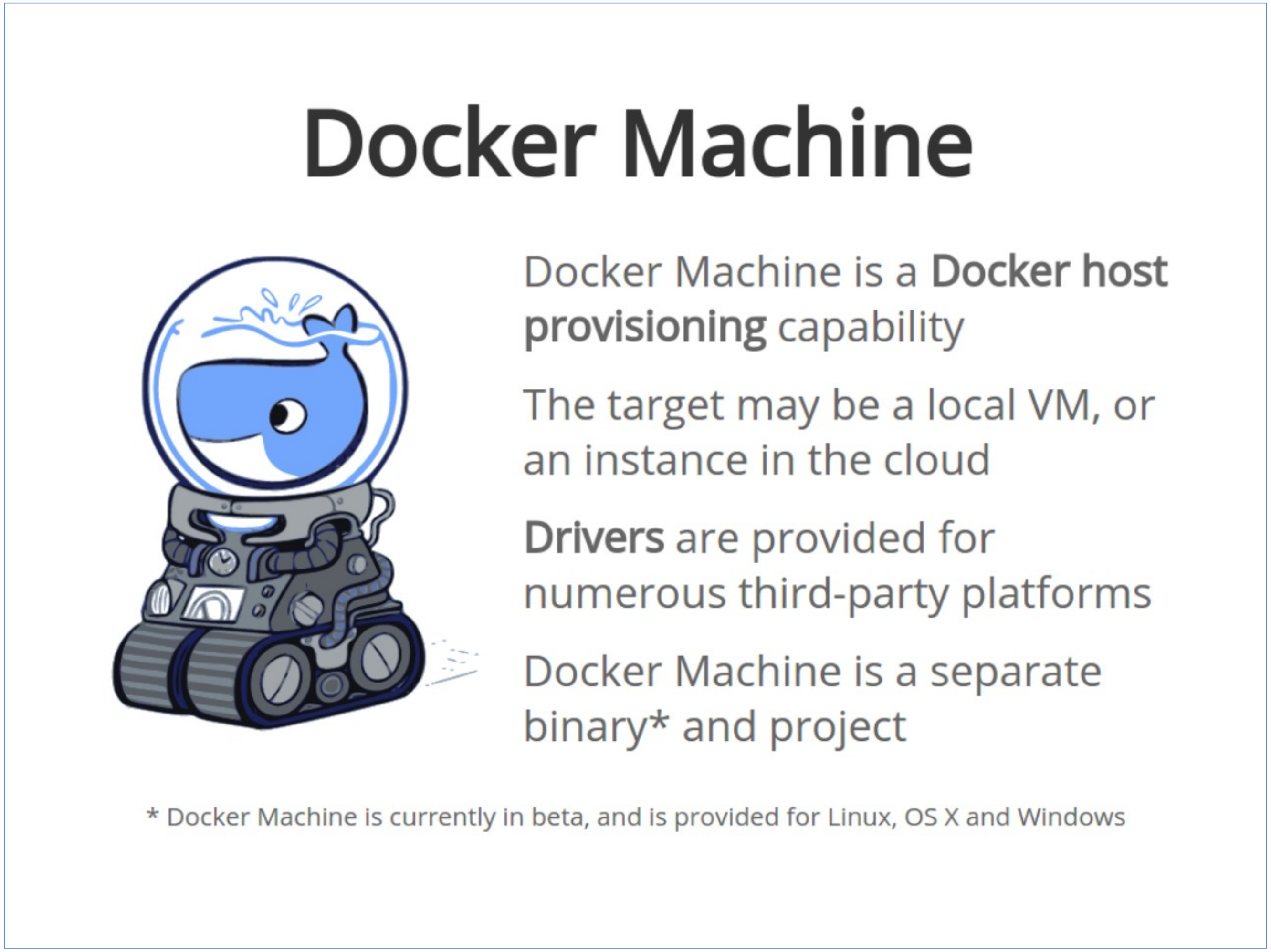
Distro	Release	Package	Repo	Version
Fedora	22	docker	updates	1.8.2
	23	docker	updates	1.8.2
CentOS	7.2.1511	docker	extras	1.8.2
RHEL	7.1	docker	extras	1.8.2
Oracle Linux	6	docker-engine	addons	1.9.1
	7	docker-engine	addons	1.9.1
openSUSE	13.2	docker	update	1.9.1
	Leap 42.1	docker	update	1.9.1

RPM Family Packages

- If you are unable to make use of the repositories provided by the Docker project, RPM package-based Linux distributions provide native packages, but these are likely to be behind the current release:

