

Journal : Systemd, Docker, NetworkD, EtcD, FleetCTL | CoreOS : Le Cloud n'est pas un Fog .

Posté par [bubar](#) le 26 février 2014 à 20:11. Licence CC By-SA.
Étiquettes : [cloud](#), [docker](#), [coreos](#)



C'est bon, avec un titre comme ça, j'ai votre attention ? :p Prenons les noms les plus trolllogènes du moment, mettons les ensemble dans un shaker, rajoutons quelques noms moins connus, secouons bien fort, et nous obtenons un délicieux cocktail : CoreOS. Il était temps de parler en quelques mots de CoreOS, en prenant soin cependant de laisser de côté un ton académique pour laisser de la place à un peu d'humour et de bullshit. Restera au lecteur à **aller voir -et tester- par lui même** de quoi il en retourne. Parceque l'avenir de linux est là. (hoo, un coincoin)

CoreOS, kékako ?

Linux for Massive Server Deployments

CoreOS enables warehouse-scale computing on top of a minimal, modern operating system

Il s'agit d'un projet de système minimaliste, sur lequel on greffe des containers applicatifs indépendants et sécurisés.

Prenez donc tout ces mots-clefs, ajoutez y "Vagrant, Amazon, VMWare, OpenStack, Facebook, Google, ChromiumOS & Gentoo" et vous obtenez une grille gagnante au bingo-réunion de la semaine.

Les bonnes idées venues d'ailleurs :

Partitions système à états :

Pourquoi ChromiumOS ? Parceque CoreOS en reprend une fonctionnalité essentielle : celle de la double partition / système : architecture dans laquelle une partition "froide" reçoit les mises à jour, et lors du reboot elle devient "partition chaude". Si cela se passe mal, un reboot automatique est opéré sur l'ancienne partition n'ayant pas reçue de mises à jour. Ceci de manière transparente. On parle de "[partition statefull](#)"

Mise à jour système en unique unité :

Plus de gestionnaires de paquets, on met à jour le système d'un seul trait. Cela réduit l'activité disque, permettant un contrôle plus fin au sein d'un cgroup spécifique afin que cette activité disque n'impacte nullement le service rendu, pendant l'écriture de la mise à jour du système. Bien sûr on perd là toute l'atomisation possible que nous permettent nos fantastiques gestionnaires de paquets, quels qu'ils soient : pour une mise à jour de sécurité sur une bibliothèque système, on se voit obligé d'écrire la totalité du RootFS. Mais ça se pèse, surtout pour un système ultra-minimal. On parle de "[Omaha](#)"

Contraindre toutes les applications à être autonomes :

Toutes les applications installées, bureau y compris, seront contraintes dans un container dédié à chacune d'elle. Elles ne verront que leurs propres arborescence, et la gestion sécuritaire en sera grandement facilité. Nous aurons donc la facilité de l'obsolète dossier "program files" de Windows, tout en ayant le meilleur des possibilités de Linux. L'application codée de la pire manière qui soit, ou pire encore, une belle application mais empaquetée par des échappés d'un asile, ne pourront plus pourrir votre système. Le second gain est la facilité de déplacement des applications d'un système à un autre. Le "_workflow" dev->qual->prod ne se fera plus que sur les contraintes de l'application elle même. On parle maintenant de "[Docker](#)" et encore de "[Omaha](#)"

Configurations distribuées et Gestion Centralisée :

Exit la complexité de maintenance des bons vieux outils que nous connaissons tous, que cela soit pour un cluster de calcul ou un cluster de services, même la plus belle possible genre à ré-installation automatique après surveillance défaillance d'un noeud, sa sortie automatique du cluster. Et bonjour le service de gestion centralisée avec une api JSON... On parle là "[d'etcD](#)". Mais aussi de "[FleetCTL](#)"

