

Repackaging in Docker Containers

Virtualization Spectrum: Containers - VMs



support@advancedinstaller.com

Victor Ciura
Principal Engineer

 [@ciura_victor](https://twitter.com/ciura_victor)

Abstract

A local VM is not always the obvious choice since setting it up is quite a hassle for many of us - you have RAM and disk space requirements plus the OS image to consider. User account setup and network configuration can sometimes be a pain; so does managing OS snapshots (clean state) & updates.

At the other end, setting up a Docker container is easy and has a minimal footprint, but has runtime restrictions.

We talk about a *spectrum* and not about a unique solution because companies have different needs and limitations on what they need to do in various repackaging scenarios.

Things we'll cover in this webinar:

- support for repackaging and testing in Docker containers
- zero-configuration: limitations and alternatives
- containers vs. hypervisors
- strategies for reducing friction while running apps in sandbox environments
- common gotchas

Agenda

Intro

Advanced Repackager

Containers vs VMs

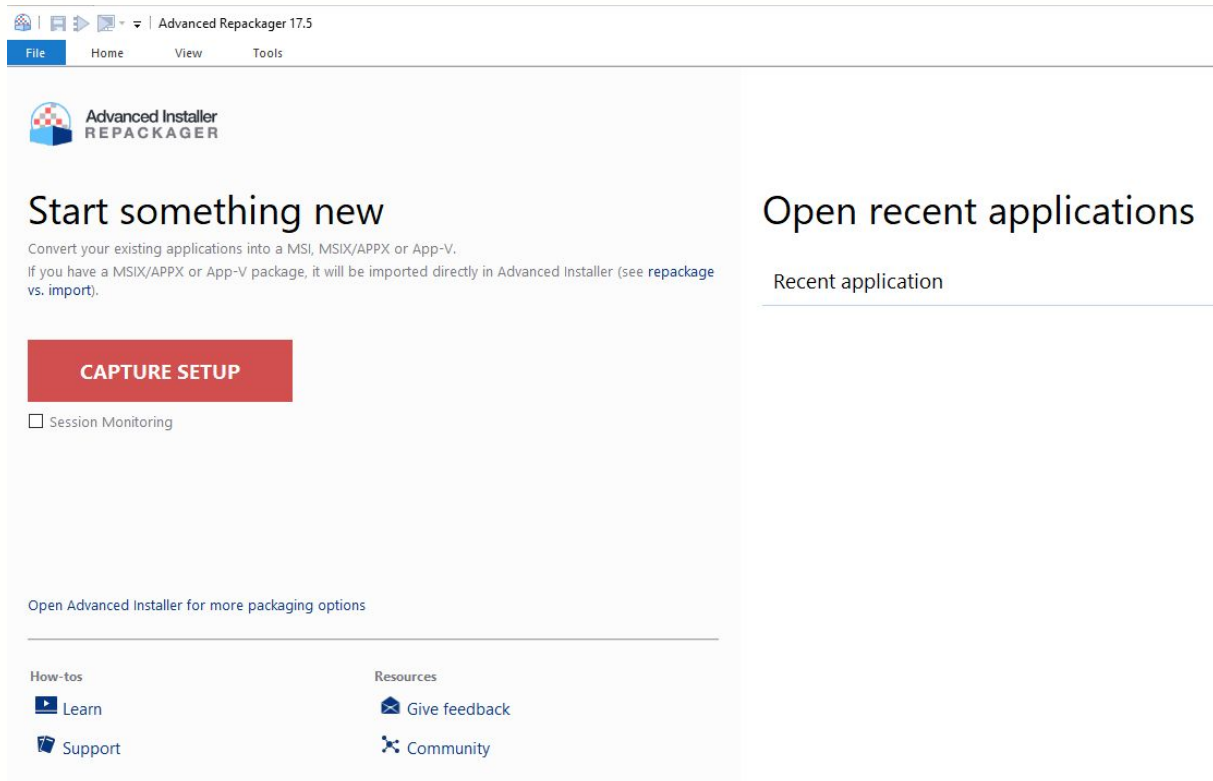
Advanced Installer integrations for Hyper-V, VMware and Docker

