
# Friedliche Koexistenz

## Linux- und Windows-Container parallel in Docker unter Windows

**Bereits seit dem Herbst-Update von Windows 10 sollten Linux-Container parallel zu Windows-Containern laufen. Mit einigen Monaten Verspätung gelingt das jetzt – zumindest in der Testversion von Docker. Was Entwickler und Administratoren erwartet.**

**Von Jan Mahn**

Kaum eine Software mit großer Verbreitung auf Desktops und Servern wird so rasant weiterentwickelt wie die Container-Plattform Docker. Nahezu jedes Update bringt neben Fehlerbehebungen auch neue Funktionen mit und das Projekt wächst immer mehr in die Breite. Das liegt auch daran, dass viele große Firmen wie IBM, HP und Cisco an einzelnen Komponenten mitarbeiten und das System insbesondere für ihre Clouds weiterentwickeln.

Auch Microsoft hat erkannt, dass Containerisierung ein attraktiver Markt ist und hat seine Unterstützung von Docker stückweise ausgebaut. Dabei geht es darum, Linux- wie Windows-Serverdienste gleichberechtigt auf einem Host laufen zu lassen. Das Ziel dahinter ist klar: Administratoren und Entwickler sollen unter Windows 10 in einer gewohnten Umgebung ihre Dienste in Containern entwickeln, testen und kontrollieren. Windows Server soll dann zur bevorzugten Plattform für den produktiven Betrieb der fertigen Container werden – auf dem eigenen Server oder in der (Azure-)Cloud. Daher umfasst die Lizenz von Windows

Server 2016 auch die Enterprise-Version von Docker.

## Gemischte Umgebung

Wer Windows-Programme im Container betreiben will, hat zwei Images zur Auswahl: „microsoft/windowsservercore“ (10,4 GByte groß) ist ein vollständiger Windows Server ohne grafische Oberfläche. Die wesentlich handlichere Alternative „microsoft/nanoserver“ ist nur 1,1 GByte groß. Dafür fehlen ihr viele Windows-Funktionen wie die Möglichkeit, MSI-Pakete zu installieren.

Wer Linux-Container betreiben wollte, musste bisher Docker in den Modus für Linux-Container umschalten und die Windows-Container damit stoppen. Docker startete früher eine Hyper-V-Maschine mit dem Betriebssystem MobyLinux, in dem es den Container ausführte und die auch dann weiterlief, wenn bereits alle Container abgeschaltet waren. Die Docker-Umgebung war damit untauglich, um zum Beispiel das Zusammenspiel von Microsofts Webserver IIS (der nur unter Windows läuft) hinter einem Varnish Cache (den es nur für Linux gibt) auf einer Maschine zu testen.

Groß war daher das Interesse, als zwei Microsoft-Mitarbeiter im September 2017 während der Ignite-Konferenz vorführten, wie ein Linux- und ein Windows-Container parallel unter Docker liefen. Groß die Freude, als die beiden das Feature bereits für das Fall Creators Update im September ankündigten. Den Zeitplan haben Microsoft und Docker stillschweigend etwas angepasst und schrittweise an der Umsetzung gearbeitet. Bereits seit Dezember 2017 gibt es die experimentelle Funktion „Linux Containers on Windows (LCOW)“. Damit bootet Docker die Linux-Container nicht mehr standardmäßig in einer Hyper-V-Maschine, sondern startet für jeden Container einen Windows-Prozess, in dem das abgespeckte Linux „LinuxKit“ läuft. Dabei kann man von einem Linux-Subsystem sprechen, es handelt sich aber nicht um das „Windows Subsystem for Linux“, mit dem man seit Windows 1709 eine Linux-Bash unter Windows ausführen kann [1]. Ende Januar 2018 wurde Docker 18.02 als Vorabversion (im sogenannten Edge-Release) fertig, das den versprochenen Mischbetrieb endlich ermöglicht.