Distribution	Release	Package	Repo	Version
Fedora	22	docker	updates	1.8.2
	23	docker	updates	1.8.2
CentOS	7.1.1503	docker	extras	1.8.2
RHEL	7.1	docker	extras	1.8.2
Oracle Linux	6	docker-engine	addons	1.8.3
	7	docker-engine	addons	1.8.3
openSUSE	13.2	docker	update	1.8.3
	Leap 42.1	docker	update	1.8.2

Notes



Docker Machine

Docker Machine is a **Docker host provisioning** capability

The target may be a local VM, or an instance in the cloud

Drivers are provided for numerous third-party platforms

Docker Machine is a separate binary* and project

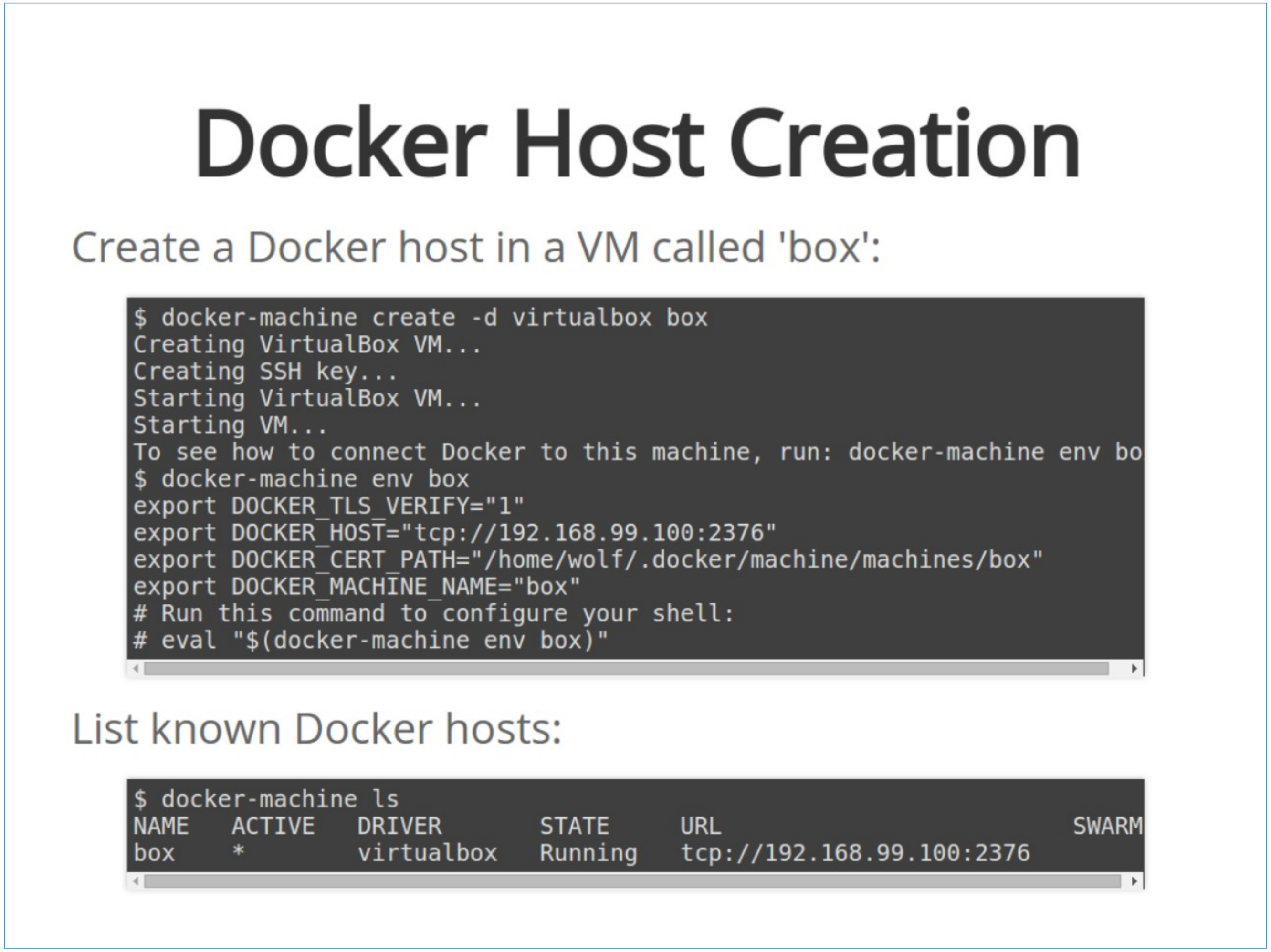
* Docker Machine is currently in beta, and is provided for Linux, OS X and Windows

