Simple Storage Replication Protocol (SSRP) for Intercloud

David Bernstein
Huawei Technologies, Ltd.
North America R&D Division
Santa Clara, California, USA
email: dbernstein@huawei.com

Deepak Vij
Huawei Technologies, Ltd.
North America R&D Division
Santa Clara, California, USA
email: dvij@huawei.com

Abstract – **Working groups have proposed building a layered set of protocols to solve Cloud Computing interoperability challenges. This is not the problem of application portability which calls for standardized programmer interfaces. This is the problem of generalized, transparent, back-end cloud-to-cloud federation. Current Intercloud designs do not envision each cloud provider establishing connectivity with another cloud provider in a Point-to-Point manner, as this will result into the n^2 complexity problem. Instead, Intercloud Directories, Exchanges, and Gateways will help facilitate as mediators for enabling connectivity and collaboration among disparate cloud providers in a manner analogous to the Internet itself. These Intercloud elements would implement a profile of protocols becoming known as Intercloud protocols. Researchers and working groups have proposed a layered set of such protocols using Extensible Messaging and Presence Protocol for transport, and Semantic Web with Resource Description Framework for resource matching. This paper builds on that work and discusses the next layer up, where a specific use case of federation is explored. We find that the back-end implementation of cloud-distributed, unstructured storage can be extended to generalized cloud-to-cloud federation using a Simple Storage Replication Protocol (SSRP), which is built on top of the base Intercloud protocol set. This paper details SSRP.**

Keywords-component: Intercloud; Cloud Computing; Cloud Storage; Cloud Federation; XMPP; RDF; SSRP.

I. INTRODUCTION

Cloud Computing has a well accepted terminology [1], and Use Cases and Scenarios for Cloud IaaS and PaaS interoperability [2][3] have been detailed in the literature along with the challenges around actually implementing standards-based Intercloud federation and hybrid clouds. Work detailing high level architectures for Intercloud interoperability have been published [4][5].

Specific implementation approaches for Intercloud protocols [6][7] have been proposed, including specifically Extensible Messaging and Presence Protocol (XMPP) [8][9] for transport, and using Semantic Web [10] techniques such as Resource Description Framework (RDF) [11] to specify resources.

Detailed approaches outlining the use of these technologies to implement the base Intercloud protocols; have been published first on the feasibility of XMPP as a control plane operations for Intercloud [12], and next how Cloud Computing resources can be described, cataloged, and mediated using Semantic Web Ontologies, implemented using RDF techniques [13]. The base topology and the base transport and resource framework provide a foundation for a specific use case of Intercloud federation, which is the subject of this work.

Here, we outline the problem of federating cloud storage services. We look at the simplest of cloud storage, which is distributed, unstructured storage, best exemplified by the Amazon S3 [14] commercial offering. Cloud Storage challenges for structured data such as relation databases are equally important but of a slightly different scope and dimensions, and not addressed in this paper. We will cover the cloud storage challenges and proposed solutions for structured data storage at a later time.

Our work has detailed a specific use case of distributed unstructured cloud storage federation. We find that the back-end implementation of cloud-distributed, unstructured storage can be extended to generalized cloud-to-cloud federation using a Simple Storage Replication Protocol (SSRP), which is built on top of the previously referenced base Intercloud protocol set. This paper details SSRP.

The rest of the paper is organized as follows: Section II outlines the proposed overall "Intercloud Topology". Section III briefly describes an overview of cloud storage, in general. Section IV specifically describes an overview of unstructured cloud storage. Section V briefly describes typical storage interoperability challenges across various cloud providers. Section VI outlines an overview of Intercloud topology. Section VII briefly delves into brief overview of major constituents of proposed Intercloud topology. Section VIII briefly describes Ontology based cloud resources catalog. Section IX briefly describes cloud resources Ontology itself. Section X describes an overview of XMPP based Intercloud negotiation and services framework proposed as part of the Intercloud standards and protocols. Section XI outlines the sequencing of Intercloud protocols for federated unstructured storage use case. Next Section XII delves deep into proposed Intercloud Enabled Federated unstructured storage system architecture. Section XIII describes an actual use case for Intercloud enabled federated unstructured cloud storage. And finally, Section XIV presents our conclusions.

 ISBN: 978-1-61208-103-8

II. REVIEW OF INTERCLOUD TOPOLOGY

Cloud instances must be able to dialog with each other. One cloud must be able to find one or more other clouds, which for a particular interoperability scenario is ready, willing, and able to accept an interoperability transaction with and furthermore, exchanging whatever subscription or usage related information which might have been needed as a pre-cursor to the transaction.

Thus, an Intercloud Protocol for presence and messaging needs to exist which can support the 1-to-1, 1-to-many, and many-to-many Cloud to Cloud use cases. The vision and topology for the Intercloud we will refer to is as follows. At the highest level, the analogy is with the Internet itself: in a world of TCP/IP and the WWW, data is ubiquitous and interoperable in a network of networks known as the "Internet".