## Experimentell

Um die neue Funktion auszuprobieren, muss Ihr PC einige Voraussetzungen erfüllen. Verwenden Sie ein aktuelles Windows 10 Version 1709, das nicht in der Insider-Preview registriert ist. In unserem Test wollte Docker mit der neuesten Preview-Version von Windows nicht starten. Laden Sie die Vorabversion „Docker für Windows (Edge)“ bei Docker herunter (ct.de/yd7p) und installieren Sie die Software. Dabei wird das Windows-Feature Hyper-V weiterhin installiert, kommt aber nur noch im reinen Linux-Container-Modus zum Einsatz.

Nachdem Docker hochgefahren ist, öffnen Sie die Eingabeaufforderung oder die PowerShell und führen Sie den Befehl `docker version` aus. Die Ausgabe sollte unter dem Oberpunkt „Server“ die Zeile „Experimental: true“ liefern. Ist das nicht der Fall, öffnen Sie die Einstellungen von Docker (Rechtsklick auf den Wal in der Taskleiste, dann „Settings“). Im Menüpunkt „Daemon“ aktivieren Sie die experimentellen Features. Der Parallelbetrieb funktioniert nur im Windows-Container-Modus. Sollte der Wert für „OS/Arch“ nicht „windows/amd64“ anzeigen, öffnen Sie das Kontextmenü hinter dem Wal und wählen Sie „Switch to Windows containers“.

## Nur fast fertig

Starten Sie zum Test zunächst einen Windows-Container, zum Beispiel den leeren Nanoserver im interaktiven Modus mit dem Befehl

```
docker run -it microsoft/nanoserver
```

Docker beginnt, das Image herunterzuladen und Sie landen auf der Kommandozeile innerhalb des Containers.

Lassen die dieses Fenster geöffnet und öffnen Sie ein zusätzliches Kommandozeilenfenster. Versuchen Sie, den beliebten Linux-Werkzeugkasten BusyBox zu starten. Der Befehl

```
docker run -it busybox
```

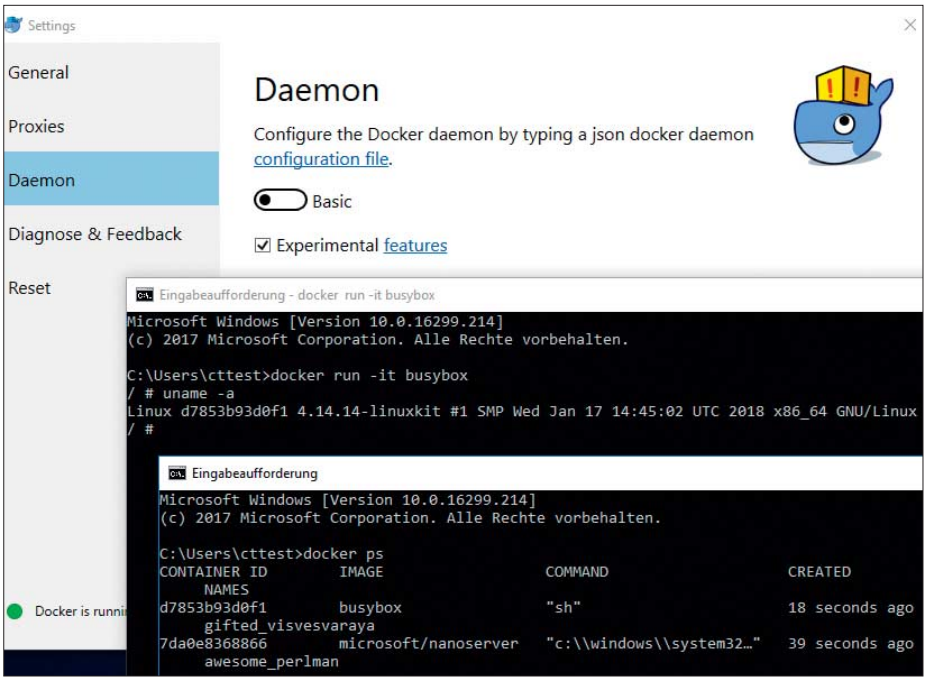
schlägt mit dem Hinweis fehl, dass es kein Image mit diesem Namen im Repository gebe. Damit Docker weiß, dass es sich um ein Linux-Image handelt, benötigt es laut Release Notes den Parameter `--platform=linux` – doch auch das führt nicht zum gewünschten Ergebnis. Offenbar wurde der Parameter für `docker run` noch nicht eingebaut. Laden Sie das Image stattdessen zunächst mit

```
docker pull --platform=linux busybox
```

herunter und starten Sie den Container anschließend mit dem Befehl

```
docker run --it busybox
```

