STUDIA INFORMATICA 2011
Volume 32 Number 3A (98)

Piotr KANDZIORA, Stanisław KOZIELSKI
Politechnika Śląska, Instytut Informatyki

SILESIAN CLOUD – KONCEPCJA BUDOWY CHMURY W MODELU IAAS

Streszczenie. W artykule przedstawiono ideę *cloud computing* oraz stworzone prototypowe środowisko w modelu IaaS, świadczące usługę gotowych serwerów z zainstalowanymi systemami operacyjnymi. Omówiono architekturę i funkcjonowanie projektowanej chmury oraz wykorzystane technologie. Wskazano przykładowe zastosowania chmury.

Słowa kluczowe: chmura obliczeniowa, wirtualizacja

SILESIAN CLOUD – IDEA OF CLOUD COMPUTING BUILT IN IAAS MODEL

Summary. The paper presents the idea of cloud computing and the prototype system built in IaaS model. The architecture and functions of the cloud and used technologies are presented in the paper. Exemplary applications of the cloud are indicated.

Keywords: cloud computing, virtual machine

1. Wstęp

Artykuł zapoznaje czytelnika z ideą *cloud computing* oraz omawia stworzone, pilotażowe środowisko w modelu IaaS (ang. *Infrastructure as a Service*), świadczące usługę gotowych serwerów (maszyn wirtualnych) z zainstalowanymi systemami operacyjnymi. Projekt przedstawia środowisko funkcjonujące całkowicie automatycznie, posiadające możliwość bezobsługowej instalacji, konfiguracji oraz rejestracji elementów (węzłów) w strukturze chmury. Opiera się na systemie Linux, technologii wirtualizacji KVM (ang. *Kernel-based Virtual Machine*) oraz na otwartym oprogramowaniu OpenNebula, służącym do budowania środowisk typu *cloud*.

2. Kilka słów na temat *cloud computing*

Cloud computing jest pojęciem stosunkowo nowym, podlegającym ciągłej ewolucji. Dotyczy pewnych zasobów takich, jak: serwery, aplikacje, usługi, dostępnych z poziomu sieci. Generalnie użytkownik nie posiada wiedzy, gdzie dokładnie się one znajdują, jednak jest w stanie określić ich lokalizację na wyższym poziomie abstrakcji, np. kraj, region, serwerownia. Cechą charakterystyczną środowisk tego typu jest elastyczność oferowanych zasobów – istnieje możliwość zwiększania potrzebnych zasobów w sposób natychmiastowy na żądanie użytkownika, zgodnie z jego zapotrzebowaniem [1]. Model płatności, obecny aktualnie w przypadku chmur komercyjnych oferowanych przez takie firmy jak Amazon, Rackspace, przewiduje naliczanie stawek adekwatnych do parametrów używanych zasobów oraz czasu korzystania (brak narzuconych stałych miesięcznych opłat).

Ze względu na charakter udostępnianych zasobów wyróżnia się kilka modeli chmur.

1. IaaS (ang. *Infrastructure as a Service*) – użytkownik otrzymuje gotową infrastrukturę komputerową w postaci wirtualnego środowiska. Ma dostęp do całego systemu operacyjnego, magazynu danych oraz w niektórych rozwiązaniach ma limitowany dostęp do architektury sieciowej. Na dostarczonym systemie operacyjnym ma możliwość zainstalowania i uruchomienia aplikacji według własnego uznania. Warstwa ta całkowicie odcina się od architektury sprzętowej – użytkownika nie interesuje zarządzanie chmurą na niższym poziomie (sprzętowym).
2. PaaS (ang. *Platform as a Service*). Jest to wyższy poziom abstrakcji w stosunku do IaaS. W tym przypadku użytkownik otrzymuje gotowe środowisko do uruchomienia swojej aplikacji (tzn. zawiera ono system operacyjny oraz potrzebne oprogramowanie).
3. SaaS (ang. *Software as a Service*). Jest to najwyższy poziom abstrakcji. Istnieje tutaj możliwość uruchomienia gotowych aplikacji firmy świadczącej usługę chmury. Użytkownik nie ma wglądu do niżej leżących warstw struktury chmury (systemu operacyjnego, magazynu danych, narzędzi systemowych) – zarządzaniem wszystkimi warstwami zajmuje się usługodawca.

Z uwagi na sposób dostępu do zasobów chmury wyróżnia się następujące typy chmur:

- **publiczne** (ang. *public cloud*), dostępne dla szerokiej gamy użytkowników,
- **prywatne** (ang. *private cloud*), stworzone, wykorzystywane oraz zarządzane przez konkretne organizacje lub osoby trzecie; często znajdujące się w środowisku chronionym zaporą ogniową,
- **współdzielone przez społeczności** (ang. *community cloud*), używane przez organizacje mające wspólne cele, wymogi bezpieczeństwa oraz politykę, zarządzane przez te organizacje lub osoby trzecie,

- **mieszane** (ang. *hybrid cloud*) będące połączeniem chmur publicznych, prywatnych oraz współdzielonych poprzez społeczności [1].

