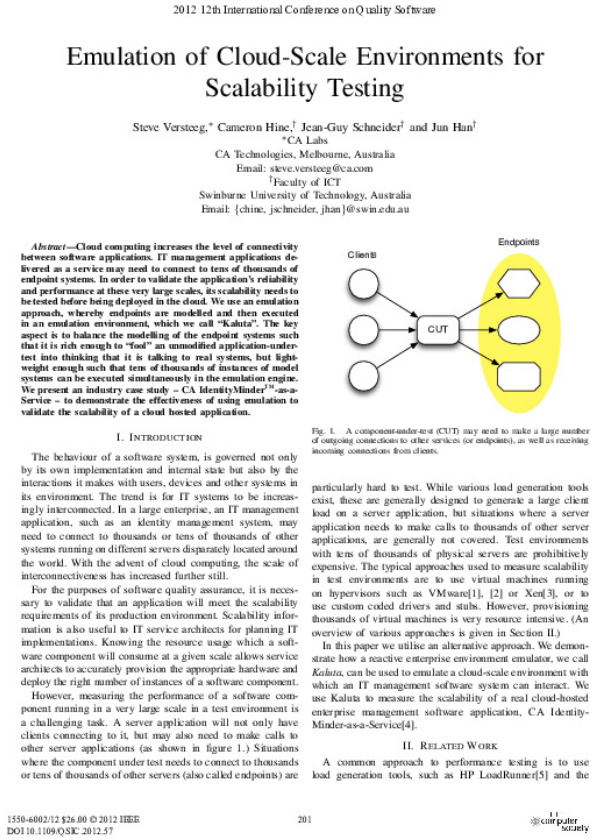
Cloud környezet emulálása skálázhatósági teszteléshez

Lévai Tamás (BUIHKR)

Szoftver verifikáció és validáció (BMEVIMMD052)

Kiindulási alap



2012 12th International Conference on Quality Software

Emulation of Cloud-Scale Environments for Scalability Testing

Steve Versteeg,* Cameron Hine,† Jean-Guy Schneider† and Jun Han†
*CA Labs
CA Technologies, Melbourne, Australia
Email: steve.versteeg@ca.com
†Faculty of ICT
Swinburne University of Technology, Australia
Email: {chine, jschneider, jhan}@swin.edu.au

***Abstract*—Cloud computing increases the level of connectivity between software applications. IT management applications delivered as a service may need to connect to tens of thousands of endpoint systems. In order to validate the application's reliability and performance at these very large scales, its scalability needs to be tested before being deployed in the cloud. We use an emulation approach, whereby endpoints are modelled and then executed in an emulation environment, which we call "Kaluta". The key aspect is to balance the modelling of the endpoint systems such that it is rich enough to "fool" an unmodified application-under-test into thinking that it is talking to real systems, but lightweight enough such that tens of thousands of instances of model systems can be executed simultaneously in the emulation engine. We present an industry case study – CA IdentityMinder™-as-a-Service – to demonstrate the effectiveness of using emulation to validate the scalability of a cloud hosted application.**

I. Introduction

The behaviour of a software system, is governed not only by its own implementation and internal state but also by the interactions it makes with users, devices and other systems in its environment. The trend is for IT systems to be increasingly interconnected. In a large enterprise, an IT management application, such as an identity management system, may need to connect to thousands or tens of thousands of other systems running on different servers disparately located around the world. With the advent of cloud computing, the scale of interconnectiveness has increased further still.

For the purposes of software quality assurance, it is necessary to validate that an application will meet the scalability requirements of its production environment. Scalability information is also useful to IT service architects for planning IT implementations. Knowing the resource usage which a software component will consume at a given scale allows service architects to accurately provision the appropriate hardware and deploy the right number of instances of a software component.

However, measuring the performance of a software component running in a very large scale in a test environment is a challenging task. A server application will not only have clients connecting to it, but may also need to make calls to other server applications (as shown in figure 1.) Situations where the component under test needs to connect to thousands or tens of thousands of other servers (also called endpoints) are particularly hard to test. While various load generation tools exist, these are generally designed to generate a large client load on a server application, but situations where a server application needs to make calls to thousands of other server applications, are generally not covered. Test environments with tens of thousands of physical servers are prohibitively expensive. The typical approaches used to measure scalability in test environments are to use virtual machines running on hypervisors such as VMware[1], [2] or Xen[3], or to use custom coded drivers and stubs. However, provisioning thousands of virtual machines is very resource intensive. (An overview of various approaches is given in Section II.)