In a world of Cloud Computing, content, storage and computing is ubiquitous and interoperable in a network of Clouds known as the "Intercloud"; this is illustrated in Figure 1.

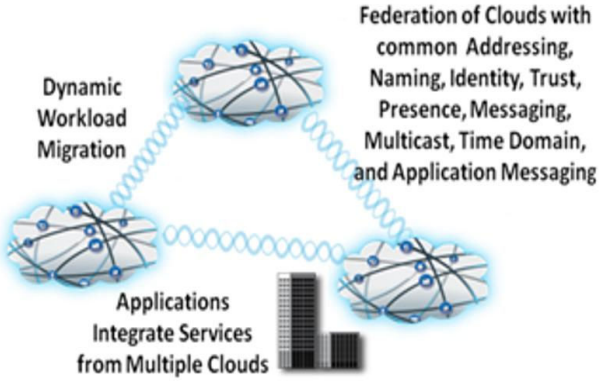

Figure 1. The Intercloud Vision

The reference topology for realizing this vision is modeled after the public Internet infrastructure. Again, we use the generally accepted terminology.

There are Public Clouds, which are analogous to ISP's and Service Providers offering routed IP in the Internet world.

There are Private Clouds which are simply a Cloud which an organization builds to serve itself.

There are Intercloud Exchanges (analogous to Internet Exchanges and Peering Points) where clouds can interoperate.

Finally, there is an Intercloud Root, containing services such as Naming Authority, Trust Authority, Directory Services, and other "root" capabilities. It is envisioned that the Intercloud root is of course physically not a single entity, a global replicating and hierarchical system similar to DNS [15] would be utilized.

All elements in the Intercloud topology contain some gateway capability analogous to an Internet Router, implementing Intercloud protocols in order to participate in Intercloud interoperability. We call these Intercloud Gateways. The entire topology is detailed in Figure 2.

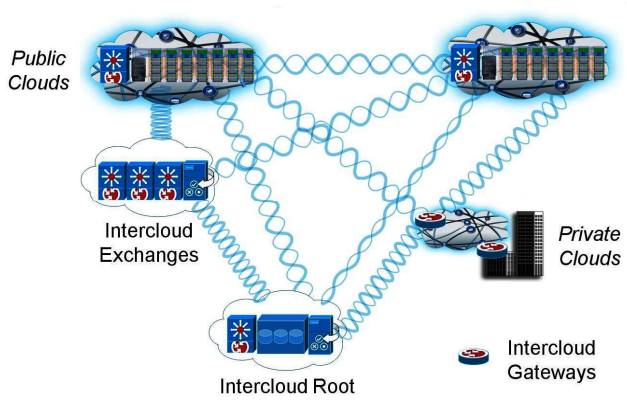

Figure 2. Reference Intercloud Topology

The Intercloud Gateways would provide mechanism for supporting the entire profile of Intercloud protocols and standards. The Intercloud Root and Intercloud Exchanges would facilitate and mediate the initial Intercloud negotiating process among Clouds. Once the initial negotiating process is completed, each of these Cloud instance would collaborate directly with each other via a protocol and transport appropriate for the interoperability action at hand; for example, a reliable protocol might be needed for transaction integrity, or a high speed streaming protocol might be needed optimized for data movement over a particular link.

III. OVERVIEW OF CLOUD STORAGE

At a very high level, cloud storage solves the following issues faced by IT organizations:

- Dynamic capacity – storage capacity is fixed once purchased/leased. Cloud storage provides an almost infinite amount of storage for data. One pays for this storage, in GB or TB per month increments, with added storage services (multi-site replication, high availability, etc.) at extra charge. Such capacity can be reduced or expanded at a moments notice.
- Offsite DR – disaster recovery for many small shops is often non-existent or rudimentary at best. Using cloud storage, data can be copied to the cloud and accessed anywhere via the internet. Such data copies can easily support rudimentary DR for a primary data center outage.
- Access anywhere – storage is typically local to the IT shop and can normally only be accessed at that location. Cloud storage can be accessed from any internet access point. Applications that are designed to operate all over the world can easily take advantage of such storage.
- Data replication – storage data is replicated for high availability. Cloud storage providers replicate

 ISBN: 978-1-61208-103-8

storage data to multiple sites so that if one site goes down other sites can still provide service transparently to the consumer.

Typically, users are provided with a very simplistic set of APIs (RESTful Web Services call) in order to be able to PUT and GET data into a big cloud of storage with the following high-level characteristics:

- End users Customers need not be aware about exact physical location of the data
- End users can advise the system of their physical location requirements for data (limiting it to a particular country or region) and the Cloud may be able to support this requirement
- The storage needs to be transparently reliable (and replicated) as far as the user is concerned
- Users access the storage over the public Internet through a Storage API

Functionally, the basic intent of the Storage API is to provide a static URI storage repository for files which will be referenced as a whole. Often, this is called "BLOB" storage, meaning "Binary Large Object", as it is not providing any structure like a file system or a database. The most successful commercial implementation of such a BLOB storage service is Amazon S3.

