Docker et Wasm

Nous avions vu précédemment qu’aujourd’hui, l’utilisation du WebAssembly ne se limite plus qu’aux navigateurs et profite d’une utilisation également côté serveur, où dans le cloud…

Docker VS WebAssembly

Les débats autour de Docker et WebAssembly sont devenus de plus en plus intenses ces derniers temps. Certains pensent que le WebAssembly va remplacer Docker, ce qui n’est pas le cas.

Le WebAssembly et Docker sont des outils complémentaires, et non des outils concurrents. En effet, le WebAssembly ne remplace pas Docker, mais peut en fait être intégré à Docker pour rendre ses applications plus rapides et plus sûres.

Le WebAssembly peut offrir une meilleure sécurité et une vitesse d'exécution plus rapide et une image Docker plus légère. Cependant, le WebAssembly est encore une technologie relativement récente, et il y a encore des contraintes de sécurité et des difficultés de débogage à prendre en compte.

À l'avenir, nous pourrions voir une plus grande adoption des applications WebAssembly dans Docker, ainsi que des mises à jour permettant plus de sécurité et de facilité de débogage. Il est possible que nous commencions à voir des applications Docker-Compose multi-services basés sur le WebAssembly, ce qui pourrait offrir une solution à certains des problèmes de scalabilité et de sécurité rencontrés aujourd'hui.

Docker With WebAssembly

Aujourd'hui, nous ne devons plus penser que le WebAssembly peut remplacer Docker, mais au contraire, s'intégrer avec ce dernier.

Docker & Linux

Aujourd'hui, toutes les images Docker partent d'une base Linux. Cette dernière peut être plus ou moins légère, mais nous avons toujours cette couche supplémentaire d'un système d'exploitation, aussi léger soit-il, qui ralentit (un peu) le démarrage d'un conteneur et sa vitesse d'exécution.

Une nouvelle très bienvenue

Fin octobre 2022, une annonce officielle tombe. Celle de la prise en charge native du WASM dans Docker. Cette fonctionnalité n’est qu’au stade de BETA, mais cette annonce va amener des changements majeurs pour le futur de Docker.

Cette prise en charge native va permettre une vitesse d'exécution beaucoup plus rapide et une image Docker beaucoup plus légère. De plus, le WASM permet de s'affranchir de l'architecture serveur et offre une meilleure sécurité.