3. Idea projektu

Celem projektu było zbudowanie prywatnej chmury w modelu IaaS (ang. *Infrastructure as a Service*). Przyjęto następujące najważniejsze założenia:

- chmura pozwala na uruchamianie maszyn wirtualnych (różnego typu systemów operacyjnych) zgodnie z zapotrzebowaniem użytkowników,
- jest łatwa we wdrożeniu – całe oprogramowanie niezbędne do działania chmury znajduje się na obrazie maszyny wirtualnej (stąd brak konieczności instalacji i konfiguracji oprogramowania w innym środowisku sprzętowym),
- jest środowiskiem automatycznym – automatyczna instalacja węzłów, automatyczna konfiguracja,
- jest łatwo skalowalna – rozszerzenie poziome zasobów wiąże się z dodaniem parametrów komputera do konfiguracji, a następnie rozpoczęciem procesu automatycznego przyłączenia do zasobów chmury,
- jest łatwa w zarządzaniu – zarządzanie odbywa się z poziomu jednego węzła,
- architektura chmury przewiduje grupowanie węzłów w podgrupy – mogą one mieć dedykowane zastosowanie oraz niezależność geograficzną.

4. Wykorzystane technologie

4.1. OpenNebula

OpenNebula [2] jest kluczowym oprogramowaniem użytym w realizowanym projekcie, stworzonym przez Distributed Systems Architecture Research (Complutense University, Madryt) [3] i pozwalającym na dynamiczne rozmieszczenie zasobów (wirtualnych maszyn) w rozproszonym środowisku, wg określonej polityki. Pozwala na szybkie dostarczenie oraz skalowalność serwisów, zgodnie z zapotrzebowaniem użytkowników. Umożliwia tworzenie chmur prywatnych (wykorzystujących technologie wirtualizacji XEN, KVM, VMware), chmur hybrydowych, rozszerzając funkcjonalność chmury prywatnej o zasoby Amazon EC2 [4] lub ElasticHosts [5] oraz chmur publicznych, udostępniając interfejs dostępowy EC2 (podzbiór pełnej funkcjonalności) [6], OGF OCCI [7]. Samo zarządzanie chmurą odbywa się z poziomu węzła głównego, posiadającego oprogramowanie OpenNebula. Dotychczasowe zastosowania

tego oprogramowania wykazują, że jest ono w stanie zarządzać tysiącami wirtualnych maszyn w środowisku rozproszonym (przykładem może być projekt realizowany w CERN [8]). Zaletą projektu jest pełna kontrola nad cyklem życia maszyny wirtualnej (dotyczy to np. sposobu uruchamiania, przegrywania obrazu, zatrzymywania), co stwarza możliwość dostosowania oprogramowania do różnych schematów działania (architektur).

Do innych, ważnych funkcji oprogramowania należą:

- zarządzanie węzłami wchodzącymi w skład chmury (np. grupowanie w obszary – klastry),
- zarządzanie obrazami wirtualnych maszyn (funkcjonalność repozytorium),
- zarządzanie instancjami wirtualnych maszyn (np. tworzenie, usuwanie, zapisywanie, migracja, restartowanie),
- zarządzanie wirtualnymi sieciami maszyn wirtualnych,
- kontekstualizacja (dostosowanie obrazów maszyn wirtualnych do pracy – np. ustawienie nazwy hosta, konfiguracja sieci, dogranie plików),
- zarządzanie użytkownikami (w tym przydzielanie limitów na zasoby, różne metody autoryzacji, np. wykorzystując bazę LDAP),
- monitorowanie zużycia zasobów, możliwość integracji z systemem pobierania płatności,
- kilka interfejsów dostępowych (np. XML-RPC API, LibVirt API, OGF OCCI API, EC2 Query API, Ruby/Java OpenNebula Cloud API),
- możliwość wykorzystania zewnętrznych pluginów, rozszerzeń oraz narzędzi (*The OpenNebula Ecosystem* [9]). Przykładem może być użycie zaawansowanego planisty zadań – Haizea [10].

Oprogramowanie OpenNebula stosuje wiele organizacji, m.in.: CERN (*European Organization for Nuclear Research*) [8], NIKHEF (*National Institute for Subatomic Physics*) [11], CRS4 (*Center for Advanced Studies, Research and Development in Sardinia*), NCHC (*National Center of High Performance Computing, Taiwan*), ESA (*European Space Astronomy Centre*), CESCA (*The Centre de Supercomputació de Catalunya*), China Mobile [12], Telefonica I+D i wiele innych. Powstało również kilka międzynarodowych inicjatyw, wykorzystujących oprogramowanie OpenNebula w zakresie badań oraz wyzwań technologicznych stawianych chmurom (np. federacje chmur, tworzenie struktur typu GRID w chmurach). Są to: EGEE (*Enabling Grids for E-sciencE*) [13], RESERVOIR (*Resources and Services Virtualization without Barriers*) [14], D-Grid (*German Grid Initiative*) [11], 4CaaSt (*Building the PaaS Cloud of the Future*) [15], StratusLab (*Enhancing Grids with Virtualization and Cloud*) [16], BonFIRE (*Building Service Testbeds on FIRE*) [17].

