Unikernel

Les **unikernels** sont des images systèmes spécialisées, où tous les processus partagent le même espace mémoire, créées en utilisant des systèmes d'exploitation bibliothèques. Le développeur sélectionne, à partir d'un ensemble modulaire, un ensemble minimum de bibliothèques qui correspondent aux services du système d'exploitation nécessaires à l'exécution de son application. Ces bibliothèques sont alors compilées avec l'application et des configurations pour créer des images fixes, à but unique, qui fonctionnent directement sur un hyperviseur ou sur du matériel sans intervention d'un système d'exploitation tel que Linux ou Windows.

Sommaire

Système d'exploitation bibliothèque

Dans un système d'exploitation bibliothèque, les limites de protection sont repoussées vers les bas niveaux, ce qui permet :

1. un ensemble de bibliothèques qui implémentent les mécanismes nécessaires pour faire fonctionner le matériel ou pour gérer les protocoles réseau;
2. un ensemble de règles permettant d'appliquer une politique d'accès et d'isolation au niveau de l'application.

Les premiers systèmes de ce type étaient Exokernel et « Nemesis »[note 1] vers la fin des années 1990.

L'architecture des systèmes d'exploitation bibliothèque présente plusieurs avantages et inconvénients en comparaison de la conception de systèmes plus classiques. Un des avantages est que, puisqu'il n'y a qu'un seul espace d'adressage, les transitions de privilège entre l'espace utilisateur et l'espace noyau, régulières sur les systèmes classiques, ne sont pas nécessaires. Ainsi, un système d'exploitation bibliothèque peut offrir des performances améliorées en offrant un accès direct au matériel sans changement de contexte. Un inconvénient est que, puisqu'il n'y a pas de séparation, il est difficile de faire fonctionner plusieurs applications de concert dans un système d'exploitation bibliothèque avec une isolation des ressources. De plus, des pilotes de matériel sont requis pour la machine sur laquelle le système d'exploitation fonctionne. Les changements fréquents du matériel font de ce besoin de réécrire des pilotes régulièrement un problème.

La virtualisation permet d'éviter ces points négatifs sur du matériel courant. Les hyperviseurs modernes permettent de simplement gérer des machines virtuelles avec le temps processeur qu'elles consomment, tout en fournissant une isolation forte. Un système d'exploitation bibliothèque fonctionnant dans une machine virtuelle n'a besoin d'implémenter que les pilotes pour le matériel virtuel stable que fournit cet hyperviseur, et peut laisser la tâche complexe de gérer le véritable matériel à l'hyperviseur. Cependant, des bibliothèques permettant d'utiliser des protocoles réseau sont toujours nécessaires pour remplacer les services fournis par un système d'exploitation plus classique. La création de ces bibliothèques de protocoles est la plus grande partie du travail d'implémentation d'un système d'exploitation bibliothèque moderne[1].

Avantages et inconvénients

Les unikernels ont un certain nombre d'avantages, mais aussi d'inconvénients, en comparaison des systèmes d'exploitation classiques.

- Une sécurité améliorée — En réduisant la quantité de code déployé, les unikernels réduisent la surface d'attaque et présentent donc une sécurité améliorée[2,3].
- Taille réduite — Il a été démontré que les unikernels ne faisaient qu'environ 4 % de la taille de bases de code équivalentes basées sur un système d'exploitation classique[4].
- Optimisation du système — En raison de la façon dont ils sont construits, il est possible d'optimiser le système entier, des pilotes à l'application, ce qui permet d'améliorer la spécialisation[5,6].
- Temps de démarrage faibles — Il a été montré régulièrement que les unikernels étaient capables de démarrer extrêmement rapidement, suffisamment pour répondre à une requête avant son expiration[7,8,9].

Ces avantages poussent les unikernels vers l'utilisation en tant que systèmes de type orienté-services ou microservices

Cependant, le haut niveau de spécialisation signifie que les unikernels ne sont pas appropriés à l'informatique généraliste, avec plusieurs utilisateurs, pour laquelle les systèmes d'exploitation classiques sont utilisés. Ajouter des fonctionnalités ou altérer un unikernel compilé est en général impossible, et l'approche acceptée est de recompiler et de déployer un nouvel unikernel avec les changements désirés.

Implémentations modernes

Il existe un grand nombre de nouvelles approches à la construction d'unikernels, qui sont à des degrés variables de maturité.

ClickOS

ClickOS[6,10] est une plateforme haute performance virtualisée pour les appliances, basée sur un système de virtualisation open-source. Des analyses de performance montrent que les machines virtuelles ClickOS sont petites (5 Mo), démarrent vite (jusqu'à 20 millisecondes), ajoutent peu de délai à l'application cible (45 microsecondes), et plus de 100 peuvent être en fonctionnement simultané, en saturant une interface 10Gb, sur un serveur peu cher standard.