Demos

Q & A

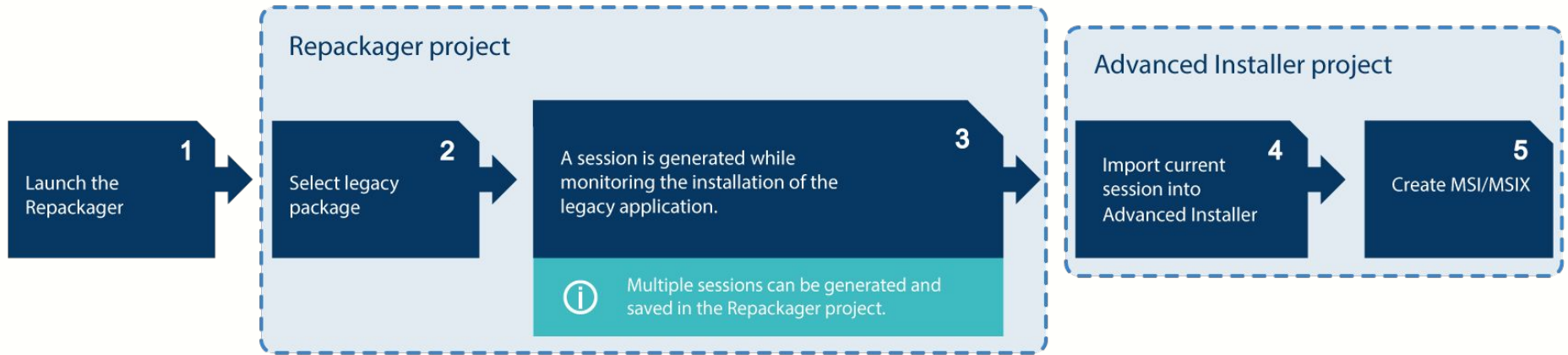
Repackager Overview

- MSI, App-V & MSIX
- High-level constructs
- Project-based



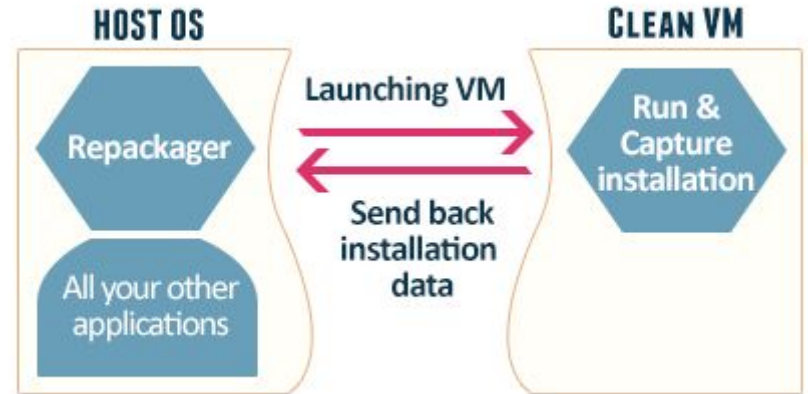
The screenshot shows the 'Advanced Installer REPACKAGER' application window. The title bar reads 'Advanced Repackager 17.5'. The menu bar includes 'File', 'Home', 'View', and 'Tools'. The main content area features the application logo, the heading 'Start something new', and a sub-heading 'Convert your existing applications into a MSI, MSIX/APPX or App-V. If you have a MSIX/APPX or App-V package, it will be imported directly in Advanced Installer (see [repackage](#) vs. [import](#)).' Below this is a prominent red 'CAPTURE SETUP' button and a checkbox for 'Session Monitoring'. At the bottom, there are links for 'How-tos' (Learn, Support) and 'Resources' (Give feedback, Community). On the right side of the interface, there is a section titled 'Open recent applications' with a sub-section for 'Recent application'.