4.2. Fully Automatic Installation

FAI (ang. *Fully Automatic Installation*) [18] jest oprogramowaniem pozwalającym zautomatyzować proces instalacji komputerów wchodzących w skład chmury. Umożliwia bezobsługową instalację systemu Linux poprzez sieć wykorzystując DHCP+PXE+TFTP+NFS. Oprogramowanie to, stworzone przez Thomasa Lange, do tej pory znalazło zastosowanie w budowie dosyć dużych systemów – np. klastra o nazwie ATLAS na Max Planck Institute for Gravitational Physics, składającego się z 1342 węzłów [18]. Sama idea FAI opiera się na klasach pozwalających na grupowanie określonych typów maszyn oraz wykonywanie specyficznych dla tych grup instalacji oraz konfiguracji. Zaletą projektu FAI jest możliwość dostosowania każdego etapu instalacji wg własnych preferencji. W realizowanym projekcie Silesian Cloud oprogramowanie FAI zostało wykorzystane do automatycznej instalacji węzłów struktury chmury oraz do przeprowadzenia ich bazowej konfiguracji, przygotowującej do dalszego etapu wdrożenia przy wykorzystaniu projektu Puppet.

4.3. Puppet

Puppet [19] jest oprogramowaniem pozwalającym na automatyczną konfigurację oraz zarządzanie różnego typu systemami operacyjnymi. Posiada własny język opisu konfiguracji, która może zostać powielona dla większej liczby komputerów, minimalizując czas potrzebny na wdrożenie systemu oraz jego zarządzanie. Idea opiera się na obecności serwera Puppet (tzw. Puppetmaster) oraz klientów Puppet, obecnych na węzłach, odwołujących się do serwera w celu pobrania konfiguracji. Sam proces autoryzacji oraz pobierania/sprawdzania dostępnej dla węzła konfiguracji odbywa się cyklicznie szyfrowaną drogą (przy wykorzystaniu protokołu SSL). Oprogramowanie to może zostać wykorzystane do wykonywania różnego typu operacji administracyjnych, np. tworzenia użytkowników, plików, katalogów, zarządzania pakietami systemowymi itp. W realizowanym projekcie Silesian Cloud, oprogramowanie Puppet zostało wykorzystane do przeprowadzania konfiguracji węzłów, tuż po ich zainstalowaniu przy wykorzystaniu projektu FAI.

4.4. Pozostałe oprogramowanie

Do pozostałych technologii użytych w projekcie należą:

- MySQL [20] – baza danych używana przez oprogramowanie OpenNebula oraz mechanizm AFG (ang. *Automatic File Generator*), służący do generowania konfiguracji systemowej na podstawie informacji z tejże bazy.
- Python i wykorzystanie protokołu XML-RPC [21] – w języku Python stworzony został mechanizm AFG oraz pomocnicze skrypty używane w węzłach chmury, których zada-

niem jest komunikacja z serwerem XML-RPC (komunikuje się on z usługą OpenNebula poprzez API dostarczone do tego projektu).

- DHCP+PXE+TFTP+NFS – używane przez FAI w celu przeprowadzenia automatycznej instalacji oraz bazowej konfiguracji węzłów.
- GlusterFS [22] – klastrowy system plików posiadający skalowalność na poziomie petabajtów i obsługę tysięcy klientów. Agreguje moduły tzw. *storage bricks* w jedną przestrzeń roboczą, dostarczając takie mechanizmy, jak replikacja danych czy też rozmieszczenie danych w różnych węzłach, wg określonego algorytmu. W realizowanym projekcie używany jest do dystrybucji obrazów maszyn wirtualnych.
- KVM (ang. *Kernel-based Virtual Machine*) [23] – technologia wirtualizacji oferowana przez system Linux, pozwalająca na uruchamianie różnych systemów operacyjnych. W projekcie Silesian Cloud OpenNebula używa KVM do uruchamiania maszyn wirtualnych za pośrednictwem funkcji oferowanych przez projekt LibVirt [24].
- Bind9 [25] – oprogramowanie udostępniające funkcjonalność DNS. W projekcie użyte w celu realizacji nazw domenowych elementów struktury chmury.

