2010 IEEE 3rd International Conference on Cloud Computing

How to Enhance Cloud Architectures to Enable Cross-Federation

Antonio Celesti*, Francesco Tusa*, Massimo Villari* and Antonio Puliafito*
*Università degli Studi di Messina, Facoltà di Ingegneria
Contrada di Dio, S. Agata, 98166 Messina, Italy.
e-mail: {acelesti,ftusa,mvillari,apuliafito}@unime.it

***Abstract*—The near future evolution of the cloud computing can be hypothesized in three subsequent stages: stage 1 "Monolithic" (now), cloud services are based on independent proprietary architectures; stage 2 "Vertical Supply Chain", cloud providers will leverage cloud services from other providers; stage 3 "Horizontal Federation", smaller, medium, and large cloud providers will federate themselves to gain economies of scale and an enlargement of their capabilities. Currently, the major clouds are planning the transition to the stage 2, but how to achieve the stage 3 is unclear because some architectural limitations have to be overcome. In this paper, considering a general cloud architecture, we highlight such limitations and propose some enhancements which add new federation capabilities. In order to address such concerns we propose a solution based on the Cross-Cloud Federation Manager, a new component placeable inside the cloud architectures, allowing a cloud to establish the federation with other clouds according to a three-phase model: discovery, match-making and authentication.**

***Keywords*-Cloud Computing, Cross(-Cloud) Federation, Cloud Architecture, Cloud Infrastructure, Heterogeneous Systems.**

I. Introduction

Cloud computing, the new computation paradigm, is pursuing new levels of efficiency in delivering services, representing a tempting business opportunity for IT operators of increasing their revenues. Services range from software to platform or infrastructure (SaaS, PaaS, IaaS), while clients might include other clouds, organizations, enterprises and single users.

Nowadays cloud environments include hundreds of independent, heterogeneous, private/hybrid clouds, but many business operators have predicted that the process toward an interoperable federated cross cloud scenarios will begin in the near future. As it has been claimed in [1], the evolution of the cloud computing market can be hypothesized in three subsequent stages: stage 1 "Monolithic" (now), cloud services are based on proprietary architectures - islands of cloud services delivered by megaproviders (this is what Amazon, Google, Salesforce and Microsoft look like today); stage 2 "Vertical Supply Chain", over time, some cloud providers will leverage cloud services from other providers. The clouds will be proprietary islands yet, but the ecosystem building will start; stage 3 "Horizontal Federation", smaller, medium, and large providers will federate horizontally themselves to gain: economies of scale, an efficient use of their assets, and an enlargement of their capabilities.

This work describes how to make up an interoperable heterogeneous cloud environment in "Horizontal Federation" configuration, where clouds can cooperate together accomplishing trust contexts and providing new business opportunities such as cost-effective assets optimization, power saving, and on-demand resources provisioning. In particular, we carry out what the involved issues are, analyzing a resource provisioning scenario and proposing an architectural solution and an implementation practice. For simplicity, in the rest of the paper, with terms such as "cross-cloud federation", "federation in cloud computing", or "cloud federation" we will refer to the above mentioned "Horizontal Federation".

We propose a *three-phase cross-cloud federation model* where the federation establishment, between a cloud needing external resources and a cloud offering resources, passes through three main phases: *discovery*, the cloud looks for other available clouds; *match-making*, the cloud selects between the discovered clouds the ones which fit as much as possible its requirements; *authentication*, the cloud establishes a trust context with the selected clouds. According to our three-phase model, we designed a module named Cross-Cloud Federation Manager (CCFM) compliant with each cloud architecture which includes three agents (Discovery, Match-making and Authentication) responsible to accomplish the aforementioned three phases. Such module represents an architectural solution which can add to any cloud cross-cloud federation capabilities, regardless its implementation.

We underline our solution is aimed to provide new functionalities to cloud providers. The solutions we introduced have the goal to setup a new enhanced cloud environment in which operators can easily apply their business models. In particular, we describe what the parameters are and how they can be setup to satisfy time by time the customer's demand.

We have also selected new technologies being in the IT world, for addressing such issues. They are XML based and range from XMPP, and XACML to SAML.

The paper is organized as follows. Section II describes the state of the art of the existing cloud middlewares, highlighting their architectural limitations regarding the cross-cloud federation perspective and underlining how some clouds

978-0-7695-4130-3/10 $26.00 © 2010 IEEE
DOI 10.1109/CLOUD.2010.46

have begun the transition from the stage 1 to the stage 2 recently. In Section III we provide a detailed analysis the requirements that, in the next future, clouds should have for the transition from the stage 2 to the stage 3. In order to achieve such goal, we consider a resource provisioning scenario of cooperating clouds, introducing the concepts of *home cloud* and *foreign cloud*. In order to achieve the stage 3, in section IV, we propose a possible enhancement to the cloud architectures for the addition of cross-cloud federation capabilities. More specifically we introduce our own three-phase federation model (discovery, match-making and authentication) and present the Cross-Cloud Federation Manager (CCFM), a module deployable in any cloud architecture with the minimal impact and responsible to accomplish the three federation phases by means of three autonomous agents. In Section V, we present an overview of the technologies which could be adopted to implement the module. More specifically, considering a reference scenario composed by several cloud, we show how the cross-cloud federation could be accomplished. Conclusions and lights to the future are summarized in section VI.