In contrast to Amazon S3, some cloud storage solutions provide various other access points, in addition to HTTP based web services APIs (SOAP or REST based). Examples of these access points may include: NFS like, FTP like, WebDAV like, CIFS like, iSCSI like, BitTorrent like, etc. Essentially, these access points are a veneer or an abstraction adapter layer on top of underlying BLOB storage access mechanism (web services in the case of Amazon S3). Additionally, there are cloud storage solutions which provide underlying structures beyond BLOB (such as Block Storage, Queue Storage etc.). These storage structures, in turn, may provide all of the above access points on top of the underlying access mechanism for the respective storage structure supported by the cloud storage provider.

This paper does not study the interoperability challenges of these examples. We look specifically at the problem of interoperability across cloud providers of unstructured (BLOB) storage.

IV. UNDERSTANDING UNSTRUCTURED CLOUD STORAGE MECHANISMS

For cloud storage providers such as Amazon S3, users do not get direct access to the cloud storage. Instead, they need to invoke HTTP based web services call in order to perform data storage functions such as store, remove or retrieve data etc. However, there are various third party tools such as "JungleDisk" which provides WebDAV based file mount so that the Amazon S3 cloud storage looks like a local device to the end users (*storage-as-storage)*.

As mentioned earlier that enabling cloud storage typically does not require any manual intervention and is essentially self-services. While provisioning storage, users may be asked to choose preferences for geographic regions in which primary copy of their content may be stored. The preference may be one or more regions. If more than one region is selected, clients may optionally choose a geographic location while uploading (PUT) the content.

If no geographic preference is provided during PUT, the system will choose from one of the preferences. If no preference is selected, the system can pick the region on its own discretion. It is typically recommended that clients choose a few regions for optimal performance and Business Continuity Planning. Essentially, the storage provisioning process is "a-la-carte menu" approach to provisioning.

V. STORAGE INTEROPERABILITY CHALLENGES ACROSS CLOUD PROVIDERS

Let us consider an interoperability use case scenario. A user is performing a function that utilizes Cloud based storage capabilities. In Cloud Computing, storage is not like disk access, there are several parameters around the storage which are inherent to the system, and one decides if they meet certain needs or not. For example, storage is typically replicated to several places in the cloud, In AWS [16] and in Azure [17] it is replicated three places. The storage API is such that, a *write* will return as successful when one replicate of the storage has been affected, and then a "lazy" internal algorithm is used to replicate the storage object to two additional places. If one or two of the storage object replicates are lost the cloud platform will replicate it to another place or two such that it is now in three places.

A user has some control over where the storage is, physically, for example, one can restrict the storage to replicate entirely in North America or in Europe. There is no ability to vary from these parameters; that is what the storage system provides.

We do envision other providers implementations might say, five replicates, or a deterministic replication algorithm, or a replicated (DR) *write* which doesn't return until and unless *n* replicates are persisted. One can create a large number of variations around "quality of storage" for Cloud.

In the interoperability scenario, suppose AWS is running short of storage, or wants to provide a geographic storage location for an AWS customer, where AWS does not have a datacenter, it would be sub-contracting the storage to another cloud service provider. In either of these scenarios, AWS would need to find another cloud, which was ready, willing, and able to accept a storage subcontracting transaction with them. AWS would have to be able to have a reliable conversation with that cloud, again exchanging whatever subscription or usage related information which might have been needed as a pre-cursor to the transaction, and finally have a reliable transport on which to move the storage itself.

 ISBN: 978-1-61208-103-8

Note, the S3 storage API is not guaranteed to succeed, if there is a failed "*write"* operation from AWS to a subscriber request, the subscriber code is supposed to deal with that (perhaps, via an application code level retry). However Cloud to Cloud, a target cloud "*write"* failing is not something the subscriber code can take care of. That needs to be reliable.

Currently, due to the proprietary nature of each cloud, every cloud environment is a silo in itself. There are no formal protocols and standards established in order to address the above mentioned issues. The intent of our work is to address storage interoperability issues such as naming, discovery, conversation setup items challenges etc. End goal is to provide a common set of standards, protocols and new components (such as cloud exchange providers role as intermediaries) to address all these interoperability issues in a seamless manner for enabling "Federated Cloud Storage" environment.

VI. OVERVIEW OF INTERCLOUD SYSTEM ARCHITECTURE

As shown in the Figure 3, we envision storage in the Intercloud environment to be federated among disparate and heterogeneous cloud environments. "Intercloud Exchanges" would be a key component for enabling the seamless federated storage environment. These exchanges would facilitate and mediate initial negotiating process among clouds.