Package Lifecycle



Repackager Targets

- Local & Server
- Remote machines
- Hyper-V
- VMware (Workstation, Player, vSphere)
- Docker containers



Repackager Targets

Each kind of repackaging environment has its strengths & weaknesses.

- Repackaging on the local PC is fast, but not a great idea (non-deterministic, pollutes the environment)

Repackager Targets

- A **local VM** is not always the obvious choice since setting it up is quite a hassle for many of us:
 - slower than direct hardware
 - you have RAM and disk space requirements
 - providing the OS image
 - activation & licensing
 - user account setup
 - network configuration can sometimes be a pain (NAT, bridged connections, subnets)
 - managing OS snapshots (clean state) & updates

Repackager Targets

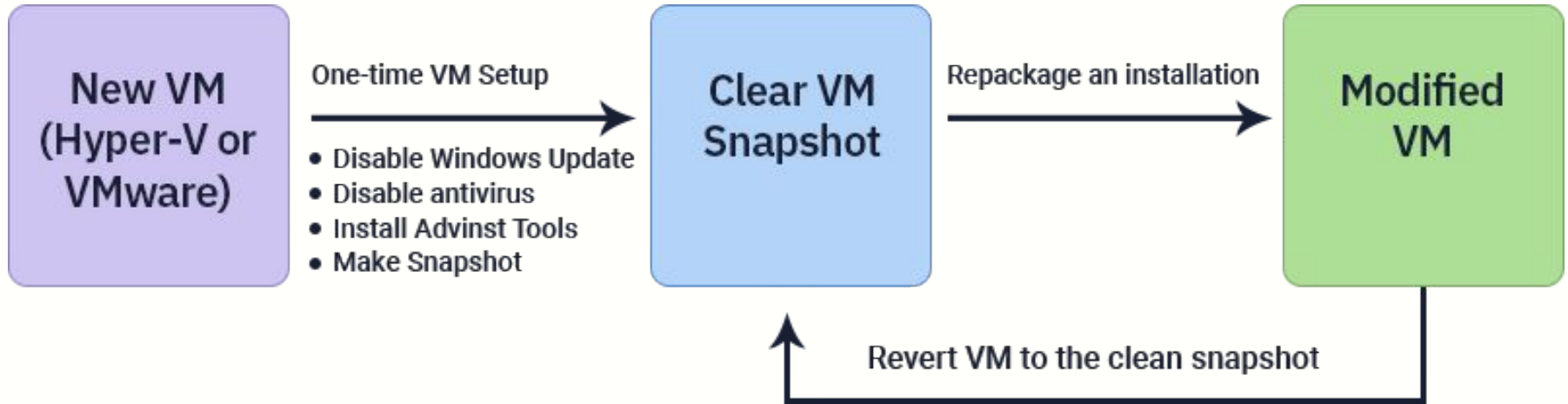
- On the other end, setting up a **Docker container** is easy and has a minimal footprint, but has runtime restrictions
 - less control over the guest OS setup
 - access to limited OS versions (legacy systems)
 - no desktop environment
 - no UI interaction (both package and app)
 - x86/x64 hassles

Repackager Targets

Each kind of repackaging environment has its strengths & weaknesses.

We acknowledge this as a **spectrum** and not as a unique solution because companies have different needs and limitations on what they need to do in various repackaging scenarios.

Hyper-V and VMware Integrations



Repackager Automation

- Automate VM management
- Invoking the package UI
- Script from manual steps



Repackaging in Docker

- Containers vs. hypervisors
- Limitations & alternatives
- Strategies for reducing friction while running apps in sandbox environments
- Common gotchas

Repackaging in Docker

Some advantages:

- local-machine like performance
- but not polluting the PC environment
- zero configuration
- good collection of curated OS images (Docker Hub)
- no OS activation or licensing
- no user account setup
- no network adapter configuration needed
- always a fresh container (from the same image)

Docker Settings

General

Automatically check for updates **TEAM**

[Paid Team plans](#) allow IT-managed organizations to disable checking for updates.