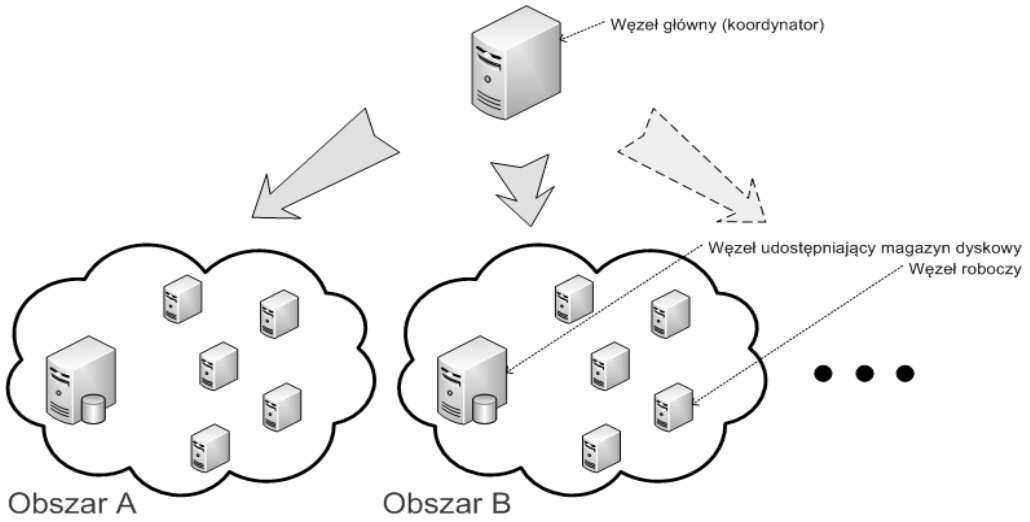
5. Architektura projektowanej chmury

Ze względu na realizowane funkcje w architekturze chmury tworzonej w ramach projektu można wyróżnić trzy rodzaje węzłów. Są to:

- węzeł główny (ang. *main node*) – pełni funkcję koordynatora całej struktury,
- węzły udostępniające magazyn dyskowy (ang. storage nodes) – służą do przechowywania obrazów maszyn wirtualnych,
- węzły robocze (ang. *worker nodes*) – komputery, na których uruchamiane są docelowo systemy wirtualne.

Dodatkowe założenia dotyczące struktury chmury Silesian Cloud:

- węzły robocze oraz węzły świadczące usługę magazynów danych są grupowane na podstawie nazwy domenowej w tzw. obszary (ang. *zones*),
- każdy obszar zawiera kilka serwerów roboczych oraz jeden serwer z obrazami maszyn wirtualnych (ang. *storage node*),
- serwery robocze korzystają wyłącznie z przydzielonego im w danym obszarze serwera z danymi maszyn wirtualnych,
- serwery z obrazami maszyn wirtualnych zawierają te same dane (są lustrzanymi odbiciami).



Rys. 1. Architektura chmury
Fig. 1. Cloud architecture

Idea obszarów niesie z sobą kilka zalet.

1. Możliwe jest umieszczenie określonych obszarów w różnych lokalizacjach geograficznych (w różnych centrach danych). W przypadku awarii jednego z obszarów niemożliwy jest dostęp tylko do części zasobów chmury – pozostałe obszary działają nadal (każdy z obszarów może posiadać własną sieć, zasilanie itp., co sprawia, że można go traktować jako osobną podgrupę).
2. Poszczególne obszary mogą być używane do serwowania danych produkcyjnych, pozostałe zaś mogą być używane do środowisk rozwojowych lub testowych.
3. Możliwy jest podział ze względu na zastosowanie maszyn wirtualnych (np. pod kątem użycia baz danych, aplikacji webowych czy też tworzenia środowisk programistycznych). Może wiązać się z tym również konkretna (dedykowana do danego zastosowania) konfiguracja sprzętowa węzłów wchodzących w skład danego obszaru.

6. Elementy składowe projektowanej chmury

Poniżej przedstawiony został opis poszczególnych elementów struktury projektowanej chmury – ich budowy z uwzględnieniem użytego oprogramowania oraz realizowanych funkcji.

Węzeł główny (ang. *main node*) koordynuje pracę całej struktury. Udostępnia mechanizmy służące automatycznej (bezobsługowej) instalacji węzłów wykorzystywanych do przechowywania obrazów maszyn wirtualnych oraz węzłów roboczych, na których uruchamiane są docelowo maszyny wirtualne. Zajmuje się zarządzaniem wirtualnymi maszynami, uruchamianymi w węzłach roboczych. Posiada mechanizmy automatycznej konfiguracji zarówno

innych elementów struktury, jak i własnej. System ten ma postać obrazu maszyny wirtualnej (stwarza to możliwość szybkiego uruchomienia projektu w innym środowisku – dostarczenie gotowego rozwiązania).

