Monitoring and Abstraction for Networked Clouds

Michael Scharf, Thomas Voith, W. Roome, Bob Gaglianello, Moritz Steiner, Volker Hilt, Vijay K. Gurbani
Alcatel-Lucent Bell Labs
E-mail: {first.last}@alcatel-lucent.com

***Abstract*—We consider the problem of building tightly coupled network and cloud management systems for “carrier clouds” based on an abstract view of the dynamic network state. Optimized resource placement in distributed clouds requires information about the internal network topology and state, in addition to other data center information. We present a distributed cloud system that accesses such data in an abstract way using the Application Layer Traffic Optimization (ALTO) protocol. The demonstration addresses how dynamic networking information can be gathered, how it can be abstracted and coupled with other data such as data center load, how it can be exposed, and how it can be integrated into a cloud management system. Our distributed cloud solution demonstrates that ALTO is well-suited for infrastructure-to-application information exposure.**

Keywords-Networked cloud; ALTO; monitoring; abstraction

I. INTRODUCTION

Cloud computing offers elastic access to computing, storage, and network resources. It allows network providers opportunities to improve their own operations as well as possibilities to generate revenue from new services. Increasingly more network providers offer carrier cloud solutions. Unlike cloud service providers running large data centers connected to the Internet, network carriers control their own network. As a result, they can offer tight network integration as differentiator, e.g., better resilience and Quality of Service (QoS).

Such carrier clouds offer several advantages compared to traditional over-the-top service provisioning: First, the cloud offer can make use of the Virtual Private Network (VPN) services offered by the network operator for enterprise customers; cloud resources can directly be attached to those VPNs. Second, due to their numerous points of presence, carriers can offer cloud services on a distributed platform with data centers closer to the user. Distributed clouds are thus better suited for the provisioning of low-latency or even real-time services. And third, the resource management can be improved by knowledge about the network internals. For instance, resource placement can optimize the service quality by taking into account internal network topology as well as resource availability.

In summary, this paradigm shift towards carrier clouds requires a much closer coupling of network and data center systems.

Application-Layer Traffic Optimization (ALTO) [1][2][3] is an application protocol developed by the Internet Engineering Task Force (IETF). Using ALTO, a network operator can expose an abstract view of the network to applications. This paper describes how ALTO can be used in a cloud management system. We have developed and integrated software that tightly couples ALTO and a cloud management system. An accompanying demonstration shows how that integration can improve the user’s experience in a distributed carrier cloud.

The rest of this paper is structured as follows. Section II introduces ALTO and its main use cases. Section III then explains the benefits of using ALTO in network clouds. In Section IV, the software architecture of our prototype is detailed. Section V then briefly describes a demonstration setup for our work. Finally, Section VI concludes the paper.