II. Related Work

This Section describes the current state-of-the-art in Cloud Computing analyzing the main existing middleware implementations, evaluating their main features, and highlighting how they lack of cross-cloud federation features.

Federation is an interesting research area in cloud computing and recently, new terms are been coined as Intercloud ("Think of the existing cloud islands merging into a new, interoperable Intercloud where applications can be moved to and operate across multiple platforms..." [2]) or Cross-cloud ("For the benefit of human society and the development of cloud computing, one uniform and interoperable Cross-cloud platform will surely be born in the near future..." [3]).

On the market are currently available both proprietary (i.e. Amazon EC2 [4], Microsoft Azure [5], Saleforce, Google and Yahoo) and several open source cloud middlewares. Considering the open source projects we distinguish two main categories: Low-Level Management, including projects such as OpenQRM [6] and OpenNebula [7], and High-Level Management including Nimbus [8], Eucalyptus [9].

OpenQRM is an open-source platform for enabling flexible management of computing infrastructures. Thanks to its pluggable architecture, OpenQRM is able to implement a cloud with several features that allows the automatic deployment of services. It supports different virtualization technologies managing Xen, KVM and Linux-VServer VEs. It also supports P2V (physical to virtual), V2P (virtual to physical) and V2V (virtual to virtual) migration. This means VEs (appliances in the OpenQRM terminology) can not only easily move from physical to virtual (and back), but that they can also be migrated from different virtualization technologies, even transforming the server image. OpenQRM is able to grant a complete monitor of systems and services by means of the Nagios tool [10] which maps the entire openQRM network and creates (or updates) its corresponding configuration (i.e. all systems and available services). Finally, OpenQRM addresses the concepts related to High Availability (HA) systems: virtualization is exploited to allow users to achieve services fail-over without wasting all the computing resources (e.g. using stand-by systems).

OpenNebula is an open and flexible tool that fits into existing data center environments to build a Cloud computing environment. OpenNebula can be primarily used as a virtualization tool to manage virtual infrastructures in the data-center or cluster, which is usually referred as Private Cloud. Only the more recent versions of OpenNebula are trying to supports Hybrid Cloud to combine local infrastructure with public cloud-based infrastructure, enabling highly scalable hosting environments. OpenNebula also supports Public Clouds by providing Cloud interfaces to expose its functionalities for virtual machine, storage and network management.

Nimbus is an open source toolkit that allows to turn a set of computing resources into an Iaas cloud. Nimbus comes with a component called workspace-control, installed on each node, used to start, stop and pause VMs, implements VM image reconstruction and management, securely connects the VMs to the network, and delivers contextualization. Nimbus's workspace-control tools work with Xen and KVM but only the Xen version is distributed. Nimbus provides interfaces to VM management functions based on the WSRF set of protocols. There is also an alternative implementation exploiting Amazon EC2 WSDL. The workspace service uses GSI to authenticate and authorize creation requests. Among others, it allows a client to be authorized based on Virtual Organization[VO] role information contained in the VOMS credentials and attributes obtained via GridShib.

Eucalyptus [9] is an open-source cloud-computing framework that uses the computational and storage infrastructures commonly available at academic research groups to provide a platform that is modular and open to experimental instrumentation and study. Eucalyptus addresses several crucial cloud computing questions, including VM instance scheduling, cloud computing administrative interfaces, construction of virtual networks, definition and execution of service level agreements (cloud/user and cloud/cloud), and cloud computing user interfaces.

Regarding the cross-cloud federation perspective, currently, OpenQRM does not have federation features, instead OpenNebula, Nimbus and Eucalyptus are almost at the stage 2 (Vertical Supply Chain), being compliant with he Amazon EC2 web service interface, which is becoming "the standard de-facto" to use external cloud services. Since, a standard guideline of how a cloud architecture should be made does not exist, each cloud middleware may be deeply different from each other. In such scenario the transition at the stage

3 (Horizontal Federation) could be thus very difficult.

III. General Concepts of Cross-Cloud Federation

In this Section we try to clarify some ideas concerning what the Horizontal Federation (stage 3) really will imply, explaining the general concept of the cross-cloud federation. In order to identify both requirements and goals, we propose a possible resource provisioning scenario where clouds might benefit of federation advantages. Cloud Computing relies its computational capabilities exploiting the concept of "virtualization". This technology is re-emerged in recent years as a compelling approach for increasing resource utilization and reducing IT services costs. The common theme of all the virtualization technologies refers to the cabability of hiding the underlying infrastructure, introducing a logical layer between the physical infrastructure and the computational processes. The virtualization is being possible thanks to Virtualization Machine Monitors (VMMs commonly known as "hypervisors"), i.e. processes that run on top of a given hardware platform, control and emulate one or more other computer environments (i.e. the virtual machines). Each of these virtual machines, in turn, runs its respective "guest" software, typically an operating system, executed as if it is installed on a stand-alone hardware platform.