Start Docker Desktop when you log in

Expose daemon on tcp://localhost:2375 without TLS

Exposing daemon on TCP without TLS helps legacy clients connect to the daemon. It also makes yourself vulnerable to remote code execution attacks. Use with caution.

Use the WSL 2 based engine

WSL 2 provides better performance than the legacy Hyper-V backend. [Learn more.](#)

Send usage statistics

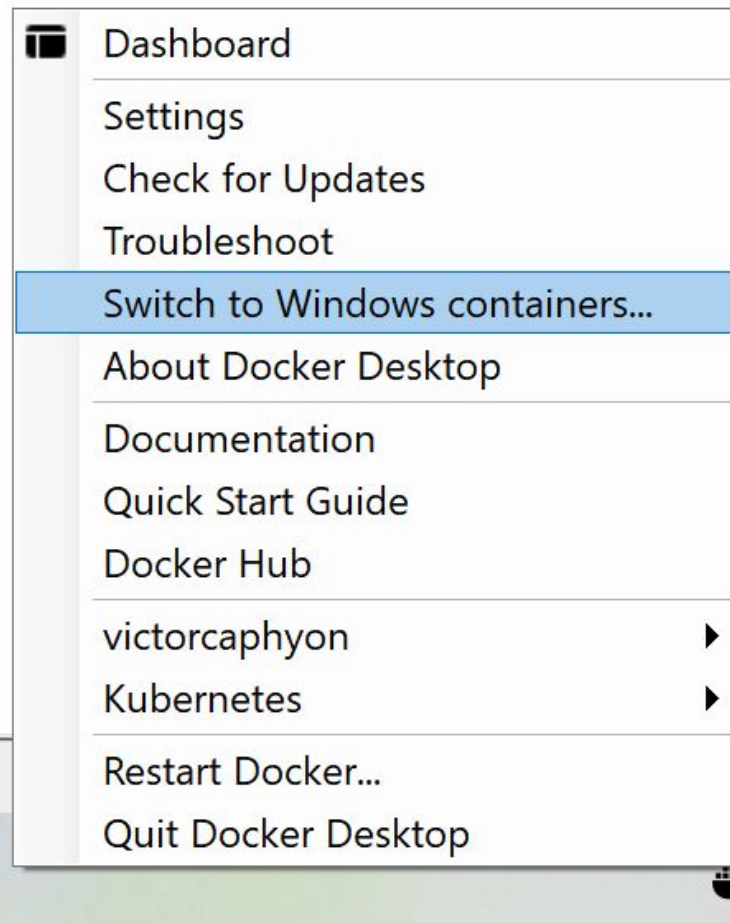
Send error reports, system version and language as well as Docker Desktop lifecycle information (e.g., starts, stops, resets).