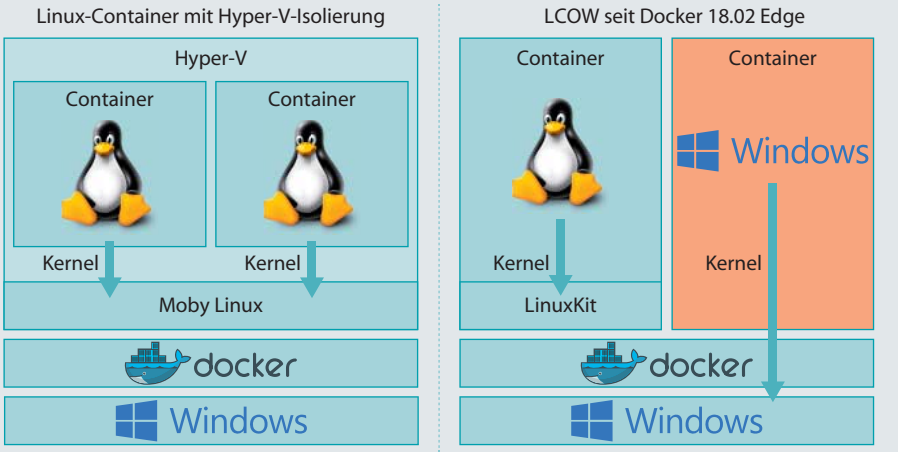
Sie landen in der Bash des Linux-Containers und können beispielsweise mit `uname -a` das zugrunde liegende Betriebssystem




**Linux- und Windows-Container laufen parallel unter Docker. Damit können Entwickler beispielsweise gemischte Umgebungen testen.**

## Linux-Container unter Windows

Die Unterstützung für Linux-Container wurde weiterentwickelt. Seit der Vorabversion 18.02 laufen sie neben Windows-Containern auf einem Host.




anzeigen – LinuxKit mit Kernel 4.14.14. Neben `docker pull` versteht auch `docker build` bereits den neuen Parameter `--platform=linux`. Sie können also auch eigene Dockerfiles schreiben, um daraus Container-Images zu erstellen.

### Noch kein Compose

Wer mehr als einen Container gleichzeitig und im Zusammenspiel betreibt, verliert schnell die Lust, die gesamte Umgebung per Hand über die Kommandozeile oder per Batch-Datei zu starten – schließlich müssen Sie Ports freigeben, Namen zuordnen, Volumes anhängen und das Netzwerk definieren. Um den Zusammenbau solcher Umgebungen zu erleichtern, hat sich `docker-compose` etabliert. Damit definieren Sie Container-Umgebungen in einer YAML-Datei und starten sie mit der Zeile `docker-compose up`. Die schlechte Nachricht: Auch `docker-compose` ist noch nicht ganz auf die neuen Mischumgebungen eingestellt und der Versuch scheitert, ein Linux-Image zu verwenden. Noch fehlt die Möglichkeit, das Platform-Attribut in der YAML-Datei zu setzen. Bei GitHub gibt es im Docker-Compose-Repository bereits ein Issue (siehe ct.de/yd7p), in dem darum gebeten wird, die Funktion nachzurüsten. Der Docker-Entwickler „shin“ hat bereits vorsichtig angekündigt, dem Wunsch in Version 1.20 nachkommen zu wollen (die aktuelle Version ist 1.18).