Fig. 1. A component-under-test (CUT) may need to make a large number of outgoing connections to other services (or endpoints), as well as receiving incoming connections from clients.

In this paper we utilise an alternative approach. We demonstrate how a reactive enterprise environment emulator, we call *Kaluta*, can be used to emulate a cloud-scale environment with which an IT management software system can interact. We use Kaluta to measure the scalability of a real cloud-hosted enterprise management software application, CA IdentityMinder-as-a-Service[4].

II. Related Work

A common approach to performance testing is to use load generation tools, such as HP LoadRunner[5] and the

1550-6002/12 $26.00 © 2012 IEEE
DOI 10.1109/QSIC.2012.57

IEEE computer society

Versteeg, Steve, et al. "**Emulation of Cloud-Scale Environments for Scalability Testing.**" *2012 12th International Conference on Quality Software. IEEE, 2012.*

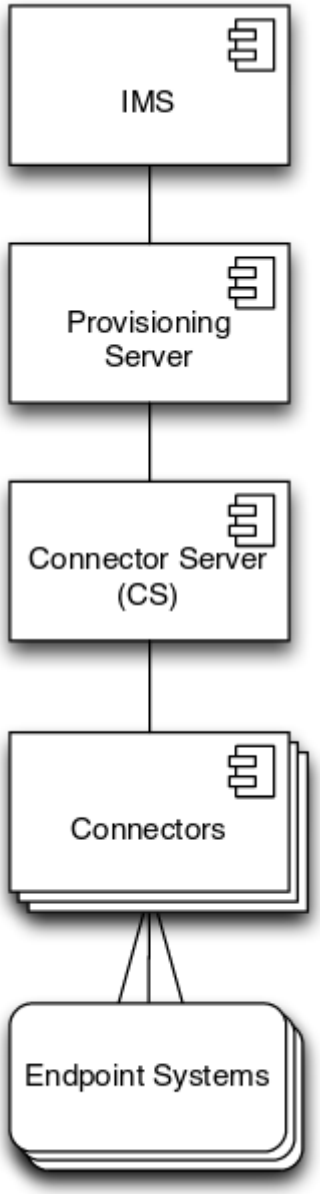
http://ieeexplore.ieee.org/document/6319250/?arnumber=6319250

Felhő alapú számítástechnika

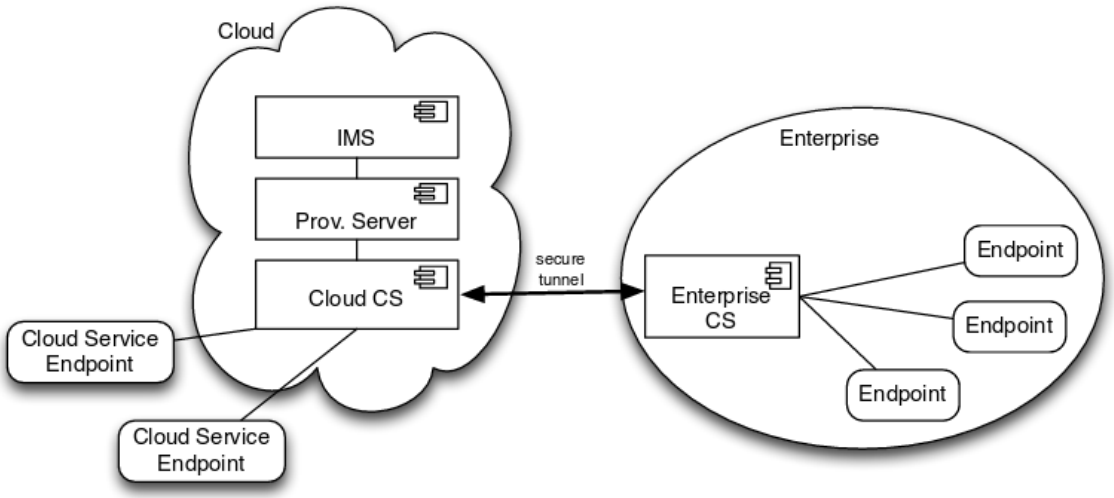
- A szoftver és az azt futtató hardver szétválasztása
  - alapja a **virtualizáció**
- Több szolgáltatásszint (pl.: IaaS, SaaS, ...)
- Sok szolgáltató, egyedi költségmodellekkel
- Tipikusan szolgáltatás-orientált architektúra
  - mikroszervizek

Esettanulmány (1/2)