Clive

Clive[11] est un système d'exploitation conçu pour fonctionner dans des environnements de calcul distribué et d'informatique en nuage, écrit dans le langage de programmation Go.

Drawbridge

Drawbridge est un prototype de recherche sur une nouvelle forme de virtualisation pour le sandboxing d'applications. Drawbridge combine deux technologies centrales : un picoprocessus, qui est un conteneur d'isolation basé au niveau processus présentant une surface d'API kernel minimale, et un système d'exploitation bibliothèque, qui est une version de Windows modifiée pour fonctionner efficacement dans un picoprocessus. [12]

Graphene

Graphene[13,14] est un système d'exploitation bibliothèque compatible avec Linux qui concentre ses efforts sur la sécurisation d'applications anciennes multi-processus, de type serveur ou shell. Graphene sépare une application multi-processus sur plusieurs picoprocessus, avec les abstractions inter-processus (signaux, files d'attente, sémaphores, etc.) coordonnées sur des flux simples. Pour les applications présentant des principes de sécurité multiples, Graphene peut sandboxer dynamiquement un picoprocessus exposé.

HaLVM

HaLVM[15] (Haskell Lightweight Virtual Machine) est un port de Glasgow Haskell Compiler, qui permet aux développeurs d'écrire des machines virtuelles haut niveau légères qui fonctionnent sur l'hyperviseur Xen.

IncludeOS

IncludeOS[16] est un système d'exploitation bibliothèque minimaliste, orienté services, open-source, et intégrable, visant les services dans le nuage. C'est actuellement un projet de recherche permettant de faire fonctionner du code C++ sur du matériel virtuel.

LING

LING[17] est un unikernel basé sur Erlang/OTP qui est capable d'interpréter les fichiers .beam. Les développeurs peuvent créer du code Erlang et le déployer en tant qu'unikernels LING. LING retire la majorité des fichiers vecteurs, n'utilise que trois librairies externes et n'utilise pas OpenSSL.

MirageOS

MirageOS[18,19] est un système d'exploitation bibliothèque qui permet de créer des unikernels pour des applications réseau à haute performance sécurisées sur une grande variété de plateformes mobiles ou dans le nuage. Il existe actuellement plus de cent bibliothèques MirageOS[20] et un nombre croissant de bibliothèques compatibles dans l'écosystème OCaml.

OSv

OSv (en) est un système d'exploitation conçu par Cloudius Systems spécifiquement pour les machines virtuelles dans le nuage[21]. Capable de démarrer en moins d'une seconde, OSv est développé dans le seul but d'exécuter une application sur un hyperviseur quelconque, ce qui offre de meilleures performances et une gestion simplifiée. OSv peut lancer des exécutables Linux non-modifiés (avec quelques limitations) et supporte les langages C, C++, Ruby, Node.js, et les langages basés sur la JVM.

Rumprun

Runprun[22] est une pile logicielle qui permet de lancer des logiciels POSIX non-modifiés dans un unikernel. Rumprun supporte de multiples plateformes, dont le fonctionnement direct sur du matériel et des hyperviseurs tels que Xen ou KVM. Il est basé sur des *rump kernel* (en) qui offrent des pilotes logiciel portables, libres, séparés en composants, et de qualité, pour notamment des systèmes de fichiers, des gestionnaires d'appels systèmes POSIX, des pilotes de matériel PCI, une pile de protocole SCSI, virtio et une pile TCP/IP[23].

Runtime.js

Runtime.js[24] est un système d'exploitation bibliothèque open-source pour le nuage qui fonctionne sur la machine virtuelle JavaScript, qui peut être incluse avec une application et déployée comme une image virtuelle légère et immuable. Runtime.js est basé sur le moteur javascript V8 et supporte actuellement l'hyperviseur QEMU/KVM.

Notes et références

Notes

1. Page de présentation du projet « Nemesis » - Nemesis (http://www.cl.cam.ac.uk/research/srg/netos/projects/archive/nemesis)