II. IETF ALTO FOR TOPOLOGY EXPOSURE

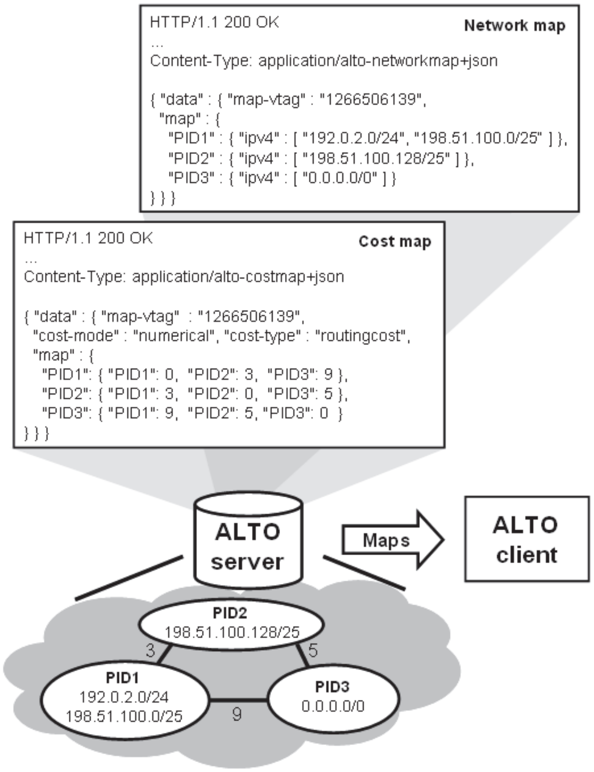
A. *ALTO Framework and Protocol*

ALTO is a request-reply protocol [1] that provides network information (loose topology structure, preferences of network paths, etc.) to applications. Such information allows an application to rendezvous with the optimal service when there is a plurality of service instances to choose from. The protocol is based on HyperText Transfer Protocol (HTTP) [4] and uses JSON encoding [5], thus allowing it to be easily integrated into web-based applications. The ALTO framework defines several information services, including the “map service,” the “map filtering service,” the “endpoint property service,” and the “endpoint cost service” [3]. These services are offered by one or more ALTO servers, which are queried by ALTO clients.

The “map service” provides a “network map” and a “cost map.” The network map presents a simplified and abstract view of the network by partitioning all network endpoints into a set of equivalence classes known as “PIDs.” An ALTO server can define those PIDs however it wants; they can be as fine-grained or as coarse as desired. The cost map is a matrix of the directed costs between those PIDs. As with PIDs, an ALTO server is free to define “cost” however it wants: bandwidth, latency, loss, expense, etc. In addition, the ALTO protocol allows a server to present several different types of costs, and clients can use the cost type they prefer.

The “endpoint cost service” allows an ALTO client to query costs or rankings for endpoints without retrieving the entire network map and cost map. As a result, the “endpoint property service” allows a client to obtain additional information about an endpoint, in the form of name-value pairs. The only mandatory property defined by the protocol is the “pid” property, whereby an endpoint can find out which PID it belongs to and can retrieve associated PIDs for other endpoints. Definition of other properties is left to the network provider; for instance, some network providers may want to impart additional information associated with an endpoint such as a connectivity type that denotes whether an endpoint is connected through an Asymmetric Digital Subscriber Line (ADSL), cable, or fibre access network.

Figure 1 gives an example of the map service. The network map partitions all endpoints into three PIDs. Each PID is defined by a set of Classless Inter-Domain Routing (CIDRs) prefixes [3]. An endpoint is in the PID corresponding to the CIDR with the longest mask that covers its Internet Protocol (IP) address. As a result, PID1 and PID2 contain a small number of endpoints, and PID3 contains everything else. The ALTO specification requires that an ALTO server is able to serve all IP addresses. The cost map also shown in Figure 1 gives the directed costs between each pair of PIDs. The ALTO server returns these maps as the response to an HTTP query, and encodes the maps using JSON.



Network map

```
HTTP/1.1 200 OK
...
Content-Type: application/alto-networkmap+json

{ "data" : { "map-vtag" : "1266506139",
  "map" : {
    "PID1" : { "ipv4" : [ "192.0.2.0/24", "198.51.100.0/25" ] },
    "PID2" : { "ipv4" : [ "198.51.100.128/25" ] },
    "PID3" : { "ipv4" : [ "0.0.0.0/0" ] }
}}}
```

Cost map

```
HTTP/1.1 200 OK
...
Content-Type: application/alto-costmap+json

{ "data" : { "map-vtag"  : "1266506139",
  "cost-mode" : "numerical", "cost-type" : "routingcost",
  "map" : {
    "PID1": { "PID1": 0,  "PID2": 3,  "PID3": 9 },
    "PID2": { "PID1": 3,  "PID2": 0,  "PID3": 5 },
    "PID3": { "PID1": 9,  "PID2": 5, "PID3": 0  }
}}}
```

Figure 1. Illustration of the ALTO architecture with a simple example for a network and cost map

B. *ALTO Use Cases*

ALTO was originally envisioned as a way to optimize traffic in peer-to-peer applications [6]. However, as the protocol evolved, it became apparent that it could find use in other domains as well. ALTO is very well suited for use with Content Delivery Networks (CDNs) [7]. CDNs consist of a large set of caching servers. When a client wants to access content offered by the CDN, the CDN's routing logic directs the request to the most appropriate caching server. CDNs can use the network topology and cost information from ALTO to make those decisions. In addition, ALTO is being considered as a promising solution for cloud computing, and it currently gets significant traction in that space [8]. We explore more on this in the next section.