- CA IdentityMinder
  (ma már CA Identity Managernek hívják)
  - identitás és hozzáférés menedzselő szoftver
  - Java EE
  - komponens alapú felépítés
    - rétegzett architektúra
    - akár több Connector Server is lehet
      - fail-over
      - load-balancing



Esettanulmány (2/2)



- Connector Server komponenst két részre szeretnék bontani
  - Fontos, hogy a **rendszer skálázhatósága** megmaradjon ebben az elrendezésben is

Skálázhatóság

- Egy rendszer **skálázódni képes**, ha tudja áteresztőképességét növelni.
- Két fő dimenziója:
  - „**Scale out**”: újabb számítási csomópontok hozzáadásával nő a teljesítmény
  - „**Scale up**”: nagyobb hardverkapacitás nagyobb teljesítményt ad

Skálázhatósági tesztelés

- Megvalósítása:
  - Rendszer- vagy komponensszinten végzett **terhelésteszt**,
  - ahol fokozatosan növeljük a terhelést,
  - közben monitorozzuk a rendszert
- Értékes kimenetei:
  - a rendszer szűk keresztmetszetei nagy terhelés alatt kiderülnek
    - jellemzően a hálózati, adatbáziskezelési és egyéb szoftveres/hardveres korlátok szoktak ilyenkor kiütközni
  - felhasználószám limit becsülhető

Teszteszközök

- A skálázhatósági tesztekre többféle megközelítés:
  - terheléstesztelő szoftverekkel
    - sok virtuális felhasználóval generálnak terhelést,
    - ahol a virtuális felhasználók egyedi mock objektumok
    - probléma: nem univerzális, mockolás
  - virtuális gépekkel kapcsolódni a rendszerhez és úgy vizsgálni
    - probléma: nem hatékony, nem skálázódik jól
    - alternatíva: konténerek (LXC, Docker, Solaris Zones)
  - Szimulátorok
  - Emulátorok `SoftArch/MTE`

KALUTA

- Emulációs keretrendszer
  - a tesztelendő szoftverkomponens és a végpontok közötti kommunikációra fókuszál
  - csak a végpont és a tesztelendő rendszer közötti kapcsolat szükséges elemeit modellezi
  - akár több több ezer kliens is emulálható egy fizikai gépen
- Fő feladatkörei:
  - protokoll modellezés
  - viselkedés modellezés
  - adat modellezés

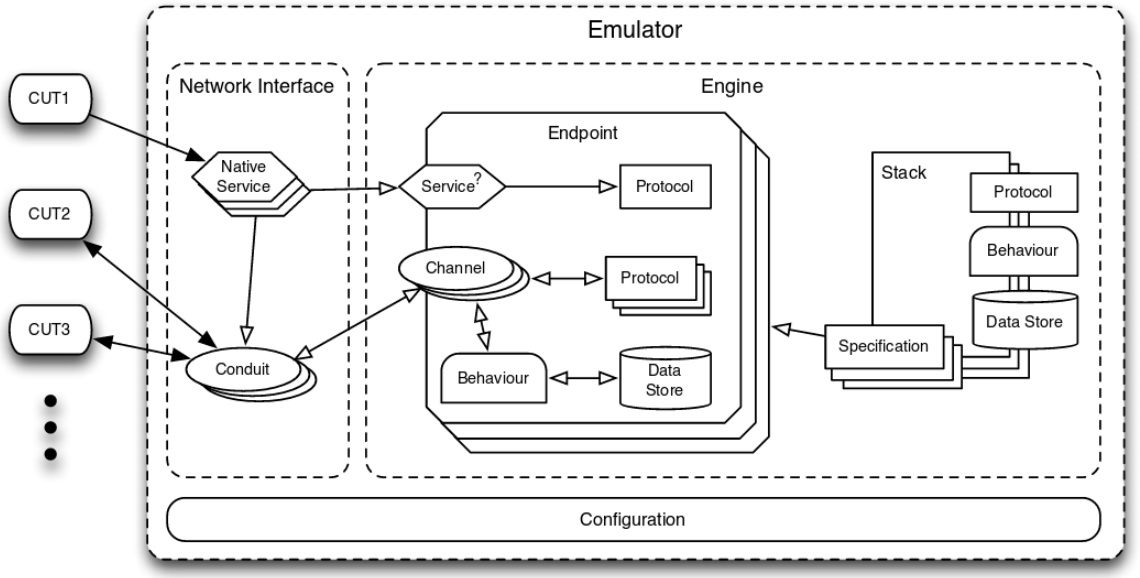
KALUTA – protokoll modell

- Rendszer és a kliensek közötti hálózati kommunikációt írja le
  - üzenet alapú
  - saját szintaxis
  - konkurenciakezelést biztosít

