

# Test de performance en intégration continue dans un *cloud* de type PaaS

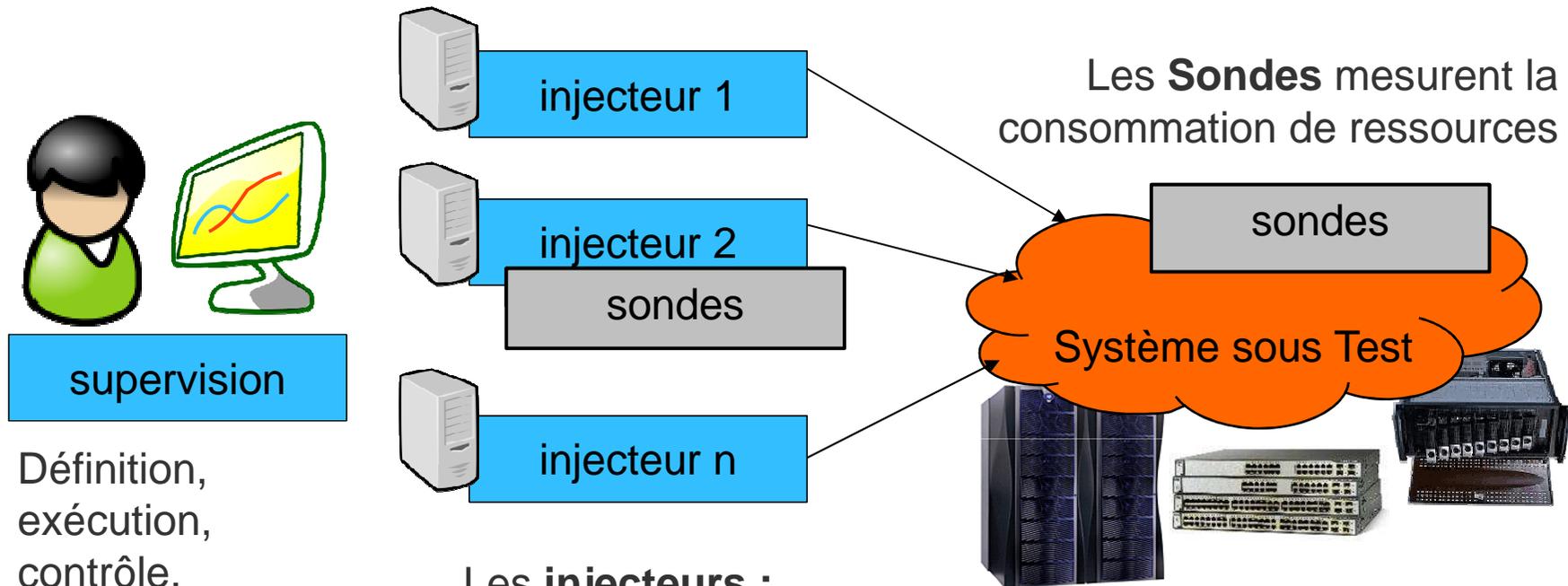
Bruno Dillenseger  
Orange Labs Grenoble

ComPAS, Conférence d'informatique en Parallélisme, Architecture et Système, Grenoble, 15 janvier 2013  
Atelier SLA pour le Cloud Computing

# En bref...

- Utilisation de technologies *cloud* pour le test de performance
- Test de performance en mode « intégration continue » sur PaaS
- Application pour surveiller les performances d'un système en exploitation (qualité de service/expérience)

# Principe du test de performance



Définition,  
exécution,  
contrôle,  
analyse  
des tests

## Les injecteurs :

- envoient des requêtes, attendent les réponses et mesurent les temps de réponse
- selon un certain **scénario** qui définit la charge
- typiquement en émulant la charge d'utilisateurs réels  
→ notion d'**utilisateurs virtuels**

# Limites du test de performance

- Comme pour toute pratique de test, le test de performance est soumis à un double compromis :



- coût des test vs retour sur investissement

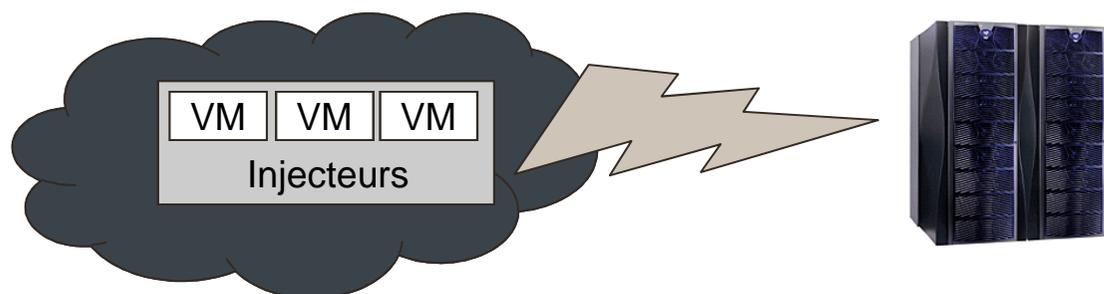


- durée des tests vs délai de mise sur le marché

- Le coût du test de performance provient notamment :
  - achat de l'infrastructure de test (matériel informatique et réseau)
  - licences des logiciels (en particulier des logiciels de test)
  - configuration, exploitation et hébergement de l'infrastructure de test
  - ... ***pour des usages ponctuels dans des campagnes de test limitées dans le temps !***
- La durée des campagnes de test est allongée par le délai de mise en place de l'infrastructure
  - système sous test
  - système d'injection de charge