Once the initial negotiating process is completed, each of these cloud environments, in turn, would collaborate with each other via "Intercloud Gateways". Intercloud Gateways would provide mechanism for authentication, support for various storage replication and storage access protocols and standards, presence/collaboration, and common "Cloud Ontology" set among heterogonous cloud environments.

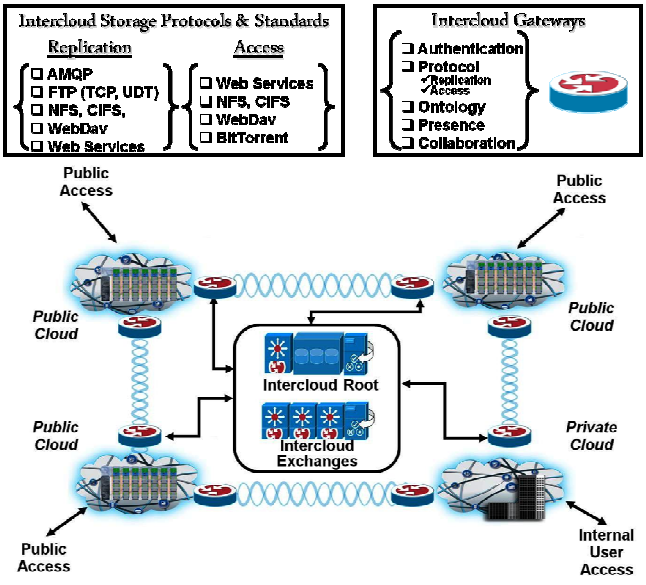

Figure 3. Intercloud enabled Federated Storage Architecture

As shown in the schematic above that cloud-to-cloud storage replication process might leverage application level protocols such as FTP or messaging protocol such as AMQP [18]. These cloud-to-cloud application protocols could be delivered over underlying transport protocol such as UDT (UDP based Data Transfer) [19] instead of traditional TCP protocol. This is mainly due to the fact that TCP protocol becomes very inefficient in a high bandwidth environment. On the other hand, public access to the federated cloud storage may still be delivered via Web Services APIs (RESTful or SOAP based) etc.

VII. INTERCLOUD ROOT, EXCHANGES AND GATEWAYS

As mentioned earlier the various providers will emerge in the enablement of the Intercloud. We first envision a community governed set of Intercloud Root providers who will act as brokers and host the Cloud Computing Resource Catalogs for the Intercloud computing resources. They would be governed in a similar way in which DNS, Top Level Domains [20] or Certificate Authorities [21] are, by an organization such as ISOC [22] or ICANN [23]. They would also be responsible for mediating the trust based federated security among disparate clouds by acting as Security Trust Service providers using standards such as SASL [24] and SAML [25].

The Intercloud Root instances will work with Intercloud Exchanges to solve the n^2 problem by facilitating as mediators for enabling connectivity among disparate cloud environments. This is a much preferred alternative to each cloud vendor establishing connectivity and collaboration among themselves (point-to-point), which would not scale physically or in a business sense.

As we mentioned earlier that all elements in the Intercloud topology contain some gateway capability analogous to an Internet Router, implementing Intercloud protocols in order to participate in Intercloud interoperability. We call these Intercloud Gateways.

Intercloud Gateways would provide mechanism for supporting the entire profile of Intercloud protocols and standards. The Intercloud Root and Intercloud Exchanges would facilitate and mediate the initial Intercloud negotiating process among Clouds.

Once the initial negotiating process is completed, each of these Cloud instance would collaborate directly with each other via a protocol and transport appropriate for the interoperability action at hand.

VIII. ONTOLOGY BASED RESOURCES CATALOG

In order for the Intercloud capable Cloud instances to federate or otherwise interoperate resources, a Cloud Computing Resources Catalog system is necessary infrastructure. This catalog is the holistic and abstracted view of the computing resources across disparate cloud environments. Individual clouds will, in turn, will utilize this catalog in order to identify matching cloud resources by applying certain Preferences and Constraints to the

 ISBN: 978-1-61208-103-8

resources in the computing resources catalog. The technologies for this use the Semantic Web which provides for a way to add "meaning and relatedness" to objects on the Web. To accomplish this, one defines a system for normalizing meaning across terminology, or Properties. This normalization is called Ontology.

Comprehensive semantic descriptions of services are essential to exploit them in their full potential. That is discovering them dynamically, and enabling automated service negotiation, composition and monitoring. The semantic mechanisms currently available in service registries such as UDDI [26] are based on taxonomies called "tModel" [27]. tModel fails to provide the means to achieve this, as they do not support semantic discovery of services [28][29].

We propose a new service directory along the lines of UDDI but based on RDF/OWL [30] ontology instead of current tModel based taxonomy. This catalog is illustrated in Figure 4. As can be seen, the catalog captures the computing resources across all clouds in terms of "Capabilities", "Structural Relationships" and Policies (Preferences and Constraints).