Show weekly tips

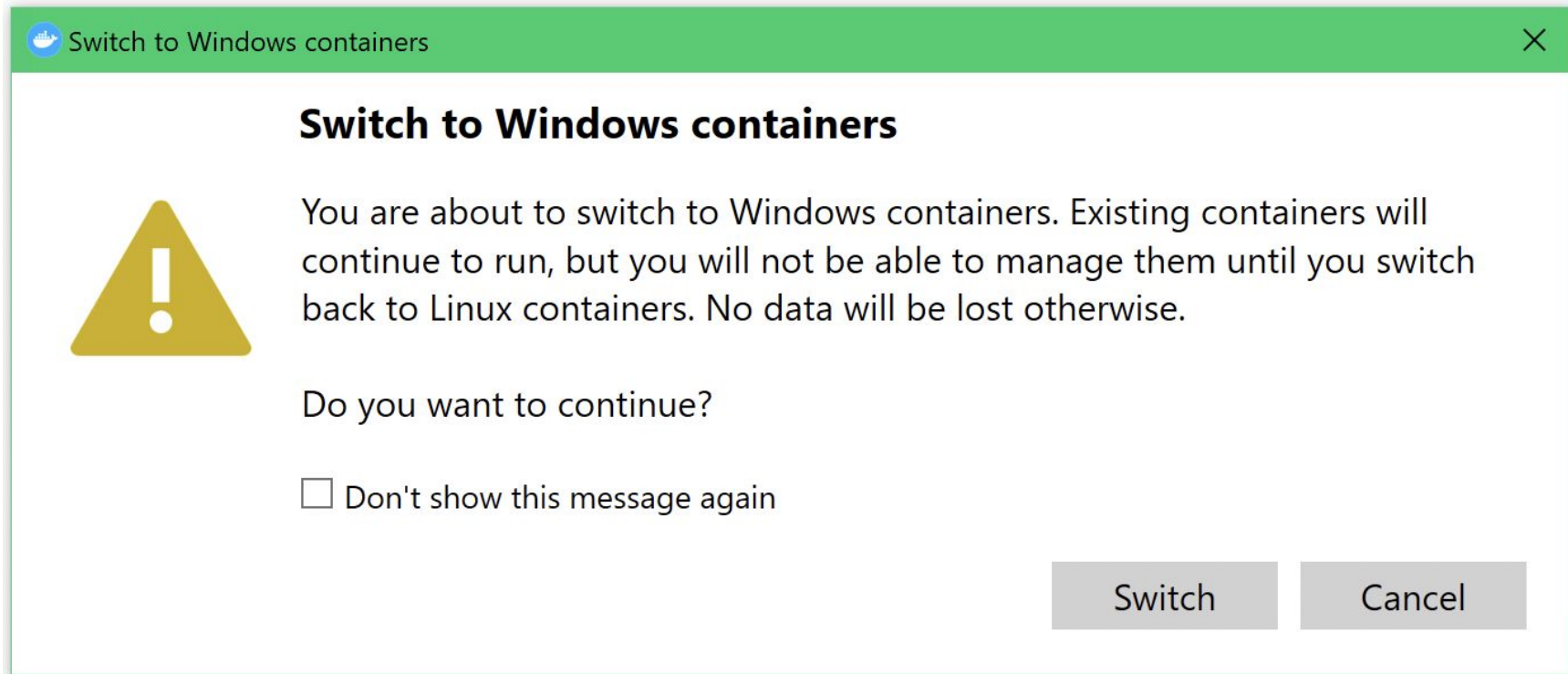
Open Docker Desktop dashboard at startup