Private clouds hold their own virtualization infrastructure where several virtual machines are hosted to provide services to their clients. In a scenario of "cross-cloud federation", each cloud operator is able to transparently enlarge its own virtualization resources amount (i.e. increasing the number of instantiable virtual machines) asking further computing and storage capabilities to other clouds. Consequently, the cloud operator will be able to satisfy any further service allocation request sent by its clients.

According to our analysis, within the above mentioned scenario we distinguish two types of cloud: *home cloud* and *foreign cloud*. Home cloud is a cloud provider which is unable to instantiate further virtual machines as the capability of its virtualization infrastructure is saturated and consequently, forwards federation requests to foreign clouds with the purpose to host its own virtual machines on their virtualization infrastructures. Instead, foreign cloud is a cloud provider which leases part of the storage and computing capabilities of its virtualization infrastructure to home clouds for free or by charge. A cloud provider could be at the same time both home cloud and/or foreign cloud.

In order to better explain such idea, we consider the reference scenario depicted in Figure 1 where the home cloud is already at the stage 2 because is able to provide services to other clouds (top part of the Figure). Moreover, the home cloud is also at the stage 3, in fact, when it realizes that its virtualization infrastructure has saturated its capabilities, in order to continue providing services to its clients (i.e. other clouds, enterprises, generic end users, etc), it decides to federate itself with foreign clouds A and B. The home cloud, besides hosting virtualization resource inside its own virtualization infrastructure, is also able to hosting virtual machines inside the foreign clouds A and B virtualization infrastructures, enlarging the amount of its available virtualization resources (See Figure 1, bottom part). Therefore, although the virtualization resources rent to the home cloud are physically placed within the virtualization infrastructures of foreign clouds A and B, they are logically considered as resources indeed hosted within the home cloud virtualization infrastructure.

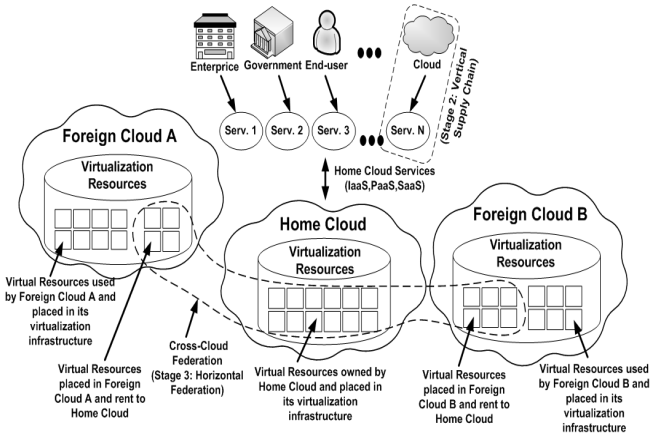

Figure 1. Cross-Cloud Scenario: basic for heterogeneous and federated clouds.

Despite the obvious advantages, the implementation of such cross-cloud federation scenario is not trivial at all because clouds are more complicated than traditional systems and the existing federation models are not applicable. In fact, while clouds are typically heterogeneous and dynamic, the existing federation models are designed for static environments where a priori agreements among the parties are needed to establish the federation. Keeping in mind the aforementioned scenario, we think cloud federation needs to meet the following requirements:

- a) *automatism and scalability*, a home cloud, using discovery mechanisms, should be able to pick out the right foreign clouds which satisfies its requirements reacting also to cloud changes;
- b) *interoperable security*, it is needed the integration of different security technologies, permitting a home cloud, for instance, to be able to join the federation without changing its security policies.

In the "interoperable security" context we identify: *1) Single Sign-On (SSO) authentication*, a home cloud should be able to authenticate itself once, gaining the access to the resources provided by the federated foreign clouds belonging to the same trust context, without further identity checks; *2)*

digital identities and third parties, each home cloud should be able to authenticate itself with the foreign clouds using its digital identity guaranteed by a third party. This latter feature is more challenging because it implies a cloud has to be considered as a subject uniquely identified by some credentials.

IV. Our Enhancements of Cloud Architectures for the Federation

In Section III, we described the concept of cross-cloud federation and its bindings with the cloud. In the following, we provide a detailed description of our architectural solution to address the cross-cloud federation issues.

A. The Three-Phase Federation Model

The cross-cloud federated infrastructure can be intended as an ephemeral interconnection of clouds, each laying out its own set of resources and exploiting different authentication mechanisms. Due to the high dynamism of a cross-cloud federated infrastructure, a flexible method for building dynamic interactions and enabling the coexistence of different and heterogeneous technologies should be provided. Our solution tries to answer all such issues considering the requirements of automatism, scalability and interoperability previously stated. Describing the federation process we point out three main different phases: *discovery*, *match-making* and *authentication*. These phases are opportunely explained in the following.

B. The Cross-Cloud Federation Manager for the Resource Provisioning

In order to identify the main components constituting a cloud and better explain the federation idea on which our work is based, we are considering the internal architecture of each cloud as the three-layered stack [11] presented schematically in Figure 2. Starting from the bottom, we can identify: *Virtual Machine Manager*, *Virtual Infrastructure (VI) Manager* and *Cloud Manager*. The VI manager is a fundamental component of private/hybrid clouds acting as a dynamic orchestrator of Virtual Environments (VEs), which automates VEs setup, deployment and management, regardless of the underlying Virtual Machine Manager layer (i.e Xen, KVM, or VMware). The Cloud Manager layer is instead able to transform the existing infrastructure into a cloud, providing cloud-like interfaces and higher-level functionalities for security, contextualization and VM disk image management.