Simplicité, rapidité

Un des objectifs de CoreOS est un boot en moins de deux secondes. Lorsqu'on sait utiliser kexec, et que l'on sait les 4 minutes chrono en main pour l'initialisation des "bios" modernes de certains serveurs, on rêve d'avoir un reboot qui devienne transparent pour les usagers... aussi transparent qu'un mini lag d'un chargement de page web... Et bien CoreOS propose mieux puisqu'il est simplissime de déménager à chaud une instance d'un container d'un système à un autre, de rebooter une machine sans passer par les phases "bios", et d'être assuré d'un boot sans encombre avec la sécurité de la double partition système. Une intervention en cas de problème se fera sans arrêt de service. Ceci n'est pas option payante et chère (suivez mon regard) qui en plus marche mal. Non : ça fonctionne, c'est libre et c'est inclu. Ces techniques ne sont pas nouvelles en soi, elles sont déjà à l'usage chez les grands noms tel que VmWare, Amazon et Google. Mais elles sont ici disponibles librement, comme dans « Logiciels Libres » ;-)

Conclusion

J'ai écrit ce journal en mode *bullshit* (sans notion de qualité pour présenter ce projet), mais ne loupez pas CoreOS. Allez y, lisez leurs docs, testez, amusez vous. Le seul point à vraiment relevé : il y a des mises à jour rapide si l'on souscrit un abonnement. Car CoreOS est aussi un projet commercial, pas seulement un magnifique bac-à-sable pour voir les évolutions en avance de phase et profiter du meilleur de notre noyau favori.

Pour tester, on a le choix :

- Amazon EC2
- Google Compute Engine
- Rackspace Cloud
- Brightbox Cloud
- Eucalyptus
- Libvirt
- OpenStack
- QEMU
- Vagrant
- VMware

Et bien sûr, une installation en dur sur son laptop ;-) avec un choix en "readonly + ram" ou en "double partition".

CoreOS

(une dizaine de pages qui, une fois lues, mettrons l'eau à la bouche de tout les geeks et nerds du coin, coin)

(6 commentaires).

[Markdown](#) [EPUB](#)

Je suis déçu...

Posté par [Atem18](#) ([site web personnel](#)) le 26 février 2014 à 22:12. Évalué à 10.

Moi qui pensait que tu allait troller sur systemd, me voilà fort contrarié.

[^] # Re: Je suis déçu...

Posté par [BeberKing](#) ([site web personnel](#)) le 27 février 2014 à 09:11. Évalué à 5.

Moi qui pensait que tu allait troller sur systemd, me voilà fort contrarié.



Moi qui pensais que tu allais troller...

Docker IO et Vagrant: essaie encore

Posté par [lfe](#) le 27 février 2014 à 05:54. Évalué à 7. Dernière modification le 27 février 2014 à 20:20.

Je suis un fan de la première heure de LXC. J'adore tout, le concept, la performance, le fait qu'on peut enfin faire du `chroot` dopé aux stéroïdes en utilisant SELinux (alors qu'on ne pouvait pas avec OpenVZ)

LXC est enfin sorti en version 1.0, on peut lancer un container en tant qu'utilisateur. Ça c'est vraiment cool. Ça me rassure d'un point de vue sécurité.

Idem pour KVM, j'adore l'intégration avec Fedora, l'interface `virt-manager` de Red Hat.

Par contre, je dois avouer que je suis vraiment déçu par Docker IO et Vagrant.

Docker IO parce que c'est juste hype, ça n'apporte rien de nouveau. J'utilise LXC pour avoir un Gentoo qui fait tourner Nginx (comme ça je peux personnaliser les options de compilation), une Debian pour mes emails et une Fedora pour mes applications erlang et python.

Ce que Docker IO offre, c'est juste des containers, pré-compilés, prêts à être lancés tel quels. Le système entier est partagé en lecture seule entre des container (par exemple un container "wordpress" a un serveur LAMP), le container lance les services, et une partition de donnée est unique pour chaque instance (dans notre exemple de wordpress, on peut écrire dans `/var/lib/mysql` et `/var/www/wordpress/uploads/`). C'est sympa dans l'idée, le problème pour utilisateur comme moi c'est que j'ai pas assez de contrôle dessus, pour un utilisateur pas trop admin sys, le problème c'est que les "templates" comme ils les appellent, ne sont pas vraiment maintenus.

Vagrant, l'idée est sympa pour le développement, ce que j'aime c'est l'idée de télécharger un cd d'installation, simuler les appuis clavier dans la VM. Comme ça on fait des installations automatiques. Le problème c'est que ça utilise VirtualBox (on a vu mieux au point de vue performance, et des grosses briques de virtualbox ne sont pas sous licence libre.) Je sais que Red Hat travaille sur un support de KVM dans Vagrant. J'attends avec impatience!