# Les promesses du *cloud de type IaaS*

- L'utilisation d'un *cloud* de type IaaS permet de disposer d'une infrastructure matérielle sur mesure à la demande
  - caractéristiques adaptées à chaque campagne
  - paiement à l'usage
  - facilités de déploiement permettant d'accélérer la mise en place et la reproductibilité (cf. images de machines virtuelles)
  - en utilisant plusieurs IaaS, on peut générer du trafic provenant de différentes zones géographiques



IaaS pour l'injection de charge

système sous test

# Quelques exemples d'outils de tests sur IaaS

- Des services d'injection de charge sur IaaS sont d'ores-et-déjà disponibles en ligne
  - e.g. BlazeMeter (blazemeter.com), CloudTest (soasta.com), NeoLoad Cloud Testing (neotys.fr)
- Il s'agit de versions « cloud » d'outils de test « classiques »
- La valeur ajoutée est dans l'encapsulation :
  - du déploiement
  - du monitoring
  - du traitement des mesures

# Cas particulier : IaaS basé sur des ressources non dédiées

## ■ Approche :

- Les collègues mettent à disposition une partie de leurs ressources à certaines périodes de la journée/semaine
- Les tests s'exécutent lorsqu'il y a suffisamment de ressources disponibles

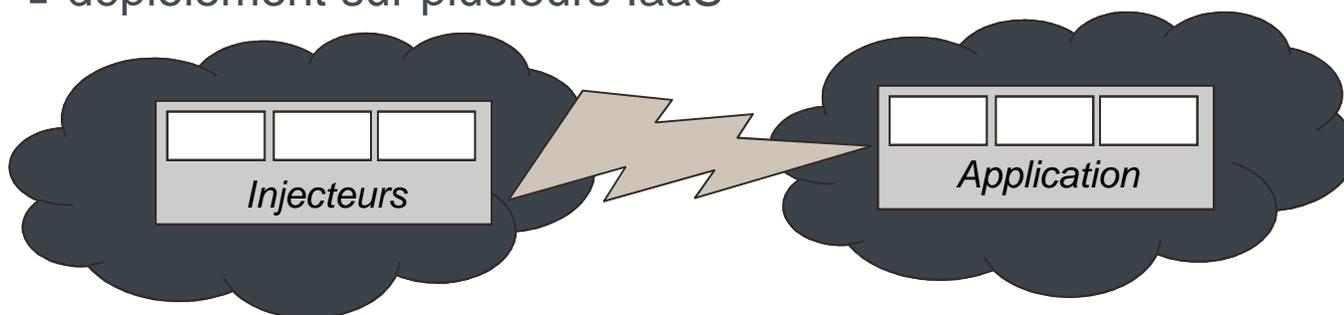
## ■ Bénéfices :

- Pas d'infrastructure dédiée
- Nombreuses sources de trafic, largement réparties



# Les promesses du *cloud* de type PaaS

- Dans le cadre d'un *cloud* de type PaaS, on bénéficie en plus de l'automatisation du déploiement :
  - de l'application développée sur le PaaS
  - du système d'injection de charge
- Quelques problèmes techniques à résoudre :
  - le système d'injection doit être configuré automatiquement pour qu'il localise le point d'entrée de l'application
  - déploiement sur plusieurs IaaS



# Intégration continue et automatisation des tests

- Les outils d'intégration continue permettent d'automatiser l'exécution de tests...
  - déclenchés selon différentes politiques (e.g. périodiquement, manuellement, détection d'un commit SVN...)
  - quelques exemples : Hudson/Jenkins, Atlassian Bamboo
  - classiquement : compilation, analyse de code, tests unitaires
  
- ... et la production de rapports de tests
  - test par test
  - tendances
  - alertes

# Tests de performance en intégration continue

- Automatisation de tests de performance :
  - pour suivre l'évolution des performances des versions successives d'un logiciel
    - e.g. test de non-régression orienté performances
  - surveillance de qualité d'expérience d'un service en ligne
    - disponibilité
    - temps de réponse de bout en bout
    - erreurs
  
- Quelques outils sont disponibles
  - e.g. pour Jenkins : plug-in Jmeter, plug-in CLIF