In a cloud architecture designed according to the aforementioned three-layered stack, all the cloud components and their respective functions are clearly defined and separated, thus introducing simplicity and efficiency when the cloud middleware has to be modified or new functionalities have to be added. In our work, we exploited such modular characteristics of the layered cloud architecture, and introduced

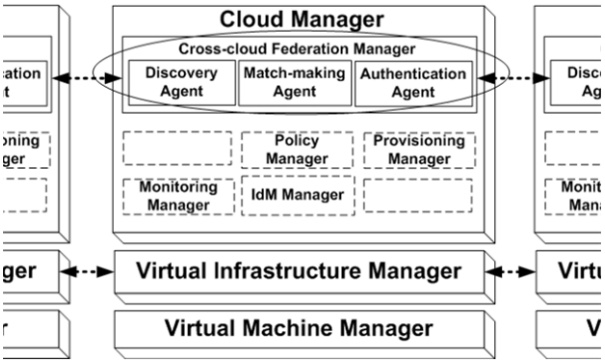

Figure 2. CCFM for the management of the cross-cloud federation inside the general three-layers cloud architecture.

a new component within the Cloud Manager layer (depicted in the top part of Figure 2), named *Cross-Cloud Federation Manager* (CCFM). The CCFM has been conceived for enabling each cloud to perform all the operations needed to pursue the target of the federation establishment.

The cross-cloud scenario we are considering can be seen as an highly dynamic environment: new clouds, offering different available resources and different authentication mechanisms could appear, while others could disappear. Taking into account such dynamism, when a home cloud needs to "lease" external resources from a foreign cloud, the first phase it will perform refers to the *discovery* of the foreign cloud which properly *matches* (phase 2) its requirements (both in terms of available resources, policies, supported authentication mechanisms, etc). Once these two phases have been performed, and the best foreign cloud has been found, in order to establish a secure interaction between the home cloud and the selected foreign cloud, an *authentication* (phase 3) process will begin.

The accomplishment of the authentication process leads to the establishment of a secure and direct connection between the home cloud and foreign cloud VI Managers (see the bottom part of Figure 2) whereby the resource provisioning can be accomplished. As consequence, the home cloud will be able to instantiate (or migrate) Virtual Resources (VMs) on the Foreign Cloud in a secure environment. The concept of migration can be seen as the opportunity to move the Virtual Machines not only in intra-site domain but also to transfer them on federated inter-site domains. In this case the migration might occur across subnets, among hosts that do not share storage and across administrative boundaries.

C. The Three CCFM Agents

The CCFM module represents the main "actor" in our three-phase federation model: accomplishing the tasks of *discovery*, *match-making* and *authentication*, it focuses on the federation issues and tries to solve them, satisfying the previously stated requirements needed in this kind of environment: automatism, scalability, interoperability and

SSO. In our design, the CCFM consists of three different subcomponents (agents), each addressing a different phase of the federation model, as reported below:

1) The Discovery Agent: It manages the discovery process among all the available clouds within the dynamic environment. Since its state is pretty flexible and dynamic, the discovery process has to be implemented in a totally distributed fashion: all the discovery agents must communicate exploiting a peer-to-peer (p2p) approach, based conveniently on the *presence* concept. This latter is indeed an enabling technology for p2p interactions, whose implementation follows the software design pattern known as publish-and-subscribe (pub-sub): an application publishes its own set of information to a centralized location (even though such location is logically centralized, it is implemented in a distributed fashion, also granting fault-tolerance mechanisms), from which only a set of authorized (subscribed) entities are able to retrieve it.

The above mentioned approach, based on the "presence" concept, could represent a valid starting point to address the foreign cloud discovery problem in a cross-cloud federated scenario. On behalf of the cloud on which belongs, each Discovery Agent, according to the pub-sub software pattern, broadcasts on-demand information about its supported features and resources state to other peers aiming to attend the the cross-cloud. When a home cloud needs to know the set of foreign clouds belonging to the federation in a given moment, along with the related capabilities, by means of its Discovery Agent, it will merely check the shared area on which such information is stored.

To accomplish such mechanisms, each Discovery Agent has to act as a *presence daemon* which exploits one or more presence protocols, in order to distribute presence information as widely as possible. Such protocols will be employed integrating functionalities which regard service discovery and management of the capabilities information, for allowing clouds to gain knowledge about the resources made available from the other entities of the federation, providing real-time mechanisms for additional use of presence-enabled systems.