Docker Machine

- Docker Machine is a Docker provisioning capability for remote hosts, where the host may be a local virtual machine or a cloud instance
- Docker Machine takes care of creating the host, installing the Docker Engine, and establishing secure communication between the local Docker client and the newly created, remote Docker host
- Docker host creation is achieved through the use of drivers which are specific to the target environment (e.g. Microsoft Azure)
- Up to version 0.5.0, drivers were integral to the Docker Machine code, but since version 0.5.0 a plugin architecture exists, with each driver being a separate binary
- The separation of the core code from the driver enables third parties to create drivers without the need to merge their code with the core project code
- The Docker Machine project will ship the following drivers with forthcoming releases, with the community and other third parties taking responsibility for other drivers: `virtualbox`, `amazonec2`, `azure`, `digitalocean`, `generic` (for use with an existing virtual machine or cloud instance) and `none` (for changing which Docker host the local client addresses)
- In addition to creating Docker hosts, the Docker Machine CLI provides a rich set of commands for retrieving information regarding the currently addressed, remote Docker host, as well as for manipulating the remote Docker host (e.g. upgrading it to a new version of Docker)
- Docker Machine is provided natively as a separate binary for Linux, Windows and OS X, but is still in beta
- Docker Machine can be downloaded from the docker/machine repository on GitHub:

  `https://github.com/docker/machine/releases`

Notes



Docker Host Creation

Create a Docker host in a VM called 'box':

```
$ docker-machine create -d virtualbox box
Creating VirtualBox VM...
Creating SSH key...
Starting VirtualBox VM...
Starting VM...
To see how to connect Docker to this machine, run: docker-machine env bo
$ docker-machine env box
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.100:2376"
export DOCKER_CERT_PATH="/home/wolf/.docker/machine/machines/box"
export DOCKER_MACHINE_NAME="box"
# Run this command to configure your shell:
# eval "$(docker-machine env box)"
```

List known Docker hosts:

```
$ docker-machine ls
NAME    ACTIVE   DRIVER       STATE     URL                         SWARM
box     *        virtualbox   Running   tcp://192.168.99.100:2376
```

Docker Host Creation

- The following example illustrates the use of Docker Machine, by establishing a Docker host on VirtualBox, using the `create` sub-command:

```
$ docker-machine create -d virtualbox box
Creating VirtualBox VM...
Creating SSH key...
Starting VirtualBox VM...
Starting VM...
To see how to connect Docker to this machine, run: docker-machine env box
$ docker-machine env box
export DOCKER_TLS_VERIFY=”1”
export DOCKER_HOST=”tcp://192.168.99.100:2376”
export DOCKER_CERT_PATH=”/home/wolf/.docker/machine/machines/box”
export DOCKER_MACHINE_NAME=”box”
# Run this command to configure your shell:
# eval ”$(docker-machine env box)”
```