A common thread in all of these use cases is that a requesting application must choose a resource from a set of resources. Such use cases are also called the "rendezvous class of applications" [9]. The key advantage of ALTO is that its base protocol is rather simple and can easily be extended according to the specific needs of new use cases.

III. ALTO INTEGRATION IN A NETWORKED CLOUD

A. *Architecture*

In order to serve the customers' needs, a cloud solution requires an integrated system for managing computing, storage, and networking resources. Network monitoring and control are particularly crucial for networked cloud solutions that are served from multiple, distributed data centers interconnected by a Wide Area Network (WAN).

Figure 2 illustrates how ALTO integrates into a networked cloud. The cloud management system gets information about the internal network topology from the ALTO server, and it uses that information to manage the resources. Most notably, the cloud management system can select the best data center to serve a given customer, taking into account both Quality-of-Service (QoS) constraints (e.g., delay) as well as network status (e.g., internal routing and load information). ALTO can thus be a simple, standardized interface to support the management of a network-aware cloud [8].

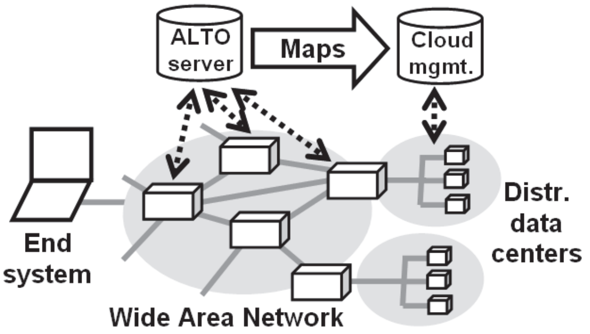


Figure 2. Network-awareness by ALTO for cloud management

B. *Gathering Input Data*

The ALTO protocol can expose network topology details, but it is not limited to that. An ALTO server can obtain the network topology and cost information though many different channels, including monitoring routing protocols and polling network equipment or network management systems.

In a cloud environment, the fundamental idea is to use the ALTO protocol to present a simplified view of the network data that can be obtained from more complex protocols like Open Shortest Path First (OSPF), Intermediate System To

Intermediate System (IS-IS), Border Gateway Protocol (BGP), Simple Network Management Protocol (SNMP), etc.

In addition, network data from ALTO can be combined with information about the computational environment to provide a comprehensive view of the resources in a networked cloud. With this information, the cloud management system can optimize resource placement by taking into account the network performance as well as the compute and storage resources.

Historically, network operators have been reluctant to expose the structure and capabilities of their network to third parties, or for that matter, to different divisions within the same company. This is also called the "topology hiding" principle. However, in case of networked clouds, operators not only have an incentive to expose at least some of that data, but they can do so in an abstract manner through a well-defined interface in the form of ALTO. The ALTO framework was explicitly designed to provide enough information about the network internals to satisfy the needs of the clients while still maintaining the network operators' desires to hide the detailed structure of the network.

First, by design, the ALTO protocol only provides an abstract view on the network. As a result, users do not see sensitive network configuration information. Second, ALTO is a win-win for both the network operators and the application designers. Not only does ALTO enable an application to make better traffic and service placement decisions (e.g., VM placement), but it also enables the network to inform applications on a traffic matrix suited to its needs, e.g., to avoid links with high costs. This gives the network more control than solutions without such an interface.

Finally, in the context of networked clouds, only the cloud management system accesses the ALTO server; ordinary users do not. The ALTO server can use firewalls or standard HTTP security techniques to ensure that only authorized clients are allowed to obtain the network data. Network information is then not exposed to the outside world.

C. *ALTO Extensions For Networked Clouds*

ALTO was originally designed to improve peer-to-peer applications, but it is a general protocol and it can be used for many different applications. In particular, cloud computing can benefit from several protocol extensions that are currently in progress. Since ALTO is based on HTTP and JSON, the base protocol [3] can easily be extended.