2) The Match-Making Agent: It accomplishes the task of choosing the more convenient foreign cloud(s) wherewith to establish a cross-cloud federation. In order to accomplish such task, the agent is responsible to manage and enforce the policies defined by the clouds. In fact both the home cloud and the available (discovered) foreign clouds are associated to meta-data containing a set of policies, each one composed of a set of rules describing respectively which resources are required and which ones are offered according to some conditions. For example a home cloud could require a certain amount of CPU and storage, with a certain QoS, supporting a target IdP for the authentication (described in the following), from the 8.00 pm to the 11.00 pm. Instead a foreign cloud could offer resources with a certain QoS, with a target authentication mechanism, at any time, denying the requests sent by certain home clouds. In order to enabling the agent to evaluate among all the available (discovered) clouds, the ones that best "fit" the requirements of its home cloud, the match-making agent should perform a policy matching.

To accomplish such mechanism, each match-making agent needs to evaluating the applicable policies, returning authorization decisions, performing an access control by enforcing the stated authorization decisions, and converting the foreign cloud policies from the native format to its own supported policies format. More specifically the cloud policies should be expressed by means of an extensible policy language ables to integrate different policy format using transformation algorithms.

3) The Authentication Agent: It, cooperating with third parties trusted entities, is responsible to create a security context between home and foreign clouds. When the authentication phase begins, the home cloud authentication agent contacts its "peer" on the foreign cloud: the authentication process between such agents (and thus the clouds) will be lead exchanging authentication information in form of meta-data, involving trusted third parties in the process. In a distributed scenario composed of hundreds of clouds, the credential managements could be very hard: each home cloud should manage hundreds of accounts which can changes over the time, each one needed for the authentication with a certain foreign cloud. In addition each cloud could support different authentication mechanisms from each other. Such concerns raises an issue we named cloud SSO authentication problem: a cloud should be able to authenticate itself with other heterogeneous clouds regardless their security mechanisms, performing the log-in once using unique credential over the time, gaining the access to all the required resources.

In order to accomplish such mechanisms, the authentication agent should use the Identity Provider/Service Provider (IdP/SP) model. Such model defines the exchange of authentication assertions between security domains, more specifically, between an IdP or asserting party (a producer of assertions) and a SP or relying party (a consumer of assertions). The model assumes, that a subject, a person or a software/hardware holds at least one digital identity on an IdP which provides SSO authentication services. SSO is the property by means of an entity is able to perform the authentication once accessing to the resources of different SPs. Since SPs are trusted with the IdP, the subject is able to perform the log-in once gaining the access to all the required resources (hosted inside the SPs).

Considering the cross-cloud federation depicted in Figure 3, both the home cloud and the foreign cloud, by means of their own Authentication Agents, represent respectively the subject and the relying party, whereas the IdP acts as the third party asserting to a foreign cloud the trustiness of the home cloud identity.

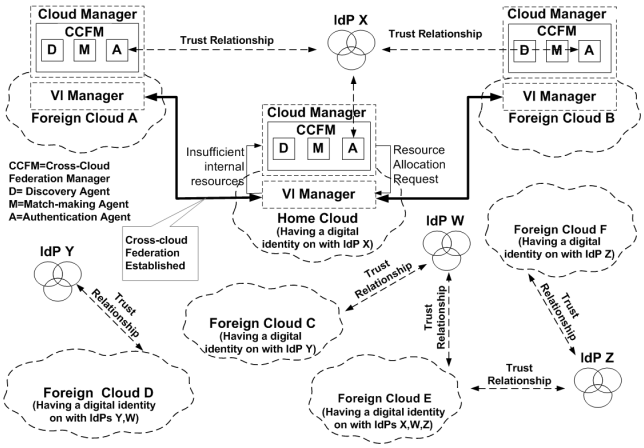

Figure 3. Entities involved in the cross-cloud federation establishment.

V. The Three-Phase Cross-Cloud Federation Process: a design overview

In this Section, considering the cross-cloud federation scenario depicted in Figure 3 including several clouds whose identities are distributed inside many trusted IdPs, we provide an overview of the technologies which could be employed to design the three agents laying in the cloud manager layer.

More specifically, among all the technologies available on the market, we picked out XMPP, XACML and SAML respectively for the design of the discovery agent, the match-making agent and the authentication agent. When the home cloud manager layer receives a request for services from its clients, it sends a resource allocation request (i.e. virtual machines) to the underlying VI manager layer. If this latter, evaluating its instantaneous workload, realizes it has not enough resources to satisfy the client's request, it notifies such situation to the cloud manager which, by means of its CCFM, starts the cross-cloud federation process.

A. The Discovery Phase

Once the home cloud manager decides to ask the foreign clouds for resources, a request is forwarded to the CCFM which, by means of its discovery agent, begins the *discovery* process to obtain a list of all the available foreign clouds. In Section IV we provided a brief overview on the mechanisms related to the discovery phase: all the Discovery Agents must communicate exploiting a peer-to-peer (p2p) approach, based conveniently on the *presence* concept. On behalf of the cloud on which belongs, each Discovery Agent, according to the pub-sub software pattern, broadcasts information about its supported features and resources state to the other peers which subscribed the cross-cloud.