--

Ruby est le résultat d'un gamin qui apprend le Java, puis jette un œil à Perl et se dit « je peux le réparer! »

[^] # Re: Docker IO et Vagrant: essaie encore

Posté par hermenegilde le 04 mars 2014 à 13:51. Évalué à 2.

J'ai pas assez de contrôle dessus



Voilà, un problème de contrôle. Quand tu es tout seul sur ton serveur pour faire 2/3 trucs, c'est sympa. Quand tu dois déployer 500 applis sur 200 serveurs, tu t'amuses pas à faire des trucs à la main. Docker, c'est sympa dans ce cas, même si ça apporte d'autres problèmes.

Je vois bien l'intérêt d'utiliser un docker avec Apache Mesos par exemple.

Quid du support d'ARMv7 & ARMv8 ?

Posté par HLFH le 01 mars 2014 à 03:29. Évalué à 1.

Sais-tu que le Raspberry Pi a ouvert le marché des Single-board computers (SBC) ?

Les SBCs ont un comparatif à jour : http://en.wikipedia.org/wiki/List_of_single-board_computers^w

D'après ce comparatif et l'historique communautaire, la communauté la plus en vue est la communauté linux-sunxi au sein duquel agit la tribu Cubie avec le Cubietruck (supportant KVM), le SBC de troisième génération tournant sur Allwinner A20 (ARMv7). <http://cubieboard.org/>

Ces µPCs sont très utilisés pour le selfhosting et donc l'environnement serveur.

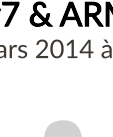
Ton CoreOS semble uniquement supporter l'architecture amd64.

Peut-être que ce créateur d'image universel peut aider pour porter CoreOS vers ARMv7 : <https://github.com/EddyBeaupre/armStrap>

[^] # Re: Quid du support d'ARMv7 & ARMv8 ?

Posté par Misc (site web personnel) le 02 mars 2014 à 10:21. Évalué à 3.

Coreos supporte uniquement amd64 car il faut bien se rendre compte que si ton but est d'avoir un consommateur de courant minimal (cas typique d'avoir une board arm à la place d'un pc complet pour chez toi), tu va pas commencer à faire un cluster de machine.



De plus, docker n'a pas l'air d'avoir eu un port ARM pendant longtemps (<https://github.com/dotcloud/docker/issues/636>), et il faut bien voir que l'avantage de docker, c'est d'avoir aussi une liste d'image dans leur index, et que les images sont mono-arch pour le moment. Et que tout refaire, ça revient à refaire tout le travail d'une distribution, ce que visiblement, les gens veulent pas (sinon, ils feraient une distro, soyons honnêtes)

Suivre le flux des commentaires

Note : les commentaires appartiennent à celles et ceux qui les ont postés. Nous n'en sommes pas responsables.

Revenir en haut de page

Derniers commentaires

- J'ai oublié d'ajouter l...
- Un appel en régie
- Re: Typo ou informa...
- Re: C'est un petit vo...
- Internet de confianc...
- Re: La réalité face à l...
- Re: C'est un petit vo...
- Re: Un fantasma qu...
- Re: Un fantasma qu...
- Layer7
- Re: Cash
- Re: Un classique des...

Étiquettes (tags)

populaires

- intelligence_artificielle
- rétro-informatique
- grands_modèles_de...
- merdification
- cybersécurité
- souveraineté_numer...
- administration_fran...
- capitalisme_de_surv...
- états-unis
- bigtech
- france
- css

Sites amis

- Agenda du Libre
- April
- Éditions D-BookeR
- Éditions Diamond
- Éditions Eyrolles
- Éditions ENI
- En Vente Libre
- Framasoft
- La Quadrature du Net
- Lea-Linux
- Open Source Initiative
- Imprimerie Grafik Plus

À propos de

LinuxFr.org

- Mentions légales
- Faire un don
- L'équipe de LinuxFr...
- Informations sur le s...
- Aide / Foire aux que...
- Suivi des suggestion...
- Wiki du site
- Règles de modération
- Statistiques
- API pour le développ...
- Code source du site
- Plan du site