Switch to Win Containers

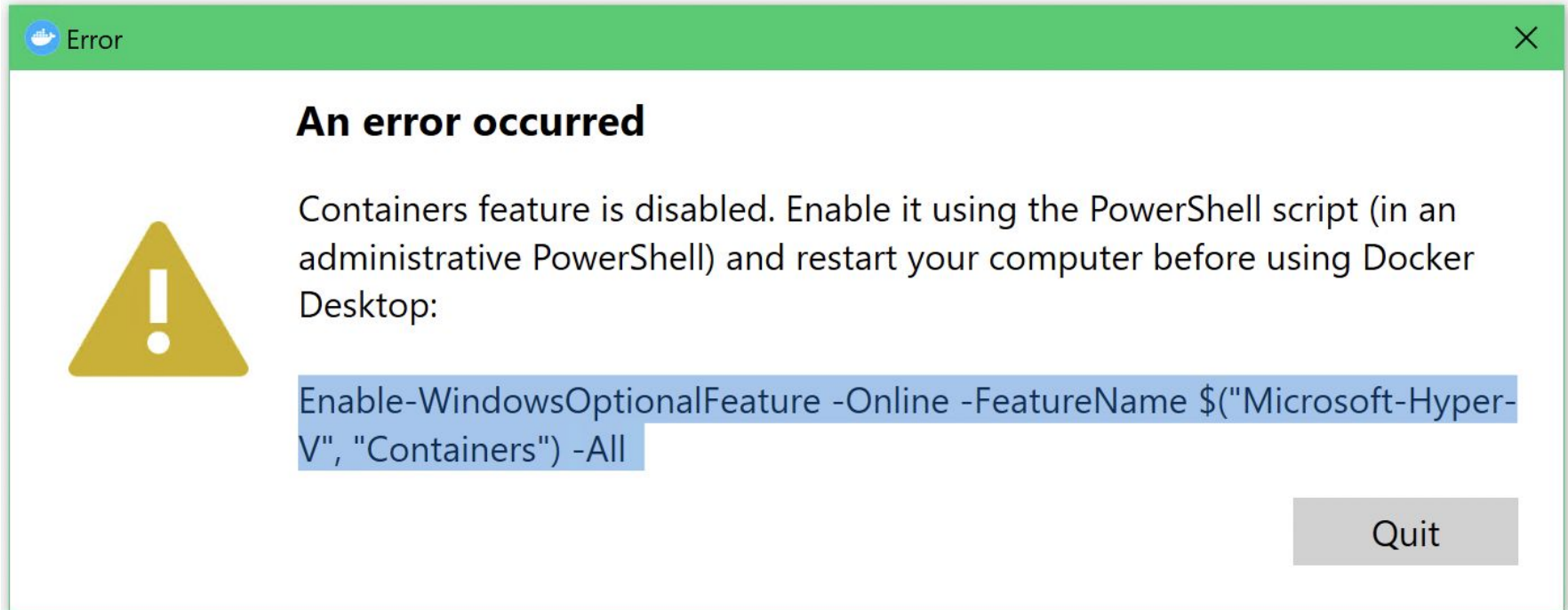
Let us take care of
all the config minutia...



Default: Linux Containers (WSL2)



Windows Features: Turn ON Containers



Docker Settings

We do all this work,
so you can focus on your task

General

- Automatically check for updates **TEAM**
[Paid Team plans](#) allow IT-managed organizations to disable checking for updates.
- Start Docker Desktop when you log in
- Expose daemon on tcp://localhost:2375 without TLS
Exposing daemon on TCP without TLS helps legacy clients connect to the daemon. It also makes yourself vulnerable to remote code execution attacks. Use with caution.
- Use the WSL 2 based engine
WSL 2 provides better performance than the legacy Hyper-V backend. [Learn more.](#)
- Send usage statistics
Send error reports, system version and language as well as Docker Desktop lifecycle information (e.g., starts, stops, resets).
- Show weekly tips
- Open Docker Desktop dashboard at startup

Isolation Modes

Windows containers offer two distinct modes of runtime isolation:

- process isolation
- Hyper-V isolation

Containers running under both isolation modes are created, managed, and *function identically*.