Węzeł ten posiada zaimplementowane następujące usługi:

- OpenNebula – najważniejsze oprogramowanie realizowanego projektu zajmujące się zarządzaniem chmurą.
- DHCP + PXE + TFTP – odpowiedzialność za adresację komputerów w sieci oraz serwowanie danych na potrzeby mechanizmu FAI.
- FAI (*Fully Automatic Installation*) – odpowiedzialność za automatyczną instalację oraz bazową konfigurację węzłów. Bazowy system, który instalowany jest w węzłach, umieszczony jest na serwerze NFS.
- Bind9 (DNS) – realizuje wewnętrzną domenę chmury. Pozwala na odwoływanie się do węzłów po ich nazwach domenowych – brak konieczności przechowywania informacji o adresacji IP na węzłach.
- Puppet (serwer) – zajmuje się automatyczną konfiguracją węzłów po ich instalacji.
- MySQL – baza danych wykorzystywana na potrzeby projektu OpenNebula (przechowuje wewnętrzne informacje, generowane przez to oprogramowanie – np. dane hostów, maszyn wirtualnych), jak również na potrzeby mechanizmu AFG.
- AFG (*automatic file generator*) – mechanizm stworzony na potrzeby tego projektu pozwalający na konfigurację usług DNS, DHCP, NFS oraz innych elementów systemu Linux węzła głównego, zgodnie z informacjami (dotyczącymi węzłów wchodzących w skład struktury chmury) dodanymi wcześniej do bazy danych.
- GlusterFS (klient) – zajmuje się automatyczną replikacją danych na serwery magazynujące dane. Używany jest przy dystrybucji obrazów maszyn wirtualnych na te serwery.
- pomocniczy serwer XML-RPC – stworzony na potrzeby projektu serwer, który zajmuje się interakcją z oprogramowaniem OpenNebula poprzez udostępnione w tym projekcie API. Serwer ten wykorzystywany jest do automatycznej rejestracji węzłów (zaraz po zainstalowaniu) w usłudze OpenNebula.

Węzły udostępniające magazyn dyskowy – komputery, których zadaniem jest udostępnianie obrazów maszyn wirtualnych. Posiadają nazwy hostów w formacie: zone-X-storage01, gdzie X określa numer identyfikacyjny obszaru, do którego należy węzeł. Węzeł ten jest:

- automatycznie instalowany oraz konfigurowany poprzez sieć (wykorzystane *Fully Automatic Installation*), proces ten obejmuje:
  - bezobsługowe tworzenie partycji systemowych,
  - instalowanie pakietów systemowych systemu Linux Debian Squeeze, wymaganych do pracy tego typu węzła,

  - konfigurację obszaru rozruchowego,
  - przygotowanie klienta Puppet (aktywacja działania) do pobierania konfiguracji z serwera;
- automatycznie konfigurowany – konfiguracja jest automatycznie pobierana po zainstalowaniu czystego węzła z węzła głównego i odbywa się za pomocą klienta Puppet. Autoryzacja klienta Puppet do serwera tego oprogramowania, uruchomionego na węźle głównym, odbywa się również w sposób automatyczny. Sam proces konfiguracji obejmuje:
  - konfigurację serwera GlusterFS,
  - udostępnienie poprzez NFS zasobu dyskowego (służącego do przechowywania obrazów maszyn wirtualnych) dla węzłów roboczych.

Węzły robocze – węzły, na których uruchamiane są maszyny wirtualne. Posiadają nazwy hostów w formacie: zone-X-kvm[0-9]+ (np. zone-A-kvm01 oznacza pierwszy węzeł tego typu w obszarze A i jest równoznaczne z wykorzystaniem serwera magazynującego obrazy maszyn wirtualnych zone-A-storage01). Węzeł ten jest:

- automatycznie instalowany oraz konfigurowany poprzez sieć (wykorzystane *Fully Automatic Installation*):
  - bezobsługowe tworzenie partycji systemowych,
  - instalowanie pakietów systemowych systemu Linux Debian Squeeze, wymaganych do pracy tego typu węzła,
  - konfiguracja obszaru rozruchowego,
  - automatyczna konfiguracja interfejsu sieciowego na potrzeby wirtualizacji. Stworzony mechanizm automatycznie wykrywa interfejs sieciowy aktualnie używany podczas instalacji (badanie trasy sieciowej) i następnie tworzy konfigurację dla urządzenia sieciowego typu bridge,
  - przygotowanie klienta Puppet (aktywacja) do pobierania konfiguracji z serwera;
- automatycznie konfigurowany – konfiguracja jest automatycznie pobierana po zainstalowaniu czystego węzła z węzła głównego (wykorzystanie Puppet) i obejmuje:
  - konfigurację użytkownika, grupy, katalogu domowego, nadanie praw, stworzenie struktury katalogów niezbędnych do działania usługi OpenNebula,
  - przesłanie kluczy ssh, potrzebnych do autoryzacji bez hasła z poziomu głównego węzła (wymagane do działania usługi OpenNebula),
  - zamontowanie katalogów serwera, na którym trzymane są obrazy maszyn wirtualnych,
  - konfigurację KVM;
- automatycznie rejestrowany w strukturze chmury – za pośrednictwem serwera Puppet przesyłany jest klient XML-RPC, który uruchamiany jednorazowo po zainstalowaniu węzła komunikuje się z serwerem XML-RPC na węźle głównym (ten z kolei dokonuje automatycznej rejestracji zgłaszanego węzła w usłudze OpenNebula).