# Exemple : tests CLIF en intégration continue



**Injecteur - IsacRunner ( Argument=random1.xis Comment= )**

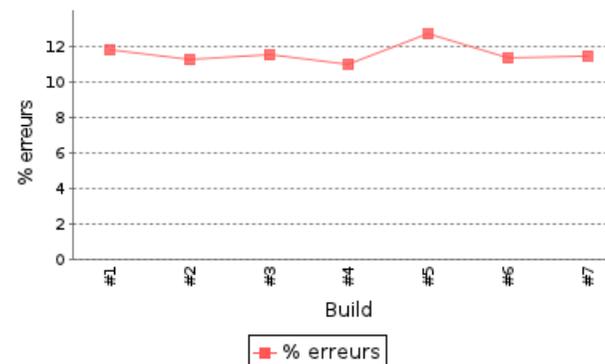
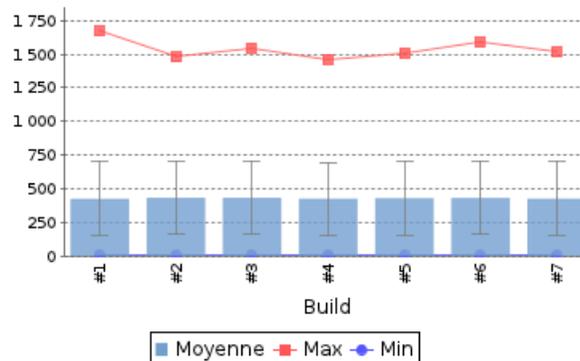
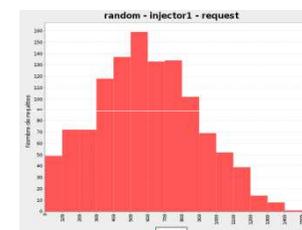
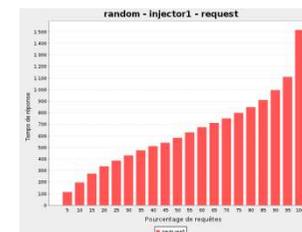
Action	Nb.	Moy.	Méd.	Min	Max	Ecart type	Débit	% Erreurs
request	1 160	596	584	3	1516	295,04	38,82	10,26%

**Injecteur - IsacRunner ( Argument=random2.xis Comment= )**

Action	Nb.	Moy.	Méd.	Min	Max	Ecart type	Débit	% Erreurs
request	1 699	308	299	1	958	181,93	57,97	12,36%

**leurs agrégées**

Action	Nb.	Moy.	Méd.	Min	Max	Ecart type	Débit	% Erreurs
request	2 859	425	385	1	1516	273,98	95,29	11,51%



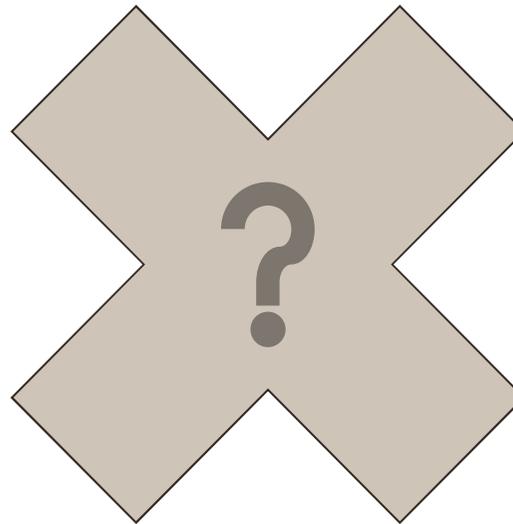
# Vers le test de performance en intégration continue sur PaaS

- L'intégration continue est un service naturel pour un *cloud* de type PaaS
- Le *cloud* de type PaaS permet de déployer le système d'injection au même titre que l'application
  - L'ensemble { application à tester, système d'injection } est vu par le système de déploiement comme une seule application virtualisée
- A suivre...
  - projet collaboratif OpenCloudware : [opencloudware.org](http://opencloudware.org)
  - déploiement multi-iaaS : [sirocco.ow2.org](http://sirocco.ow2.org)
  - test de performance : [clif.ow2.org](http://clif.ow2.org)

# Conclusion et perspectives

- Les promesses du *cloud* sont très attractives pour le test en charge, que ce soit au niveau IaaS, ou au niveau PaaS
  - réduction des coûts d'infrastructure
  - réduction des délais de mise en œuvre
  - automatisation et virtualisation favorisent la reproductibilité des déploiements de logiciel et réduisent le risque d'erreur
- Le *cloud* de type PaaS facilite particulièrement la mise en œuvre des tests de performance en intégration continue
  - application possible à la surveillance de qualité de service
- Attention : le *cloud* n'offre pas toutes les garanties et les informations exigées par une véritable métrologie :
  - quasiment aucune maîtrise de l'infrastructure physique
  - perturbations entre machines virtuelles, perturbations réseau
  - monitoring insuffisant

Vos questions ici



[bruno.dillenseger@orange.com](mailto:bruno.dillenseger@orange.com)