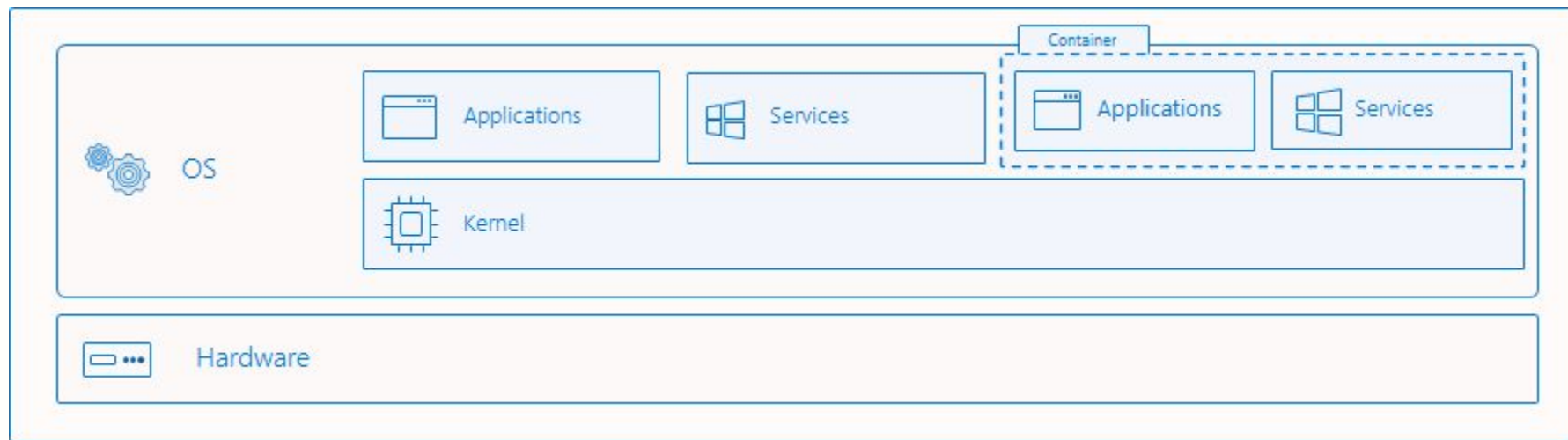
They also produce and consume the same container **images**.

Isolation Modes

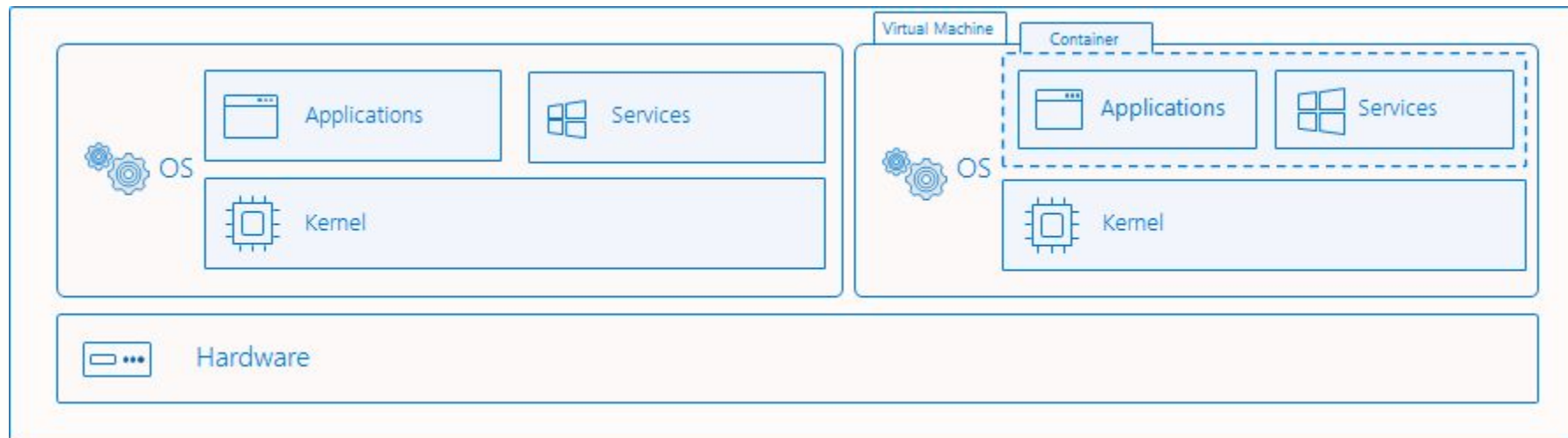
The **difference** between the isolation modes is to what degree of **isolation** is

- between the container and the host operating system
- between the container and all of the other containers running on that host