There are a number of proposed ALTO extensions that optimize ALTO for networked clouds. Table I provides an overview of recently suggested ALTO extensions.

TABLE I. PROPOSED ALTO EXTENSIONS FOR CLOUDS

Cloud requirement	Potential solution	Ref.
Efficient maps for multiple costs (e.g., network+CPU)	Multi-cost maps	[10]
Scalable, incremental updates for dynamic data	Incremental updates	[11]
Time scheduling for data center workload patterns	Cost scheduling	[12]

Further potential ALTO enhancements include a server-initiated notification mechanism and coordination mechanisms between ALTO servers. However, since the requirements for these extensions are in early stages of discussion in the ALTO working group, we do not elaborate on them further in this document.

IV. SOFTWARE ACHITECTURE

A. *Overview*

We have developed a prototype for a distributed networked cloud system that shows the benefits of using the ALTO protocol for exposing network information.

Our prototype system incorporates all relevant components for an Infrastructure-as-a-Service (IAAS) offer from a distributed cloud. The system consists of an open source cloud management system, our in-house ALTO server, virtual machines, and an emulated network. Figure 3 shows the main components of our prototype, which are further explained below.

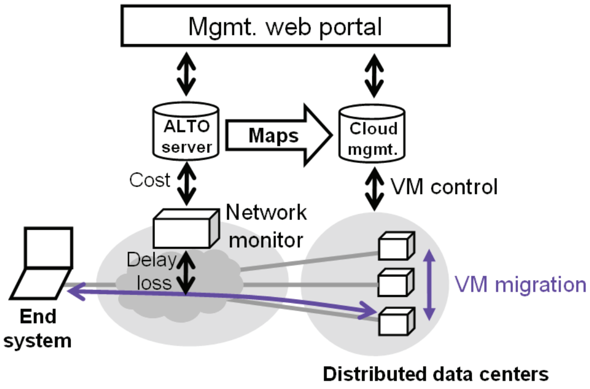


Figure 3. System architecture for our prototype demonstrating the use of ALTO in a networked cloud

B. *ALTO Server*

We implemented our own ALTO server. It offers all services defined by the ALTO protocol according to referenc [3], including the "map service," the "map filtering service," the "endpoint property service," the "endpoint cost service," as well as the "directory service" which defines the URIs for all available services. Our server is fully standard-compliant and has successfully participated in interoperability tests organized by the IETF (see [13]).

The ALTO server internally maintains an information base that encodes the preferences as configured by the operator of the server. The ALTO information base includes the network locations, their corresponding properties, as well as the costs between pairs of network locations.

C. *Network Monitoring Tool*

As depicted in Figure 3, we also implemented a monitoring tool to dynamically update the ALTO costs based on live measurements of network properties, such as delay and loss. This provides the required information about the network infrastructure. For instance, we use Internet Control Message Protocol (ICMP) measurements in the system to monitor the current network status, and we update the costs in the ALTO server accordingly.

The ALTO costs are calculated from a weighted sum of the latency and packet loss values:

$$C = \alpha \cdot L + \beta \cdot P \qquad (1)$$

Therein, C is the ALTO cost, L the measured latency, and P the measured packet loss ratio. The weight factors α and β are assigned such that the cost implied by a loss rate larger than few percent exceeds the cost by typical Internet latency values. Such loss rates are a sign of significant congestion and indicate that the corresponding path should be avoided if possible.

As a result, higher delays or packet loss result in an increase of the ALTO costs. The use of a single cost value abstracts from the specific network performance metrics used as input. Obviously, additional network performance metrics can easily be taken into account as well, if corresponding measurement data is available.

In order to ensure that the ALTO server correctly represents the actual network conditions, access to up-to-date monitoring data is needed. However, obtaining real-time status information results in processing and communication overhead. The system thus has to trade-off the data accuracy and the measurement effort. Figure 4 illustrates these steps as well as the inherent trade-off between the accuracy of information and the level-of-detail.