The accomplishment of such mechanisms, in our opinion, could be based on the employment of "open technologies", in order to allow an high degree of interoperability and flexibility among the entities. As already stated, in such context the concept of *presence* plays the lead: it first emerged as an aspect of communication systems, especially Instant Messaging (IM), allowing users to see the availability of their friends. Today, the huge role that IM has had in establishing presence is evident with the available protocols, such as Instant Messaging and Presence Service (IMPS) [12], Session Initiation Protocol (SIP) for Instant Messaging and Presence Leveraging Extensions (SIMPLE) [13] and the Extensible Messaging and Presence Protocol (XMPP) [14]. This latter, in our opinion, represents the best solution to the service discovery problem in a federated infrastructure: although its standard features refers to provide basic information about the entities availability, thanks to its extensible paradigm, XMPP can also include further abilities which match our scenario requirements.

The Discovery Agent of the home cloud, in order to gain knowledge about all the available foreign clouds, is able to retrieve other entities' availability information merely querying the "shared location" on which it is published. More specifically, in a XMPP based scenario, the "location" is identified with a specific *XMPP room* where the clouds aiming to take part the federation hold a subscription. The method employed from a home cloud to obtain such information is based on a particular XMPP info-query message, exchanged in the form of XML "stanzas" between the home cloud and the foreign clouds, which is marked from the XML tags *<iq></iq>*.

In order to discover information about the foreign clouds' features and their respective resource state, the requesting home cloud must send an IQ stanza of type "get", containing an empty *<query/>* element qualified by the *http://jabber.org/protocol/disco#info* name-space, to all the foreign clouds which subscribed to the federation room. In the following, in order to show how the discovery process should be performed using XMPP, we setup a server and tested a message exchange between the home cloud "homecloud.org" discovering the foreign cloud "foreign-cloudA.net".

```
<iq type='get'
    from='homecloud.org'
    to='foreigncloudA.net'
    id='2g46s'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Each target entity (foreign cloud) then must either return to the home cloud an IQ stanza of type "result", or return an error. The result will contain a *<query/>* element qualified by the *http://jabber.org/protocol/disco#info* name-space, which in turn contains one or more *<identity/>* elements describing static information (i.e. who the cloud asserts to be, which services it provides, etc.) and one or more *<feature/>* elements describing mutable information over the time (i.e. the offered computing capabilities, the time based resource availability, the offered QoS, the trusted IdPs,

a cloud black-list, etc.)