Docker WASM

Intérêt

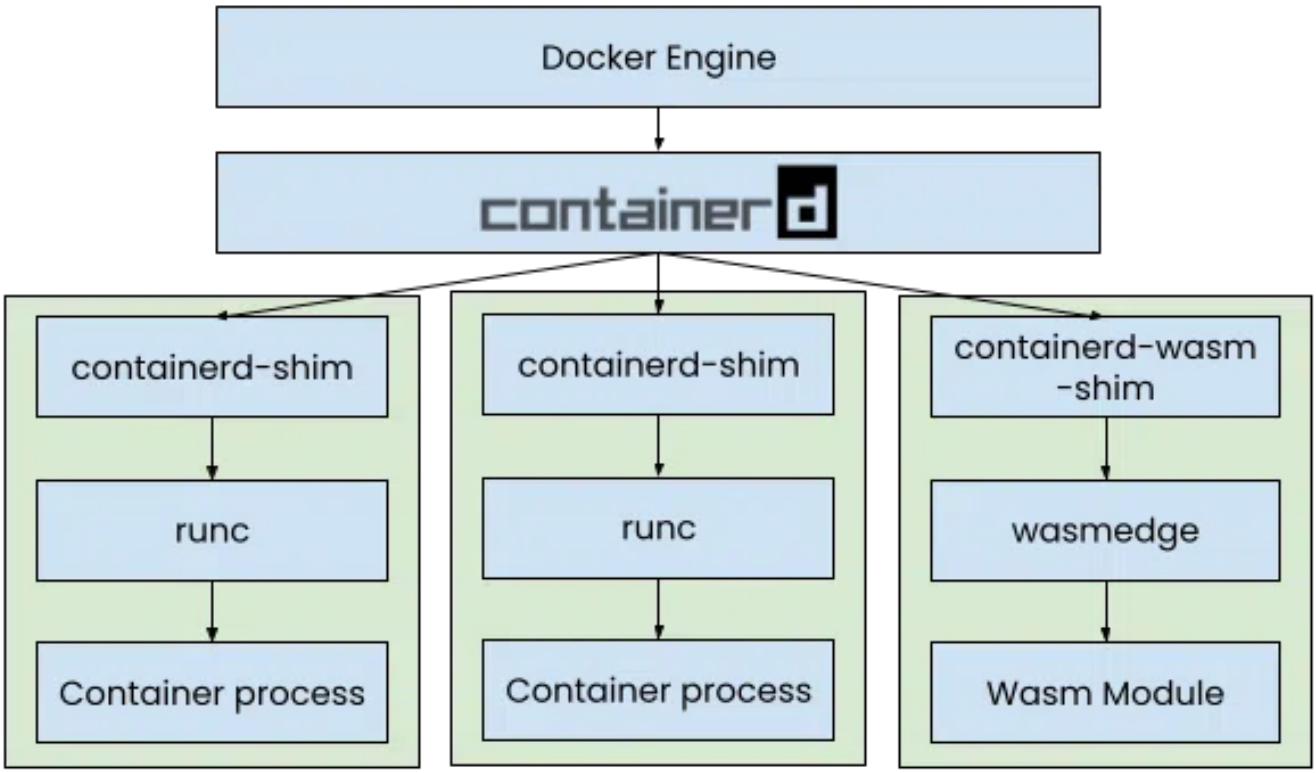
Qu’est-ce que cela va apporter à Docker ?

- Plus de sécurité
- Une vitesse de démarrage / arrêt beaucoup plus grande
- Des images beaucoup plus légères
- Libéré de l’architecture du serveur

Très utile donc dans un environnement Cloud qui nécessite du scaling très rapide, de la légèreté et vitesse de démarrage (coucou les lambda)

Cependant, l'utilisation de WebAssembly dans un environnement Docker est encore très limitée, car le WebAssembly est encore une technologie relativement récente. Il n'y a pas encore beaucoup de bibliothèques disponibles pour le WebAssembly, et les contraintes de sécurité sont toujours une préoccupation. De plus, il est encore très difficile de déboguer du code WebAssembly, ce qui est une tâche courante dans le développement logiciel.

Runner containerd



Le WebAssembly fonctionne dans Docker grâce au moteur d'exécution Runner containerd. Ce moteur est responsable de l'exécution des conteneurs et des microservices gérés par Docker. Il prend en charge plusieurs formats d'images, dont l'image WebAssembly, qui est compilée à partir du code source.

Une fois compilée, l'image WebAssembly est envoyée au moteur d'exécution Runner containerd, qui la charge et l'exécute.

Le runner WebAssembly actuel de Docker supporte wasm-edge, et pourrait supporter d'autres formats à l'avenir.