Process Isolation



Hyper-V Isolation



Default Isolation Mode

Windows containers running on **Windows Server** *default* to running with **process isolation**

Windows containers running on **Windows 10 Pro/Enterprise** *default* to running with **Hyper-V isolation**

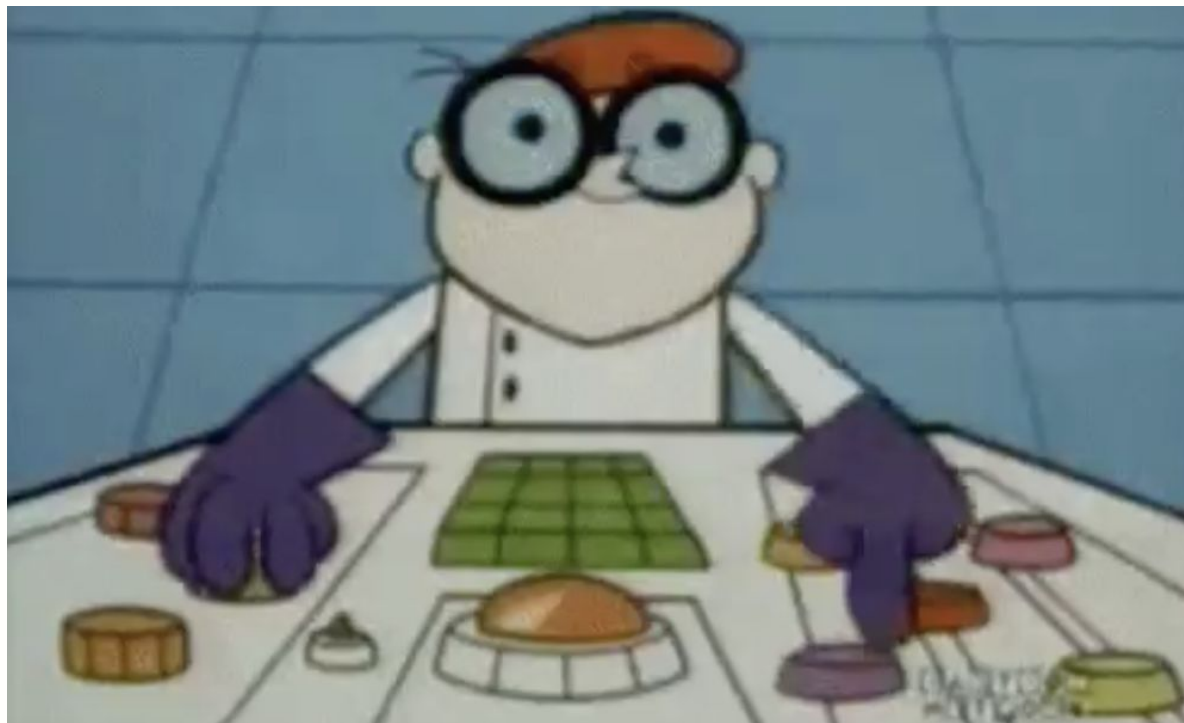
 Running with *process isolation* on Windows 10 Pro/Enterprise is meant for **development/testing**.

You can be explicit about it

```
> docker run -it --isolation=hyperv  
mcr.microsoft.com/windows/servercore:latest cmd
```

```
> docker run -it --isolation=process  
mcr.microsoft.com/windows/servercore:latest cmd
```

Demo Time



Next Steps

Testing packages and apps in Docker containers (headless)

Let us know what you would like to see here.

support@advancedinstaller.com

Learn more

www.advancedinstaller.com/repackager

Q & A



Ask us anything

support@advancedinstaller.com

Repackaging in Docker Containers

Virtualization Spectrum: Containers - VMs



support@advancedinstaller.com

Victor Ciura
Principal Engineer

 [@ciura_victor](https://twitter.com/ciura_victor)