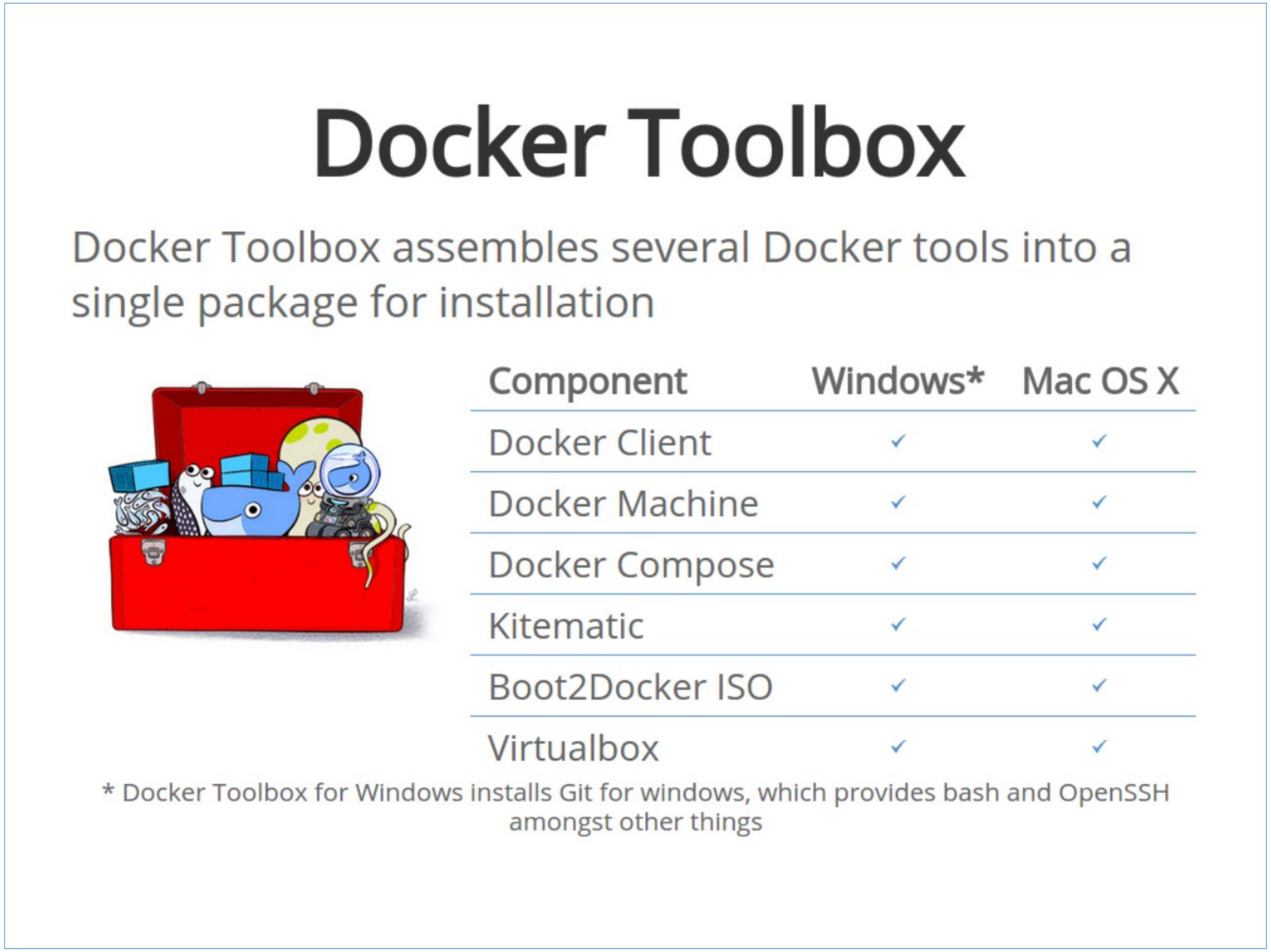
- We can see our new Docker host, using the `docker-machine ls` command:

```
$ docker-machine ls
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM
box       *        virtualbox   Running   tcp://192.168.99.100:2376
```

- The output tells us that the currently active Docker host addressable by the client, is running at the URL specified, is called `box`, is a Virtualbox VM, and is not part of a Docker Swarm cluster
- To test whether it's possible to communicate with the Docker host that's been created, we can use the command `docker info`:

```
$ docker info | grep Name
Name: box
```

Notes



Docker Toolbox

Docker Toolbox assembles several Docker tools into a single package for installation

Component	Windows*	Mac OS X
Docker Client	✓	✓
Docker Machine	✓	✓
Docker Compose	✓	✓
Kitematic	✓	✓
Boot2Docker ISO	✓	✓
Virtualbox	✓	✓

* Docker Toolbox for Windows installs Git for windows, which provides bash and OpenSSH amongst other things

Docker Toolbox

- Docker Toolbox is a bundle of Docker-related tools, packaged in an installer for convenience
- It's primarily for Windows and OS X platforms, where the Docker Engine does not run natively, although there are plans for a Linux variant of Docker Toolbox
- The following components are installed by Docker Toolbox:

  Docker Client: the client component of the Docker Engine, which provides a CLI for interacting with the server
  Docker Machine: for provisioning remote Docker hosts
  Docker Compose: a tool for orchestrating a group of containers that comprise a logical application
  Kitematic: a tool which provides a graphical user interface to the Docker Client CLI
  Boot2Docker ISO: an ISO for creating a minimal Linux virtual machine for hosting Docker
  Virtualbox: a hypervisor for running a local Linux virtual machine

- Subsequent to new Docker releases, a new version of the Docker Toolbox is made available, which allows for selectively upgrading toolbox components
- The Windows version of Docker Toolbox also gives the option of installing Git for Windows, which provides a number of tools which are useful when using Docker (e.g. bash, OpenSSH)
- The Docker Toolbox installer can be downloaded from the `docker/toolbox` repository on GitHub:

  https://github.com/docker/toolbox/releases

- Double click the installer to install the contents of the Docker Toolbox

Notes