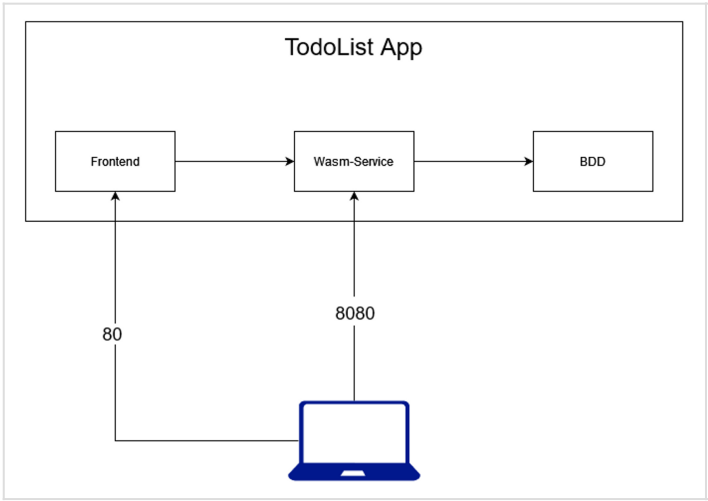
TD Énoncé

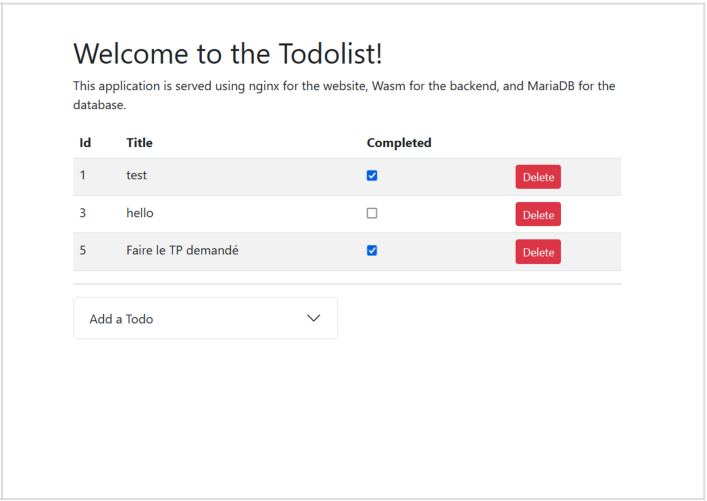
Ressources

“https://docs.docker.com/desktop/wasm/”

“https://github.com/second-state/wasmedge-containers-examples/blob/main/http_server_wasi_app.md”

“https://nigelpoulton.com/getting-started-with-docker-and-wasm/”





TD - Correction

TD - Performances & Comparaison

Conclusion

Le WebAssembly offre un grand nombre d’avantages pour l’utilisation de Docker. Il permet une vitesse d'exécution plus rapide, des images Docker plus légères et de s'affranchir de l'architecture serveur. Néanmoins, le WebAssembly est encore une technologie relativement récente, et il y a encore des contraintes de sécurité et des difficultés de débogage à prendre en compte.

À l'avenir, nous pourrions voir une plus grande adoption des applications WebAssembly dans Docker, ainsi que des mises à jour permettant plus de sécurité et de facilité de débogage. Il est possible que nous commencions à voir des applications Docker-Compose multi-services basés sur le WebAssembly, ce qui pourrait offrir une solution à certains des problèmes de scalabilité et de sécurité rencontrés aujourd'hui.

Ressources

todolist-rust-mysql-linux.zip

todolist-rust-mysql-wasm.zip

todolist-rust-mysql performances.zip

https://thenewstack.io/when-webassembly-replaces-docker/

https://www.programmez.com/actualites/webassembly-integre-dans-docker-preversion-34570

https://www.youtube.com/watch?v=9JVV2qrp080

https://techcrunch.com/2022/10/24/docker-launches-a-first-preview-of-its-webassembly-support/?guce_referrer=YW5kcm9pZC1hcHA6Ly9jb20uZ29vZ2xlLmFuZHJvaWQuZ29vZ2xlcXVpY2tzZWFyY2hib3gv&guce_referrer_sig=AQAAACCiX71P1TpECiG4S7K4MZJhVEzhfEzieZqCXtiIepBnMPEPumRydxLFPmA-N0bXv7iersP3wh28LIU3VB1Qn9oXd8d5B6FH1GJTrXUhTjIRZDl0tW3a80BvHd4TcFGT0DFkDAmt5Iq4dxRJYknQe9lYCwZDHIbaxRyj5P4nR7G3&guccounter=2

https://docs.docker.com/desktop/wasm/

https://github.com/second-state/wasmedge-containers-examples/blob/main/http_server_wasi_app.md

https://nigelpoulton.com/getting-started-with-docker-and-wasm/

https://docs.docker.com/desktop/wasm/

Revision #1
Created 2023-05-05 14:04:21 UTC by Noé Larrieu-Lacoste
Updated 2023-05-05 14:14:17 UTC by Noé Larrieu-Lacoste