Références

1. « Unikernels: Rise of the Virtual Library Operating System » (http://queue.acm.org/detail.cfm?id=2566628) (consulté le 31 août 2015)
2. « Why Unikernels Can Improve Internet Security » (http://www.linux.com/news/enterprise/cloud-computing/820669-why-unikernels-improve-internet-security) (consulté le 31 août 2015)
3. Anil Madhavapeddy, Richard Mortier, Rotsos Charalampos, David Scott, Balraj Singh, Thomas Gazagnaire, Steven Smith, Steven Hand et Jon Crowcroft, « Unikernels: Library Operating Systems for the Cloud », *SIGPLAN Notices (ASPLOS 13)*, vol. 48, nº 4, mars 2013 (DOI 10.1145/2499368.2451167 (https://dx.doi.org/10.1145/2499368.2451167), lire en ligne (http://anil.recoil.org/papers/2013-asplos-mirage.pdf))
4. David Kaloper-Meršinjak, Hannes Mehnert, Anil Madhavapeddy et Peter Sewell, « Not-Quite-So-Broken TLS: Lessons in Re-Engineering a Security Protocol Specification and Implementation », *Proceedings of the 24th USENIX Security Symposium (USENIX Security 15)*, 2015 (lire en ligne (https://nqsb.io/nqsbtls-usenix-security15.pdf))
5. Anil Madhavapeddy, Richard Mortier, Ripduman Sohan, Thomas Gazagnaire, Steven Hand, Tim Deegan, Derek McAuley et Jon Crowcroft, « Turning Down the LAMP: Software Specialisation for the Cloud », *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, 2010 (lire en ligne (http://anil.recoil.org/papers/2010-hotcloud-lamp.pdf))
6. .Joao Martins, Ahmed Mohamed, Costin Raiciu et Felipe Huici, « Enabling Fast, Dynamic Networking Processing with ClickOS », *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, 2013 (DOI 10.1145/2491185.2491195 (https://dx.doi.org/10.1145/2491185.2491195), lire en ligne (http://nets.cs.pub.ro/~costin/files/hotsdn13.pdf))
7. « Just-in-Time Summoning of Unikernels (v0.2) » (http://www.skjegstad.com/blog/2015/08/17/jitsu-v02/), sur *Magnus Skjegstad* (consulté le 30 août 2015)
8. « Zerg » (http://zerg.erlangonxen.org/), sur *Zerg — an instance per request demo* (consulté le 30 août 2015)
9. .Anil Madhavapeddy, Thomas Leonard, Magnus Skjegstad, Thomas Gazagnaire, David Sheets, David Scott, Richard Mortier, Amir Chaudhry, Balraj Singh, Jon Ludlam, Jon Crowcroft et Ian Leslie, « Jitsu: Just-In-Time Summoning of Unikernels », *the 12th USENIX Conference on Networked Systems Design and Implementation (NSDI)*, 2015 (ISBN 978-1-931971-218, lire en ligne (http://anil.recoil.org/papers/2015-nsdi-jitsu.pdf))
10. « ClickOS and the Art of Network Function Virtualization » (https://www.usenix.org/system/files/conference/nsdi14/nsdi14-paper-martins.pdf) (consulté le 31 août 2015)
11. « The Clive Operating System » (http://lsub.org/export/clivesys.pdf) (consulté le 31 août 2015)
12. « Drawbridge » (http://research.microsoft.com/en-us/projects/drawbridge/), sur *Microsoft Research* (consulté le 30 août 2015)
13. Tsai Chia-Che, Kumar-Saurabh Arora, Nehal Bandi, Bhushan Jain, William Jannen, Jitin John, Harry Kalodner, Vrushali Kulkarni, Daniela Oliviera et Donald E. Porter, « Cooperation and Security Isolation of Library OSes for Multi-process Applications », *Proceedings of the Ninth European Conference on Computer Systems (EuroSys)*, 2014 (DOI 10.1145/2592798.2592812 (https://dx.doi.org/10.1145/2592798.2592), lire en ligne (http://www3.cs.stonybrook.edu/~porter/pubs/tsai14graphene.pdf))
14. « Graphene library OS » (https://github.com/oscarlab/graphene), Stony Brook University (consulté le 31 janvier 2016)
15. HaLVM (https://galois.com/project/halvm/)
16. IncludeOS (http://www.includeos.org)
17. « Erlang on Xen: at the heart of super-elastic clouds » (http://erlangonxen.org/) (consulté le 31 août 2015)
18. MirageOS (https://mirage.io)
19. « MirageOS: A programming framework for building type-safe, modular systems » (https://mirage.io/) (consulté le 31 août 2015)
20. « MirageOS TROVE » (https://github.com/mirage/mirage-www/blob/master/TROVE) (consulté le 31 août 2015)
21. Avi Kivity, Glauber Costa, Pekka Enberg, Nadav Har'El, Don Marti et Vlad Zolotarov, « OSv: Optimizing the Operating System for Virtual Machines », *2014 USENIX Annual Technical Conference*, juin 2014 (lire en ligne (https://www.usenix.org/system/files/conference/atc14/atc14-paper-kivity.pdf))
22. (en) Rumprun (http://repo.rumpkernel.org/rumprun).
23. (en) « Rump Kernels » (http://rumpkernel.org/), sur *rumpkernel.org* (consulté le 31 août 2015).
24. Runtime.js (http://runtimejs.org/)

Articles connexes

- Exo-noyau
- Microkernel

Politique de confidentialité
À propos de Wikipédia
Avertissements
Contact
Développeurs
Statistiques
Déclaration sur les témoins (cookies)