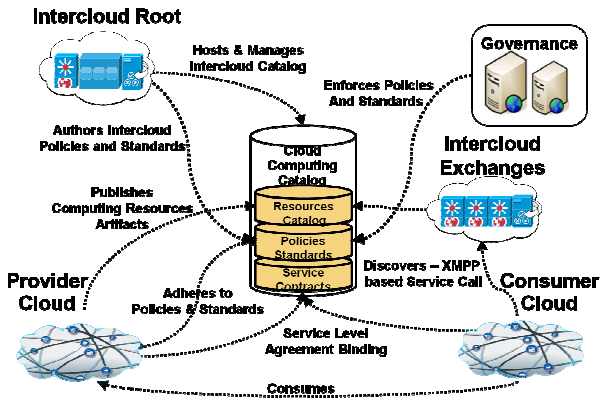

Figure 4. Cloud Computing Catalog

IX. RDF BASED RESOURCES ONTOLOGY

In order to ensure that the requirements of an intercloud enabled cloud provider are correctly matched to the infrastructure capabilities in an automated fashion, there is a need for declarative semantic model that can capture both the requirements and constraints of computing resources.

The chief objectives of the planned configuration are to provide cost effective use of computing resources and to meet the business objectives of the enterprise. In order to automate an environment whereby software agents versus traditional human users discover and consume services, intelligent ontology based service registries are needed for dynamically discovering and provisioning computing resources across various computing cloud environments (Amazon, Azure etc. etc.). We are proposing ontology based semantic model that captures the features and capabilities available from a cloud provider's infrastructure. These capabilities are logically grouped together and exposed as standardized units of provisioning and configuration to be consumed by another cloud provider/s. These capabilities are then associated with policies and constraints for ensuring compliance and access to the computing resources.

This model not only consists of physical attributes but quantitative and qualitative attributes such as "Service Level Agreements (SLAs)", "Disaster Recovery" policies, "Pricing" policies, "Security and Compliance" policies, and so on.

Our earlier work [13] explains how resources can be described, cataloged, and mediated using Semantic Web Ontologies with RDF. Although the terms "taxonomy" and "ontology" are sometimes used interchangeably, there is a critical difference. Taxonomy indicates only class/subclass relationship whereas Ontology describes a domain completely. The essential mechanisms that ontology languages provide include their formal specification (which allows them to be queried) and their ability to define properties of classes. Through these properties, very accurate descriptions of services can be defined and services can be related to other services or resources. Figure 5 shows a high level schematic of proposed ontology based semantic model.

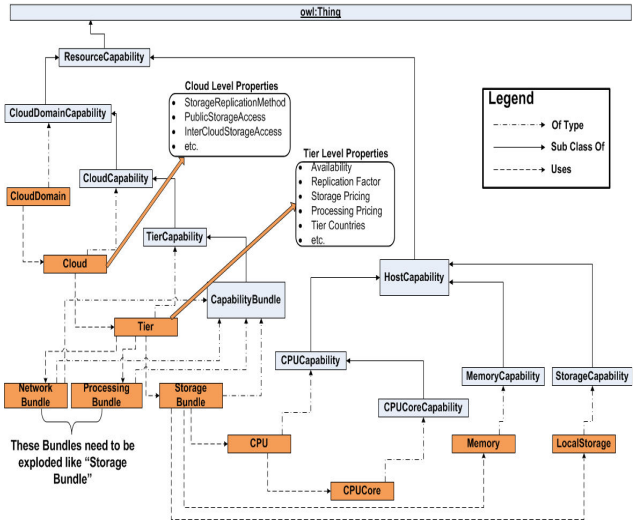

Figure 5. Cloud Computing Resources Ontology

X. XMPP BASED INTERCLOUD PROTOCOLS NEGOTIATION AND SERVICES FRAMEWORK

Part of interoperability is that cloud instances must be able to conduct dialog with each other. As part of the "Federated Storage" use case, one cloud must be able to find another cloud, which for a particular interoperability scenarios, is ready, willing, and able to accept an interoperability transaction with and furthermore, exchanging whatever subscription or usage related information which might have been needed as a pre-

 ISBN: 978-1-61208-103-8

cursor to the transaction. Thus, an Intercloud Protocol for presence and messaging needs to exist.

Extensible Messaging and Presence Protocol (XMPP) is exactly such a protocol. XMPP is a set of open XML technologies for presence and real-time communication developed by the Jabber open-source community in 1999, formalized by the IETF in 2002-2004, continuously extended through the standards process of the XMPP Standards Foundation. XMPP supports presence and structured conversation of XML data.

Our earlier work [12] explains in great detail as far as feasibility of XMPP as control plane operations protocol for Intercloud.