7. Funkcjonowanie projektowanej chmury

Po uruchomieniu, stworzonego w ramach projektu, obrazu maszyny wirtualnej, zawierającej zaimplementowane mechanizmy (wymienione wcześniej) można przystąpić do budowy chmury. Zasada działania oprogramowania została przedstawiona w dwóch etapach. Pierwszy etap obejmuje okres od momentu dodania czystego węzła do konfiguracji aż do momentu zainstalowania oraz skonfigurowania oprogramowania, niezbędnego do działania tego węzła w chmurze. Etap drugi z kolei opisuje funkcjonowanie usługi OpenNebula – przedstawione zostało działanie zmodyfikowanych sterowników transportowych na przykładzie uruchamiania oraz zatrzymywania maszyny wirtualnej.

7.1. Etap I - instalacja oraz bazowa konfiguracja

Etap rozpoczyna się od dodania czystego węzła do konfiguracji w węźle głównym. W tym celu wykorzystuje się mechanizm AFG (*Automatic File Generator*), który dodaje określoną przez administratora konfigurację – adres IP, adres MAC, nazwa hosta – do systemu oraz bazy danych.

Po dodaniu węzła do konfiguracji przystępuje się do procesu instalacji poprzez sieć, przy wykorzystaniu mechanizmu FAI. Po załadowaniu instalatora FAI (przy wykorzystaniu DHCP+PXE+TFTP+NFS), w zależności od nazwy hosta podanej wcześniej przez administratora do mechanizmu AFG (np. zone-A-kvm01, zone-A-storage01), dany węzeł zostaje przypisany do klasy, która określa jego profil instalacji oraz konfiguracji. W realizowanym projekcie wszystkie węzły należą do klasy NODE (określa ona np. zainstalowanie wspólnych dla wszystkich typów węzłów pakietów, sposób partycjonowania dysków), dodatkowo węzły robocze należą do klasy KVM, natomiast węzły magazynujące obrazy maszyn wirtualnych do klasy STORAGE. Po przypisaniu węzła do danej klasy następuje etap instalacji, obejmujący partycjonowanie dysku, instalację pakietów bazowych oraz dodatkowych, określonych do konkretnej klasy. W końcowym etapie przeprowadzana jest podstawowa konfiguracja, również zależna od nazwy klasy. Dla węzłów roboczych ważnym elementem na tym etapie jest stworzenie konfiguracji urządzenia sieciowego typu bridge, które będzie wykorzystywane przez usługę OpenNebula, w celu udostępniania maszynom wirtualnym sieci. W tym celu jest wykorzystywany stworzony mechanizm, wykrywający aktualnie używany interfejs sieciowy (badanie drogi sieciowej) i tworzący z niego odpowiednią konfigurację. Jednym z końcowych elementów procesu instalacji jest aktywacja klienta Puppet, w celu późniejszego pobierania konfiguracji z serwera głównego. Po zakończeniu instalacji następuje restart maszyny oraz uruchomienie zainstalowanego systemu. Podczas uruchamiania węzła z zainstalowanym systemem następuje aktywacja klienta Puppet, który następnie komunikuje się z serwerem tej

aplikacji (autoryzacja następuje w sposób automatyczny), znajdującym się na węźle głównym i pobiera konfigurację. Pobierana konfiguracja zależy od typu węzła (a dokładnie od jego nazwy hosta). Przewiduje tworzenie katalogów, użytkowników oraz przesłanie kluczy ssh, potrzebnych do autoryzacji bez hasła. Dla węzłów roboczych dodatkowo zostaje przesłany klient XML-RPC, który jest uruchamiany jednorazowo. Zadaniem tego klienta jest odwołanie się do stworzonego serwera XML-RPC, znajdującego się na węźle głównym, w celu rozpoczęcia procedury rejestracji tego węzła w usłudze OpenNebula, poprzez dostarczone do tego projektu API. Rejestracja do danego obszaru odbywa się na podstawie nazwy hosta węzła. Przykładowo węzeł o nazwie zone-A-kvm01 zostanie przypisany do obszaru o nazwie zone-A, a jeśli obszar ten nie istnieje, wówczas zostanie on stworzony automatycznie.

7.2. Etap II - OpenNebula

Posiadając zainstalowaną oraz skonfigurowaną strukturę chmury można przystąpić do etapu jej zarządzania, przy wykorzystaniu oprogramowania OpenNebula. Oprogramowanie to posiada szeroki wachlarz funkcji. W tym rozdziale zostały przedstawione jedynie najważniejsze elementy dodane oraz zmodyfikowane na potrzeby realizowanego projektu. Są to odpowiednio mechanizm replikacji obrazów wirtualnych maszyn oraz zmodyfikowany sterownik transportowy usługi OpenNebula, którego rola polega na zarządzaniu obrazami wirtualnych maszyn - przegrywaniem, klonowaniem, usuwaniem oraz tworzeniem. Dokonane modyfikacje polegają na wsparciu zewnętrznego serwera magazynującego dane obrazów maszyn wirtualnych oraz na tworzeniu obrazu różnicowego w formacie *copy-on-write* (*qcow2*) [26], w chwili tworzenia instancji maszyn wirtualnych.