Vorerst können Sie das Problem der fehlenden Compose-Unterstützung umgehen, indem Sie alle Images, die in der Umgebung vorkommen, zunächst per `docker pull` oder `docker build` erstellen oder herunterladen und erst dann die Umgebung mit `docker-compose up` starten. Findet docker die benötigten Images auf der Festplatte, startet die Umgebung – für den Produktiveinsatz ist das natürlich, wie das gesamte Feature, noch nicht geeignet.

### Neuer Steuermann an Bord

Nicht nur das Unternehmen Docker Inc. entwickelt Docker weiter und so begann Google 2014, eine eigene Anwendung zur Verwaltung von Docker-Containern auf verteilten Host-Rechnern zu entwickeln. Diese stand in Konkurrenz zu Dockers eigener Lösung `docker swarm`. Kubernetes, Griechisch für Steuermann, hat sich zu einem weit verbreiteten Werkzeug entwickelt und ist mittlerweile an die Cloud Native Computing Foundation übergeben worden. In einer Kubernetes-Infrastruktur hat ein Server mit der Rolle „Master“ die Aufgabe, die benötigten Dienste in Containern auf sogenannten „Nodes“ auszuführen. Der Administrator kann bestimmen, wie viele Instanzen („Replicas“) jeder Dienst haben soll. Fällt ein Node aus, startet ein Master die Container auf einem anderen Node.

Im letzten Jahr gab das Unternehmen Docker Inc. bekannt, Kubernetes in Docker zu integrieren. Mit Docker Version 18.02 ist Kubernetes jetzt auch in Docker for Windows angekommen – wenn auch in einer sehr frühen Phase. Der Befehl `kubectl` kann bisher aber nur Umgebungen mit einer einzelnen Maschine einrichten.

Um Kubernetes zu aktivieren, müssen Sie aktuell den gemischten Modus verlassen und im Menü (Rechtsklick auf den Wal) in den Linux-Modus wechseln. Anschließend finden Sie unter „Settings“ den neuen Eintrag „Kubernetes“, in dem Sie die Funktion aktivieren können. Die Installation gibt leider keine Rückmeldung über den Fortschritt und zeigt auch keine Fehler. Der Kubernetes-Server läuft selbst in einem Container, wird aber von Befehlen wie `docker ps` nicht angezeigt. Um das zu ermöglichen, aktivieren Sie den Haken „Show system containers“. In unserem Test funktionierte die Installation nicht, der Status blieb auf „Kubernetes is starting“. GitHub-Issue 1649 (ct.de/yd7p) zeigt, dass es sich nicht um ein Einzelproblem handelt. Sollten Sie selbst auf Probleme in der Testversion stoßen und diese bei GitHub melden wollen, sollten Sie unter „Diagnose und Feedback“ einen Satz mit Diagnosedaten hochladen und die zugehörige ID veröffentlichen.

### Andere Welt

Parallele Linux- und Windows-Container funktionieren technisch, ganz ohne Nachdenken gelingt das aber nicht. Probleme gibt es immer dann, wenn Sie bestehende Container-Projekte übernehmen wollen. Die meisten Entwickler verwenden in ihren Hinweisen zur Inbetriebnahme Befehle, die nur unter Linux funktionieren. Um eine lange Zeile in der Linux-Kommandozeile umzubrechen, nutzen sie den Backslash „\“, der unter Windows nicht funktioniert. Um solche Aufrufe zu übernehmen, entfernen Sie die Umbrüche und setzen Sie alles zu einer Zeile zusammen. Auch Pfadangaben in Aufrufen und Compose-Files müssen Sie per Hand an die Windows-Welt anpassen. Solche Schwierigkeiten werden auch die nächsten Versionen von Docker nicht lösen können – anders als viele der über 700 Probleme, die im GitHub-Repository „docker/for-win“ auf Bearbeitung warten. *(jam@ct.de)* c't

#### Literatur

[1] Jan Mahn, Peter Siering, Microsoft hat Linux fertig, Das Windows Subsystem für Linux installieren und einrichten, c't 22/2017, S. 108

***Downloads und GitHub-Issues:***
***ct.de/yd7p***