XI. SEQUENCING THE PROTOCOLS FOR INTERCLOUD ENABLED FEDERATED UNSTRUCTURED CLOUD STORAGE

The following is a high level sequence for "Intercloud Enabled Federated Unstructured Cloud Storage" amongst disparate cloud storage providers. "Inter-Cloud Exchanges" facilitate the negotiation process among disparate heterogeneous clouds in order to enable a seamless federated storage environment.

Detailed code samples, and how Cloud Computing resources can be described, cataloged, and mediated using RDF techniques for the various steps in this sequence diagram are outlined in our earlier work.

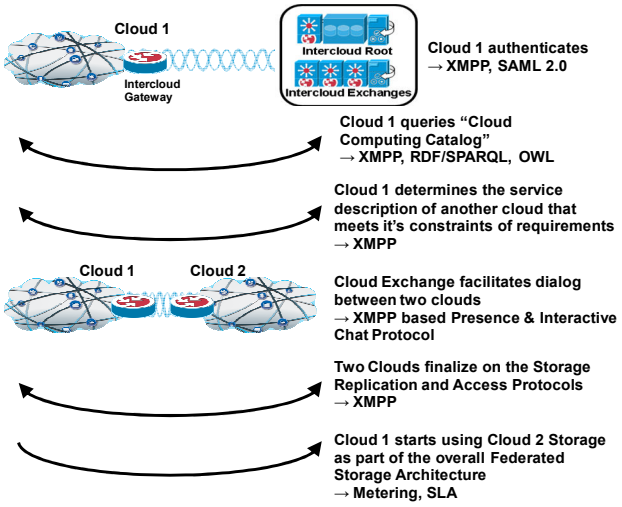

Figure 6. SSRP Unstructured Cloud Storage Sequence Diagram

XII. SSRP ENABLING CLOUD STORAGE ARCHITECTURE

To describe it in very simple terms, unstructured BLOB storage is a large-scale URI storage system. The goal of unstructured BLOB storage architecture is to provide a scalable mass storage solution. The system is designed to be scalable both in terms of total data stored, as well as the number of requests per second for that data.

In a cloud storage environment, due to the sheer number of data storage objects that must be stored and managed, it is very cumbersome and inefficient for traditional file systems supported by network storage models such as SAN and NAS to store and organize these data storage objects. These objects are stored near the users that will access them. Cloud vendors leverage their own proprietary storage solutions in order to efficiently mange the scale and QoS for data storage.

At its core, it is a middleware layer which virtualizes mass storage, allowing the underlying physical storage to be SAN, NAS, or DAS etc. The architecture also manages the replication of data between storage clusters in geographically distributed datacenters. The application can specify fine-grained replication policies, and the architecture layer replicates data according to the policies.

Data stored is typically organized over a two-level namespace. At the top level are buckets–similar to folders or containers–which have a unique global name and serve several purposes: they allow users to organize their data, they identify the user to be charged for transfers, and they serve as the unit of aggregation for audit reports. Each bucket, in turn, can store an unlimited number of data objects. Each object has a name, an opaque blob of data, and metadata consisting of a small set of predefined entries of user-specified name/value pairs. From optimization standpoint, it strives to locate data close to users to reduce latency.

Users can create, modify and read objects in buckets through the REST interface, subject to access control restrictions. End users typically create collections of files, and each file is identified with a URL. This URL can be embedded directly in a web page, enabling the user's browser to retrieve files from the cloud storage system directly, even if the web page itself is generated by a separate HTTP or application server. URLs are also virtualized, so that moving or recovering data on the back end file system does not break the URL.

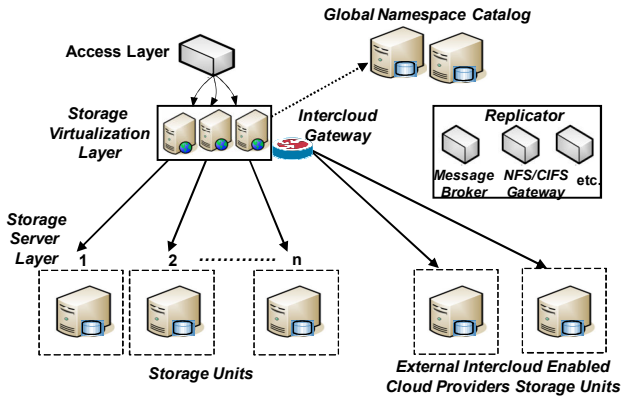

Figure 7. SSRP Enabled Cloud Storage Architecture

Figure 7 illustrates an SSRP enabled federated cloud storage architecture works. Following is a brief description of various components of this architecture.

- Access Layer: This layer is responsible providing various access methods to the underlying cloud based virtualized storage.

Protocols supported are HTTP based web services APIs (RESTful and SOAP based), *storage-as-storage* methods such as NFS, FTP, WebDAV, CIFS and iSCSI etc.