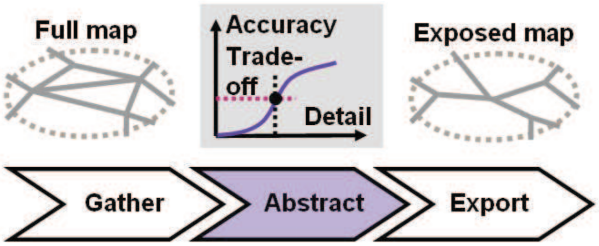


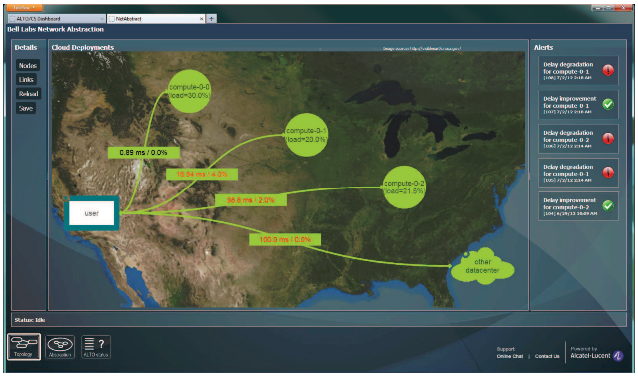
Figure 4. Processing and abstraction of monitoring information for ALTO

D. Cloud Management System And Web Portal

Our prototype also includes a cloud management system and Web portals. We use CloudStack [14] to manage the virtual machines (VMs). CloudStack is an open source software system for creating, managing, and deploying infrastructure cloud services. CloudStack can use several hypervisors for virtualization, and it also offers various Application Programming Interfaces (APIs). It should be noted that our architecture is generic and CloudStack could be replaced by another cloud management system.

One of the key challenges in carrier clouds is allocating resources intelligently. A cloud manager must take into account the computation power, storage capacity, and network connectivity to properly allocate the requested resources to the available infrastructure (data center sites, VPNs., etc.).

We implemented a Web portal for resource management and administration in a distributed cloud. The portal provides two views. One is a graph of the network, showing the computing nodes and communication links. The graph is appropriately abstracted and simplified. Figure 5 shows such a graph for the demonstration setup. This view shows the latency and loss for each link as measured in the network. In addition, it also shows the CPU load for each computing node.



Delay degradation for compute-0-1 [108] 7/2/12 2:18 AM

Delay improvement for compute-0-1 [107] 7/2/12 2:18 AM

Delay degradation for compute-0-2 [106] 7/2/12 2:14 AM

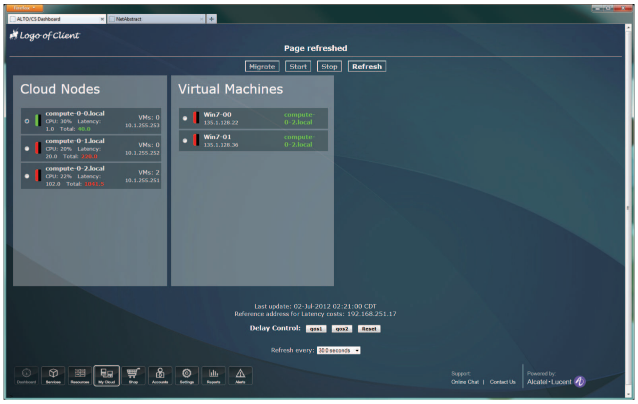
Delay degradation for compute-0-1 [105] 7/2/12 2:14 AM

Delay improvement for compute-0-2 [104] 6/29/12 10:09 AM

Status: Idle

Figure 5. Screenshot of our Web portal prototype that shows how a network performance issue is detected and exposed to ALTO

The second view is a dashboard for managing the VMs (Figure 6). The dashboard is a tabular view of the VMs and computing nodes (hosts). It shows which VM is running on which host, the total load on each host, and the cost of communicating with that host. The dashboard gets the communication costs from the ALTO server, and the host and VM information from the CloudStack manager via well defined and published interfaces. In addition, the dashboard allows the user to migrate a VM from one host to another, also via CloudStack's management API.