```
<iq type='result'
    from='foreigncloudA.net'
    to='homecloud.org'
    id='2g46s'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <identity
        category='cloud'
        type='cross-cloud-federation-enabled'
        name='foreign-cloud-A'/>
    <identity
        category='cloud'
        type='european'
        name='foreign-cloud-A'/>
    <feature var='http://foreigncloudA.net/amount/cpu'/>
    <feature var='http://foreigncloudA.net/amount/storage
        '/>
    <feature var='http://foreigncloudA.net/amount/memory
        '/>
    <feature var='http://foreigncloudA.net/availability/
        time'/>
    <feature var='http://foreigncloudA.net/QoS'/>
    <feature var='http://foreigncloudA.net/authentication/
        IdP'/>
    <feature var='http://foreigncloudA.net/cloud-black-
        list'/>
  </query>
</iq>
```

As well as for the foreign cloud "foreigncloudA.net", the home cloud discovery agent performs such task process for all the other members of the federation room.

B. Match-making Phase

Both static and mutable information related to the discovered clouds could be furthermore classified in quantifiable and unquantifiable. For instance, the amount of available CPU, RAM and QoS are expressed as a numeric value whereas the trusted IdPs and the cloud black-list are expressed in a string form.

In order to accomplish the match-making, the above mentioned information related to each discovered foreign cloud, is stored within a cache. Subsequently, a specific module named *Context Handler*, maps each entry of the cache from the native XML format (extracted from the XML stanza) to the *XACML Policy* format, containing both quantifiable and unquantifiable information. Each policy relative to a given discovered foreign cloud, is compared with the home cloud's policies, to establish if that foreign cloud fits the home cloud's requirements.

Two policies examples are reported in the following:
Home Cloud Policy*: I accept CPU X_1, RAM Y_1, with QoS Z_1 from any foreign cloud trusted with the IdP T.*
Foreign Cloud Policy*: I provide CPU X_2, RAM Y_2, with QoS Z_2 to all clouds except cloud A and I'm trusted with the IdPs T, H.*
The match-making process consists of two different steps: a first one is performed analysing both unquantifiable and quantifiable information belonging to a certain range; the second one, considering the set of foreign clouds filtered in the step 1, carries out a numerical analysis on the quantifiable parameters to select the best foreign cloud. In the step 1, depicted in top part of Figure 4, each policy generated

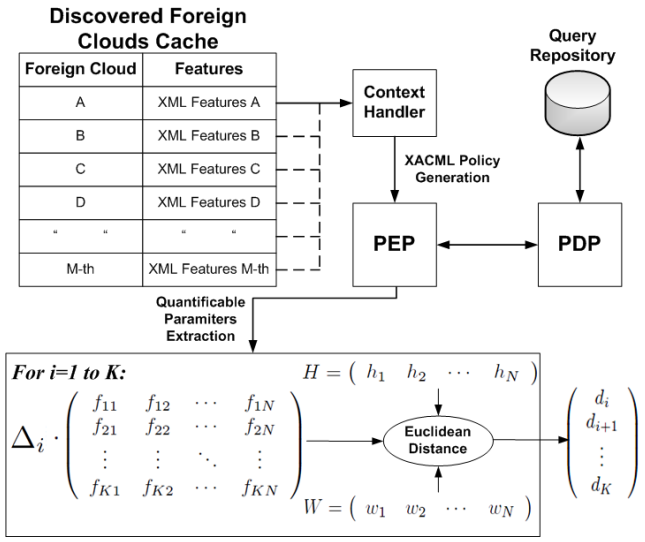

Figure 4. Sequence of operations accomplished during the match-making phase.

from the context handler is caught from the XACML Policy Enforcement Point (PEP) and forwarded to the XACML Policy Decision Point. This latter, using XACML combination algorithms, performs a policy matching between the foreign cloud policy and the home cloud policies stored inside its query repository. The combination algorithm verifies if the policy of a given foreign cloud matches the home cloud's policies considering both the unquantifiable information (e.g. the trusted IdP, the blacklisted clouds) and the quantifiable information belonging to a certain range (e.g. QoS metric greater than a threshold, resource availability from 10.00 AM to 5.00 PM, etc).

After the pre-filtering has been performed using XACML, in the step 2, for each foreign cloud whose policy matches the home cloud policies, an entry including its quantifiable parameters (i.e. CPU and RAM amount, offered QoS, etc.) is created inside a KxN matrix $F = (f_{ij})$, as represented in the bottom part of Figure 4. More specifically, the i-th row of matrix F could be considered as a vector P^{i-th}, whose N components are the quantifiable parameters of the i-th foreign cloud. Introducing $\Delta_i = [\delta_{i1}\ \delta_{i2}\ \ldots\ \delta_{iK}]$, we extract $P^{i-th} = (p_j) = [p_1\ p_2\ \ldots\ p_N]$, the row vector associated to the i-th F matrix's row, computing $P^{i-th} = \Delta_i \cdot F$. The Δ_i elements are defined according to the Kronecker delta. Considering the vector P^{i-th}, associated to the first row of the matrix F, $P^{1-st} = \Delta_1 \cdot F$, where $\Delta_1 = [\delta_{11}\ \delta_{12}\ \ldots\ \delta_{1K}] = [1\ 0\ \ldots\ 0]$.

As well as the foreign clouds, also the home cloud's quantifiable parameters represent a N-dimensional vector. In order to pick out which foreign clouds fit the home cloud requirements, a comparison method, based on the weighted

euclidean distance is accomplished according to the formula

$$d_{P^{i-th}H} = d_i = \sqrt{\sum_{j=1}^{N} w_j (p_j^{i-th} - h_j)^2},\ 1 \leq i \leq K.$$

P^{i-th} represents the vector of N quantifiable parameters of the foreign cloud i (the i-th row of matrix F), H the vector of N quantifiable parameters of the home cloud and W, the vector whose N elements w_j represent the weight associated to the j-th term of the euclidean distance. Computing the aforementioned formula iteratively on the K P^{i-th} vectors extracted from matrix F, the vector D containing K value of distance d_i is obtained. Each d_i provides an estimation of how the home cloud requests are satisfied from the i-th foreign cloud associated to the vector P^{i-th}. Therefore, sorting in ascending order the vector D, the home cloud obtains the sequence of foreign clouds (from the best to the worst) with whom a federation can be established.

We remark, the algorithm presented in this phase allows to follow the dynamic evolution of the home cloud. In particular, it enables home clouds to drive their economies of scale and their businesses. Both vectors H and W are time dependent: [$H(t)$ and $W(t)$]; in our view, the n parameters of H and W vectors can change time by time according to the home cloud business models. In fact there is a relevant binding between the Service Level Agreement (SLA) and the vector W. When home cloud stipulates whatever SLA with its customers, automatically this agreement affects the weight of the W parameters (w_i). Home cloud can decide whether to satisfy a customer's demand, economically more convenient than another one, acting on W parameters, modifying them and searching the most suitable foreign cloud(s).

C. Authentication Phase

In the last phase, in order to establish a federation with the foreign cloud previously chosen, a cloud SSO *authentication* process has to be started by the home cloud. Such process will involve: the authentication agent of the home cloud, the corresponding peer of the foreign cloud and the IdP trusted with the foreign cloud, on which the home cloud has a digital identity. Once the home cloud and the foreign cloud authentication agents have established a trust context, their respective underlying VI manager layers setup a low-level trust context allowing the cross-cloud resource provisioning. Therefore, the home cloud VI manager will be able to instantiate virtual resources on the foreign cloud VI manager.

The employment of the IdPs presents some advantages well fitting our cross-cloud federation scenario: even though each cloud has its internal security mechanisms, whatever the *foreign cloud* is, regardless of its authentication mechanisms, by means of IdPs a *home cloud* will be able to authenticate itself with other foreign clouds already having a trust relationship, exploiting the well-known concept of SSO. The resource provisioning in cross-cloud federation may be solved establishing trust relationships between the clouds using several IdPs containing the credentials of the cloud asking for resources.

To address the cloud SSO authentication problem many technologies could be employed such as Shibbolet [15], OpenID [16], and Security Assertion Markup Language (SAML) [17]. This latter, in our opinion, is the more flexible one because allows to create specific profile to address different authentication scenarios. To show how the authentication phase could be performed, we developed a new SAML cloud federation profile. Such profile was designed to enable a home cloud to perform SSO authentication on several foreign clouds both having a trusted relationship with the home cloud's IdP and regardless their security mechanisms.

In our profile, both the home cloud and the foreign cloud, by means of their own Authentication Agents, represent respectively the subject and the relying party, whereas the IdP acts as the third party asserting to a foreign cloud the trustiness of the home cloud identity. The profile has been designed as a combination of the following SAML elements: an assertion including an authentication statement, a request-response protocol, and a SAML SOAP Binding.

In the following is presented an example of XML document issued by foreign cloud to the home cloud. More specifically, such document contains an authentication request used by the home cloud to perform the SSO authentication on the IdP trusted with the foreign cloud.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/
    envelope/">
  <S:Body>
    <ns2:AA-ForeignCloud-A-ResReqResponse xmlns:ns2="
        http://webservices/">
      <return>
        <samlp:AuthnRequest xmlns:samlp="urn:oasis:names
            :tc:SAML:2.0:protocol" xmlns:saml="urn:
            oasis:names:tc:SAML:2.0:assertion" ID="dfa6
            " Version="2.0" IssueInstant="2010-01-12T18
            :34:42Z" AssertionConsumerServiceIndex="0">
          <saml:Issuer>https://cloudA.net/SAML2</saml:
              Issuer>
          <samlp:NameIDPolicy
           AllowCreate="true"
           Format="urn:oasis:names:tc:SAML:2.0:nameid-
               format:transient"/>
        </samlp:AuthnRequest>
      </return>
    </ns2:AA-ForeignCloud-A-ResReqResponse>
  </S:Body>
</S:Envelope>
```