- Global Namespace Catalog: This catalog maintains a Global Namespace across multiple heterogeneous and distributed storage systems, data centers, and administrative domains across the storage Federation (across clouds), independent of their physical storage infrastructure. This catalog is used by the "Storage Virtualization Layer" to determine where a given "Storage Object" is physically located. It maps the Logical Namespace of the storage object to the Physical Storage resources within the overall storage federation. Logical Resources are used to group one or more replicas of Physical Resources, making it transparent to the user where the storage is ultimately stored. Logical resources, are used for load balancing among several storage replicas.
- Storage Virtualization Layer: This layer is essentially the heart of a typical cloud storage system and is responsible for virtualization of the underlying raw storage. This layer determines what to do with the request. For example, if there is a GET request it will first determine which storage instance this needs to be routed to in order for processing. In the event of storage instance unavailability, this layer will determine the next best storage replica the GET call needs to be forwarded to. It uses Global Namespace Catalog to determine the overall routing process. In the case of federated storage where replicas might be dispersed across multiple cloud systems, "InterCloudStorageAccess" is a cloud resources ontology element initially negotiated among storage cloud systems. It determines how the storage is accessed across cloud system boundaries. The "InterCloudStorageAccess" methods might include: NFS/sNFS, CIFS, iSCSI, WebDAV, Web Services (RESTful or SOAP based), BitTorrent, etc.
- Replicator: This layer is responsible for keeping multiple copies of data across cloud storage clusters, storage Replication is the sole purpose of this module. The write operations (PUT/DELETE) are propagated to other physical locations when they are committed to the primary storage location. It allows for replication across datacenters for a cloud. This component is responsible for N-way replication. In the case of federated storage where replicas might be replicated across multiple cloud systems, "StorageReplicationMethod" is a cloud resources ontology element initially negotiated among storage cloud systems. It determines how the data storage is replicated across system boundaries. The "StorageReplicationMethod" methods might include: AMQP based Message Broker delivered over UDT or TCP, FTP/sFTP delivered over UDT or TCP, NFS/sNFS, CIFS, iSCSI, WebDAV, Web Services (RESTful or SOAP based), BitTorrent, etc. These storage replication protocols are negotiated as part of the initial conversational dialog (using XMPP protocol) between cloud exchanges and participant cloud vendors.
- Storage Server Layer: This layer actually handles the reads/writes/deletes for the data storage objects to their physical location.

XIII. THE CONSUMER VISIBLE USE CASE – MOBILE STORAGE ROAMING

Once a federated storage cloud system has been enabled, we can envision the experience for an end user. A typical application of cloud storage will be to provide a transparent persistence for a mobile user.

As the user makes and receives calls, adds to their address book, snaps photos, and takes video, all of this content will immediately be streamed up to the cloud, persisting the data in a reliable way for the user. The mobile service provider will arrange for a cloud entry point which is "closest" to the mobile user so that they get the best uploading performance possible.

The cloud will use its internal replication algorithms and mechanisms to copy the data to several geographically storage nodes, most likely distributed around the geography for which the mobile service provider has coverage. If the user flies across the country, they will find their content replicated close to them and be able to experience superior download performance for their content as well. The user will also immediately be able to access or share their content from alternative channels, such as a web browser, or an Internet connected television, in real-time, as the cloud makes this content available.

Next, the user flies outside of the coverage area of the home service provider such that he is roaming for voice and messaging and data traffic. The mobile network already is designed to allow the roaming provider to service the user by providing the handling of this voice, messaging, and data traffic through an agreement with the home service provider such that the user does not realize he is being serviced by a roaming provider (perhaps other than the fees he will end up paying). However, his content is on a different continent that he is. It would be optimal for the user to provide an equivalent service for all of his data, such that for uploading and for downloading, some working set of his data was persisted in a nearby replicate.

This requires that the clouds which are implementing the storage for the mobile users, from the home and the roaming provider, operated in a federated manner, as we have described with SSRP. If this was the case, the mobile user would experience reliable and "nearby" storage, with all of the replication and performance

 ISBN: 978-1-61208-103-8

capabilities he enjoys with his home service provider. This scenario is illustrated in Figure 8.

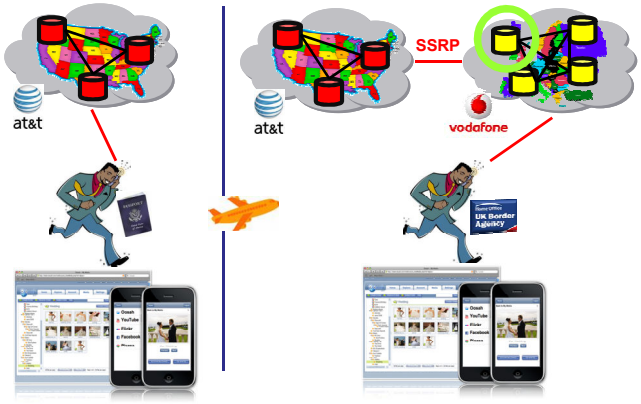

Figure 8. SSRP Enabled Cloud Storage Roaming use case

Here we show the user traveling from the USA, and ending in the UK, and the replicate which the roaming provider serves up to him, because they know where he is by virtue of his connectivity, is served up nearby to him, as illustrated.