W chwili rejestracji obrazu maszyny wirtualnej na głównym węźle jest on automatycznie replikowany (wykorzystanie projektu GlusterFS) do serwerów magazynujących dane, a dokładnie mówiąc do lokalizacji udostępnianej węzłom roboczym poprzez udział NFS. W momencie zapotrzebowania uruchomienia maszyny wirtualnej (zlecenie wykonywane jest poprzez węzeł główny) automatycznie wybierany jest węzeł roboczy w danym obszarze, posiadający w tym momencie wolne zasoby. Wówczas tworzony jest obraz *copy-on-write* (*qcow2*). Wszystkie zmiany wykonane w stosunku do wzorcowego obrazu maszyny wirtualnej, umieszczonego w węźle magazynującym, przechowywane są w pliku różnicowym (delta), tworzonym bezpośrednio na dysku węzła roboczego. W momencie zapisywania stanu maszyny wirtualnej (zatrzymywaniu jej) obraz *copy-on-write* zapisywany jest w węźle magazynującym dane – dzięki temu przy późniejszym odtworzeniu zapisanej maszyny wirtualnej może zostać on przegrany na inny węzeł, posiadający w tej chwili wolne zasoby. Wykorzystanie pliku różnicowego pozwala na bardzo szybkie tworzenie kolejnych instancji wirtualnych maszyn (brak konieczności kopiowania całego obrazu wirtualnej maszyny).

8. Status projektu

Struktura projektowanej chmury ulega ciągłej ewolucji – wszystkie omówione w artykule elementy chmury zostały sprawdzone eksperymentalnie. W aktualnie badanej architekturze serwery uruchamiające maszyny wirtualne (*worker nodes*) są jednocześnie serwerami magazynującymi dane (*storage nodes*). Przy takim podejściu za pomocą systemu plików GlusterFS utworzono wspólną przestrzeń dyskową w konfiguracji *distributed-replicated* (odpowiednik RAID10). Obrazy maszyn wirtualnych są składowane na wspólnej przestrzeni, a przed uruchomieniem są kopiowane na lokalny magazyn dyskowy węzła (*workera*).

Ponadto, w ramach projektu zrealizowano następujące rozwiązania:

- mechanizm automatycznego budowania systemu operacyjnego dla węzła głównego (tworzenie dedykowanego systemu operacyjnego na żądanie),
- pełna kontrola kodu przy wykorzystaniu systemu kontroli wersji (+ mechanizm automatycznego budowania pakietów systemu Debian),
- nacisk na technologię wirtualizacji XEN,
- API w języku Python dla usługi OpenNebula (OCA Python API) – zostało przyjęte przez twórców tego oprogramowania,
- mechanizm monitorowania zużycia zasobów (czasu kopiowania obrazów, liczby uruchomionych wirtualnych maszyn przez użytkowników),
- mechanizm synchronizacji czasowej wszystkich węzłów w obrębie chmury,
- wiele testów wydajnościowych GlusterFS, które pozwoliły na dobranie optymalnego sposobu korzystania (opcji tuningowych, sposobu montowania zasobów GlusterFS) z tego systemu plików,
- mechanizm wykonywania zadań z poziomu serwera głównego (*main node*) na innych serwerach, przy wykorzystaniu systemu kolejkowego RabbitMQ [27] jako warstwy transportowej oraz projektu Celery, pozwalającego na rozproszone uruchamianie zadań [28],
- zapewnienie izolacji sieciowej uruchomionych wirtualnych maszyn (zabezpieczenie przed zmianą adresów MAC/IP przez użytkowników wirtualnych maszyn – *de facto* przed przejęciem IP wirtualnej maszyny innego systemu w infrastrukturze).

9. Przykładowe zastosowania zbudowanej chmury

Wachlarz zastosowań chmur jest bardzo szeroki. Poniżej przedstawiono kilka przykładowych zastosowań chmury:

- funkcjonalność prywatnych serwerów dla użytkowników (hosting),

- przeniesienie aktualnej infrastruktury IT do chmury na różnych poziomach:
  - serwerów (całych systemów operacyjnych),
  - aplikacji (konkretnego oprogramowania, np. baz danych, serwerów stron WWW),
- wykonywanie obliczeń wymagających dużych mocy obliczeniowych (HPC w chmurze) – np. z wykorzystaniem mechanizmów programowania współbieżnego MPI,
- grupowanie systemów wirtualnych – uruchamianie klastrów wysokiej dostępności oraz zrównoważonego obciążenia w chmurze,
- przeprowadzanie testów aplikacji użytkowników – wykorzystanie Selenium oraz integracja z systemem ciągłej integracji, np. Hudson.

