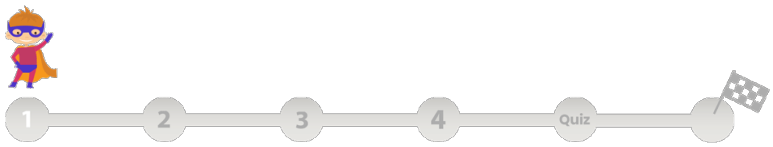
1 - Pet versus Cattle



1. Pet versus Cattle

En matière de gestion de parc informatique, il existe **deux mondes diamétralement opposés**. Si vous voulez mettre en place une équipe DevOps, vous devez impérativement comprendre les différences entre l'approche **« pet » et « cattle ».**

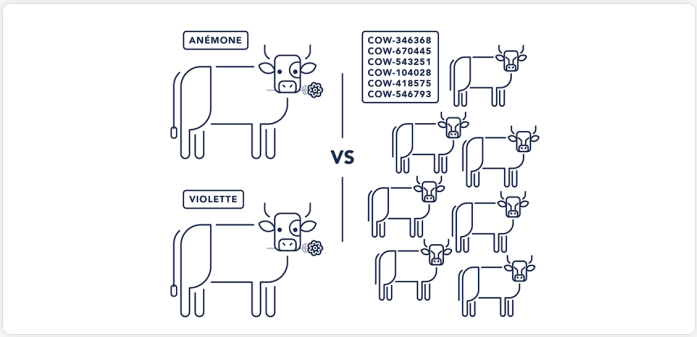


Figure 1 : "pet" versus "cattle"

1.1 Approche « pet », animaux de compagnie

Cette approche historique est adoptée depuis les prémices de l'informatique. Dès lors qu'il a fallu gérer des parcs informatiques, les premiers administrateurs système (sysadmins) ont commencé à **nommer leurs serveurs (physique ou virtuel)** pour les reconnaître plus facilement.

Ces sysadmins s'occupaient régulièrement de **prendre soin des serveurs** afin de toujours avoir **des services fonctionnels.**
Chaque serveur avait une fonctionnalité spécifique : il y avait le serveur dédié aux **bases des données**, celui dédié aux **applications web de production, de développement,** etc. ... Si un serveur tombe en panne, c'est la catastrophe, il faut se réunir entre experts et trouver une solution pour le remettre sur pied le plus vite possible. Les gestions des **mises à jour** doivent être faites **indépendamment sur chaque machine,** le plus souvent manuellement.

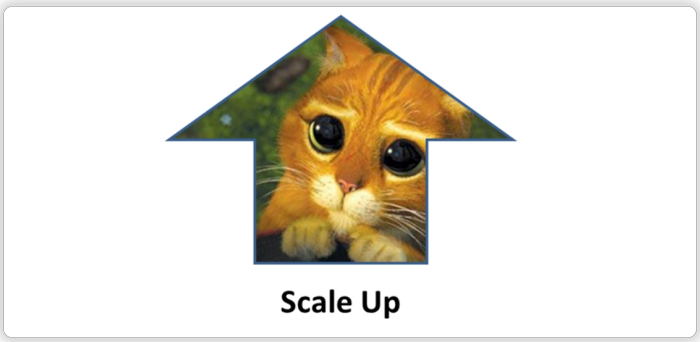


Figure 2 : Approche « pet », facilite la scalabilité verticale

De par son **approche très conservatrice** des serveurs, les **problèmes de montée en charge** de l'approche **"pet"** ne peuvent être résolus qu'avec une **scalabilité verticale**, ce qui n'est **pas la meilleure stratégie** dans la culture **DevOps**.

Cette approche s'applique à **tout type de serveurs** : les mainframes, les serveurs solitaires, les équilibreurs de charge (load-balancer) en haute disponibilité, les pare-feux (actif/actif ou actif/passif), les systèmes de base de données conçus comme maître/esclave (actif/passif), et bien d'autres systèmes.

L'approche « Pet » est toujours utilisée dans de nombreuses sociétés, mais elle n'est **pas vraiment pertinente avec la philosophie du DevOps.** En effet, il parait compliqué de gagner en performance et en rapidité s'il faut passer beaucoup de temps à s'occuper de ses serveurs.

1.2 Approche « cattle », transformation en bétail

Avec l'arrivée de la **virtualisation**, l'approche « pet » perd peu à peu son sens.
Il y a de plus en plus de serveurs à administrer : on commence donc naturellement à **utiliser des scripts** pour déployer des **modifications en masses sur des grappes de VM.** On perd petit à petit toute l'intimité que l'on pouvait avoir avec ses serveurs qui commence à se noyer dans la masse.

Des **outils automatisés** font leurs apparitions afin de **provisionner** et **configurer automatiquement** les instances virtuelles.

Les serveurs sont conçus pour être défaillants, ils ne sont plus irremplaçables. Lors de défaillance, aucune intervention humaine n'est requise, car des mécanismes permettent le **« routage autour des défaillances »** en redémarrant les serveurs défaillants ou en répliquant les données. Les exemples incluent les réseaux de serveurs Web, les bases de données multi-maîtres telles que les clusters Cassandra et à peu près tout ce qui est conçu pour fonctionner en cluster multi-maîtres.

Figure 3 : Approche « cattle », facilite la scalabilité horizontale

Il faut se détacher des machines que l'on veut administrer. **Finis les serveurs nommés "srv_bdd01.my-company.com" ou "srv-lb02.my-company.com".**
Il faut considérer ses machines comme du bétail qui vit et qui meurt. **Si un serveur tombe en panne, on le jette sans souci de savoir ce qu'il a.**

Lorsqu'un **serveur pose problème**, on ne l'emmène **plus chez le vétérinaire**, mais directement à **l'abattoir**. Cette vision est un peu dure, mais elle résume très bien la **différence fondamentale entre les approches « pet » et « cattle »**.

Pour voir la fiche complète et les documents attachés, rendez-vous sur
https://elearning.26academy.com/course/play/5aa2661754470b34d4e474f9