XIV. CONCLUSIONS

The conclusion is that we have gone into great detail to test the proposal that Intercloud Topology and Protocols are suitable for a federated cloud storage use case scenario. We call the collection of these protocols and the resource definitions, SSRP. We have also described a meaningful real-world use case for this.

The next stages of our work are to develop details for Intercloud Topology and the governance process in support of proposed Intercloud Topology.

REFERENCES

[1] Youseff, L. and Butrico, M. and Da Silva, D., *Toward a unified ontology of cloud computing*, GCE'08 Grid Computing Environments Workshop, 2008.

[2] Lijun Mei, W.K. Chan, T.H. Tse, *A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues*, APSCC pp.464-469, 2008

[3] *Cloud Computing Use Cases Google Group (Public)*, at http://groups.google.com/group/cloud-computing-use-cases, http://www.scribd.com/doc/18172802/Cloud-Computing-Use-Cases-Whitepaper ,

[4] Buyya, R. and Pandey, S. and Vecchiola, C., *Cloudbus toolkit for market-oriented cloud computing,* 1st International CloudCom, 2009

[5] Yildiz M, Abawajy J, Ercan T., Bernoth A., *A Layered Security Approach for Cloud Computing Infrastructure*, ISPAN, pp.763-767, 2009

[6] Bernstein, D., Ludvigson, E., Sankar, K., Diamond, S., and Morrow, M., *Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability*, ICIW '09. Fourth International Conference on Internet and Web Applications and Services, pp. 328-336, 2009

[7] Bernstein, D., *Keynote 2: The Intercloud: Cloud Interoperability at Internet Scale*, NPC, pp.xiii, 2009 Sixth IFIP International Conference on Network and Parallel Computing, 2009

[8] *Extensible Messaging and Presence Protocol (XMPP): Core,* and related other RFCs at http://xmpp.org/rfcs/rfc3920.html

[9] *XMPP Standards Foundation* at http://xmpp.org/

[10] *W3C Semantic Web Activity*, at http://www.w3.org/2001/sw/

[11] *Resource Description Framework (RDF*), at http://www.w3.org/RDF/

[12] Bernstein, D., Vij, D., *Using XMPP as a transport in Intercloud Protocols,* Proceedings of CloudComp 2010, the 2nd International Conference on Cloud Computing, Barcelona, Spain, 2010.

[13] Bernstein, D., Vij, D., *Using Semantic Web Ontology for Intercloud Directories and Exchanges*, Proceedings of ICOMP'10, the 11th International Conference on Internet Computing, Las Vegas, USA, 2010.

[14] *James Murty, programming Amazon Web Services; S3, EC2, SQS, FPS, and SimpleDB, O'Reilly Press, 2008*

[15] *Domain Names – Concepts and Facilities,* and related other RFCs, at http://www.ietf.org/rfc/rfc1034.txt

[16] *Amazon Web Services*, at http://aws.amazon.com

[17] *Microsoft Azure*, at http://www.microsoft.com/azure/default.mspx

[18] *Advanced Message Queuing Protocol*, at http://jira.amqp.org

[19] *UDT – UDP-based Data Transfer*, at http://udt.sourceforge.net

[20] *Domain Name System Structure and Delegation*, at http://www.ietf.org/rfc/rfc1591.txt

[21] *Internet X.509 Public Key Infrastructure, Certificate Policy and Certification Practices Framework,* at http://tools.ietf.org/html/rfc3647

[22] *The Internet Society,* at http://www.isoc.org/

[23] *The Internet Corporation for Assigned Names and Numbers*, at http://www.icann.org/

[24] *Simple Authentication and Security Layer (SASL*), at http://tools.ietf.org/html/rfc4422

[25] *Security Assertion Markup Language (SAML*), at http://saml.xml.org/saml-specifications

[26] *OASIS UDDI Specification TC,* at http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uddi-spec

[27] *UDDI Registry tModels,* at http://www.uddi.org/taxonomies/UDDI_Registry_tModels.htm

[28] Paolucci, M., Kawamura T., Payne T., and Sycara K., *Importing the Semantic Web in UDDI*, Web Services, E-Business and Semantic Web Workshop, 2002.

[29] Moreau, L. and Miles, S. and Papay, J. and Decker, K. and Payne, T., *Publishing semantic descriptions of services*, First GGF Semantic Grid Workshop, held at the Ninth Global Grid Forum, Chicago IL, USA, 2003

[30] Web Ontology Language*,* at http://www.w3.org/TR/owl-features/

[31] *SPARQL Query Language for RDF*, at http://www.w3.org/TR/rdf-sparql-query/