10. Podsumowanie

Technologia *Cloud computing* staje się bardzo popularna. Według prognoz Gartnera do 2012 roku 20% firm będzie korzystać z usług bazujących na chmurach, wirtualizacji oraz komputerach roboczych, uruchamianych w sieciach korporacyjnych [29]. Stworzony projekt wpisuje się w ten trend, udostępnia zasoby w postaci wirtualnych maszyn (model IaaS chmury). Oprogramowanie użyte w projekcie – np. OpenNebula – sprawdza się w dużych projektach prowadzonych przez takie organizacje, jak: CERN, ESA czy też ChinaMobile. Automatyzacja oprogramowania infrastruktury chmury oraz łatwość skalowania są zaletami projektu – dzięki stworzeniu obrazu maszyny wirtualnej dostarczającej całą funkcjonalność projekt może być wdrożony przy małym nakładzie czasu. Oczywiście jest to koncepcja wymagająca dalszej pracy – stąd też niektóre rozwiązania mogą ulec rozbudowie, modyfikacji lub wymianie na inne.

BIBLIOGRAFIA

1. The National Institute of Standards and Technology Definition of Cloud Computing, http://csrc.nist.gov/groups/SNS/cloud-computing/.
2. OpenNebula.org OpenSource Toolkit for Cloud Computing, http://opennebula.org/.
3. DSA (Distributed Systems Architecture) Research Group at Complutense University of Madrid, http://dsa-research.org/.
4. Amazon Elastic Compute Cloud (Amazon EC2), http://aws.amazon.com/ec2/.
5. ElasticHosts, http://www.elastichosts.com/.
6. Amazon EC2 Query Interface, http://docs.amazonwebservices.com/AWSEC2/2009-04--04/DeveloperGuide/index.html?using-query-api.html.

7. OGF OCCI API, http://www.occi-wg.org/.
8. OpenNebula implementation at CERN, http://lists.opennebula.org/pipermail/users-opennebula.org/2010-April/001886.html.
9. The OpenNebula Ecosystem, http://www.opennebula.org/community:ecosystem.
10. Haizea open-source virtual machine-based lease management, http://haizea.cs.uchicago-.edu/.
11. Big Grid Virtualization of worker nodes. Working group progress report, https://wiki-.nbic.nl/images/f/f6/ProgressReport-1.0.pdf.
12. China Mobile Cloud Computing Research and Site Update, https://opencirrus.org-/system/files/DAY%20TWO_07_Hong_China%20Mobile.pdf.
13. EGEE - Enabling Grids for E-sciencE, http://www.eu-egee.org/.
14. RESERVOIR – Resources and Services Virtualization without Barriers, http://www-.reservoir-fp7.eu/.
15. #4CaaSt – Building the PaaS Cloud of the Future, http://4caast.morfeo-project.org/.
16. StratusLab – Enhancing Grids with Virtualization and Cloud, http://www.stratuslab-.eu/doku.php.
17. BonFIRE – Building Service Testbeds on FIRE, http://www.bonfire-project.com/.
18. FAI Project, http://fai-project.org.
19. Puppet Labs, http://www.puppetlabs.com/.
20. MySQL, http://www.mysql.com/.
21. Python Programming Language, http://python.org/.
22. GlusterFS, http://www.gluster.org/.
23. Kernel-based Virtual Machine, http://www.linux-kvm.org/page/Main_Page.
24. LibVirt: The virtualization API, http://libvirt.org/.
25. BIND, http://www.isc.org/software/bind.
26. The QCOW2 Image Format, http://people.gnome.org/~markmc/qcow-image-format.html.
27. RabbitMQ, http://www.rabbitmq.com/.
28. Celery Project, http://celeryproject.org/.
29. Gartner Highlights Key Predictions for IT Organizations and Users in 2010 and Beyond, http://www.gartner.com/it/page.jsp?id=1278413.

Recenzenci: Prof. dr hab. inż. Tadeusz Czachórski
Prof. dr hab. inż. Henryk Krawczyk

Wpłynęło do Redakcji 4 kwietnia 2011 r.

Abstract

The paper presents the idea of cloud computing and the prototype system built in IaaS model (called Infrastructure as a Service). Architecture of the developed cloud is based on Linux, KVM virtualization technology (called Kernel-based Virtual Machine) and on open source software OpenNebula used to build a cloud computing environments. Infrastructure of the cloud gives the possibility to run on-demand ready operating systems as virtual machines in a distributed environment consisting of multiple computers called nodes. Computers included in this structure are divided according to the performed functions. There are the following nodes: the master node (coordinator, resource manager), the working nodes (used to run virtual machines) and the nodes storing the data (virtual machines images). Authors emphasized the system automation - the components of the environment are automatically installed, configured and registered in the service responsible for monitoring and managing the resources (virtual machines). This approach minimizes the time needed for system management and makes it easily horizontally scalable. Thanks to it, adding a new node to infrastructure takes only a few minutes. The whole cloud was built using the leading open source technologies based on Linux (KVM, OpenNebula, FAI, Puppet, GlusterFS, BIND, MySQL, Python and others). The paper presents the details of system working. Exemplary applications of the cloud are indicated in the paper. The paper presents the first step of the system development – it is as yet rather concept, which requires further work.

Adresy

Piotr KANDZIORA: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, raveenpl@gmail.com
Stanisław KOZIELSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, stanislaw.kozielski@polsl.pl