Such authentication process is fundamental for the subsequent establishment of a secure channel between the home cloud VI manager and one or more foreign cloud VI manager(s). Once the secure channel has been established the home cloud is able to gain the access to the required resources offered by the foreign cloud.

VI. Conclusions and Future Works

In this paper we tackled the cross-cloud federation problem performing an in depth analysis and proposing a three-

phase (discovery, match-making, and authentication) model. An architectural solution including the Cross-Cloud Federation Manager (CCFM) has been designed and described. Furthermore, an overview of the possible implementation practice of three agents has been presented, pointing out the technologies on which such implementation could rely. In future, we plan to study the performances of such cross-cloud federation scenario, evaluating the components behavior, either employing real testbeds or by means of a simulated environment, including hundreds of clouds dynamically joining and leaving federations.

ACKNOWLEDGEMENTS

The research leading to the results presented in this paper has received funding from the European Union's seventh framework programme (FP7 2007-2013) Project RESERVOIR under grant agreeement number 215605.

REFERENCES

[1] T. Bittman, "The evolution of the cloud computing market," *Gartner Blog Network, http://blogs.gartner.com/thomas_bittman/2008/11/03/the-evolution-of-the-cloud-computing-market/*, November 2008.

[2] Sun Microsystems, Take your business to a Higher Level - Sun cloud computing technology scales your infrastructure to take advantage of new business opportunities, guide, April 2009.

[3] W. Li and L. Ping, "Trust model to enhance security and interoperability of cloud environment," in *Cloud Computing*, pp. 69–79, November 2009.

[4] Amazon Elastic Compute Cloud (Amazon EC2): http://aws.amazon.com/ec2/.

[5] Windows Azure Platform, http://www.microsoft.com/windowsazure/.

[6] OpenQRM, "the next generation, open-source Data-center management platform", http://www.openqrm.com/.

[7] B. Sotomayor, R. Montero, I. Llorente, and I. Foster, "Resource Leasing and the Art of Suspending Virtual Machines," in *High Performance Computing and Communications, 2009. HPCC '09. 11th IEEE International Conference on*, pp. 59–68, June 2009.

[8] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good, "On the Use of Cloud Computing for Scientific Workflows," in *SWBES 2008, Indianapolis*, December 2008.

[9] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus Open-Source Cloud-Computing System," in *Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on*, pp. 124–131, May 2009.

[10] Nagios, The Industry Standard in IT Infrastructure Monitoring: http://www.nagios.org/.

[11] B. Sotomayor, R. Montero, I. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," *Internet Computing, IEEE*, vol. 13, pp. 14–22, September 2009.

[12] Instant Messaging and Presence Service (IMPS), http://www.ietf.org/rfc/rfc2778.txt.

[13] Session Initiation Protocol (SIP) for Instant Messaging and Presence Leveraging Extensions (SIMPLE), http://www.ietf.org/dyn/wg/charter/simple-charter.html.

[14] Extensible Messaging and Presence Protocol (XMPP), http://xmpp.org/.

[15] The Shibboleth system standards, http://www.openqrm.com.

[16] OpenID Authentication 2.0, OpenID Foundation, http://openid.net/specs/openid-attribute-exchange-2_0.html, 2007.

[17] SAML V2.0 Technical Overview, OASIS, http://www.oasis-open.org/specs/index.php#saml.