$$
\begin{aligned}
\underline{Base} = \;& ?UnbindRq \triangleright \mathbf{0} \\
& + !DisconNot \triangleright \mathbf{0} \\
& + ?BindRq \triangleright !BindRes.\underline{Base} \\
& + ?SearchRq.\underline{Base}\,[\underline{Search}] \\
& + ?ModRq.\underline{Base}\,[!ModRes.\mathbf{0}] \\
& + ?AddRq.\underline{Base}\,[!AddRes.\mathbf{0}] \\
& + ?DelRq.\underline{Base}\,[!DelRes.\mathbf{0}] \\
& + ?ModDNRq.\underline{Base}\,[!ModDNRes.\mathbf{0}] \\
& + ?CompareRq.\underline{Base}\,[!CompareRes.\mathbf{0}] \\
& + ?AbandonRq.\underline{Base} \\
\textbf{and} \\
\underline{Search} = \;& !SearchResEntry.\underline{Search} \\
& + !SearchResDone.\mathbf{0} \\
\textbf{in}\ & \underline{Base}
\end{aligned}
$$

Fig. 4. An example protocol specification (LDAP). '?' represents a received message, '!' a sent message, '+' is a choice, '.' represents a continuation, '[]' is for dynamic extensions, '**0**' is a termination, and '$\triangleright$ **0**' denotes a termination of a process and all its extensions.

KALUTA – architektúra



- CUT – Component Under Test
- Implementáció
  - Hálózati interfész – Java
  - Engine – Haskell
  - Kommunikáció közöttük: Apache Thrift,
  - akár külön hosztokon is futhatnak a komponensek

KALUTA – működés közben

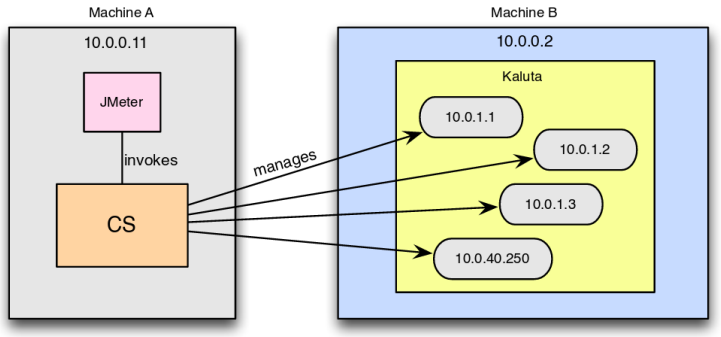
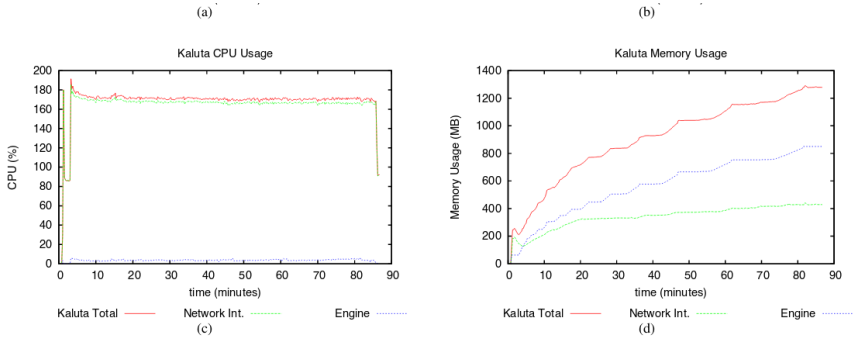


Fig. 6. Experimental setup: Machine A hosts the CS. A JMeter script invokes the CS to manage 10,000 Kaluta-emulated endpoints running on Machine B.

- 2 fizikai hoszt
- 10 000 emulált végpont
- 270 000 LDAP üzenet

- Eredmény:
  - A mérés során előkerült egy memóriaszivárgás is, amely észrevételéhez ekkora terheléstesztet kellett.



Összefoglalás

- Skálázhatóság fontos tulajdonsága a szoftvernek
  - elosztott rendszereknél különösen
- Skálázhatósági teszt hasznos
  - Fejlesztők számára: ki tudja mutatni az a rendszer és annak komponenseinek a szűk keresztmetszeteit
  - Üzemeltetők számára: felhasználószám limit becslések
- A KALUTA keretrendszer hatékony megoldást képes nyújtani a skálázhatósági tesztekhez