Page refreshed

compute-0-0.local CPU: 30% Latency: 1.0 Total: 40.0 VMs: 0 10.1.255.253
compute-0-1.local CPU: 20% Latency: 20.0 Total: 220.0 VMs: 0 10.1.255.252
compute-0-2.local CPU: 22% Latency: 102.0 Total: 1041.5 VMs: 2 10.1.255.251

Win7-00 135.1.128.22 compute-0-2.local
Win7-01 135.1.128.36 compute-0-2.local

Last update: 02-Jul-2012 02:21:00 CDT
Reference address for Latency costs: 192.168.251.17
Delay Control: qos1 qos2 Reset
Refresh every: 30.0 seconds

Figure 6. Screenshot of our Web portal prototype dashboard

In addition to simply displaying the various types of costs, the dashboard also identifies problem hosts (i.e., those hosts suffering from CPU overload or high communication costs), and uses intelligent optimization algorithms to suggest alternatives, considering both the CPU load and the communication costs.

E. *Cloud Services*

Carrier clouds offer cloud services from distributed data centers. The decentralization and tight network integration enables services with better performance and higher availability. The focus of this work is Infrastructure-as-a-Service (IaaS), i. e., a carrier cloud that offers a constant low-latency access to VMs, automatically migrating them closer to the user when the network detects that the latency or other pertinent cost is too high. The exposure of network information is particularly relevant for services that have high networking requirements, such as low latency (gaming, remote desktop, etc.) or high bandwidth (most notably, video services).

F. *Open Issues*

Carrier clouds raise several challenges that are not fully addressed by this paper and will require further work. First, this paper describes proof-of-concept solutions and software prototypes. Further studies and development work are needed to evolve the demonstrated scenario into a mature and commercially available solution. For instance, while our prototype uses a rudimentary placement optimization algorithm a production system would use more sophisticated algorithms such as those described by Alicherry and Lakshman [15].

Second, for now we've focused on carrier clouds: integrated service offerings where a single network operator provides both the data centers and the communication network. If different organizations are responsible for those infrastructures, we will need extensive inter-domain communication. The ALTO framework does not completely address these issues, and may require future extensions [8].

There are also discussions whether ALTO can be used even more generally in the context of Software-Defined Networks (SDN) [16]. As shown in Figure 7, the SDN paradigm implies that specific network details can be programmed through a north-bound interface that provides abstracted network mappings. SDN is a promising network architecture for cloud computing. Gurbani et al. [9] show why ALTO is a well suited mechanism for network abstraction in data centers and clouds using the SDN paradigm.

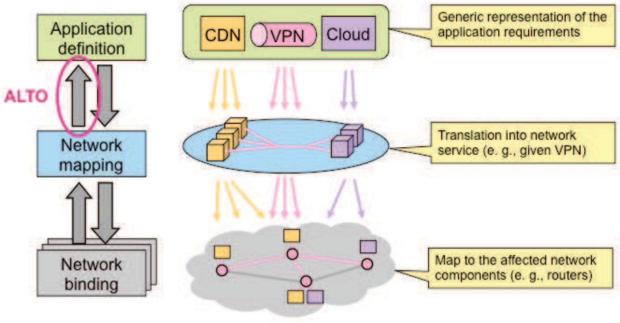


Figure 7. ALTO as a candidate solution for network mapping in SDN (cf. Ref. [11])

Furthermore, we do not consider risks from hidden dependencies in clouds, as described in [17]. Topology exposure could result in stability risks due to unpredictable interactions between independently developed but interacting cloud computations. In addition, in this paper we've neglected resilience issues and correlated failures, which can be a major problem in cloud computing [17].

Finally, wide-spread use of carrier clouds could result in further challenges. For instance, each server in a cloud will need a global routable address. We take it for granted that the cloud provider can get enough Internet Protocol version 4 (IPv4) addresses, and we do not discuss the specific implications of using Internet Protocol version 6 (IPv6) in carrier clouds.

V. NETWORKED CLOUD RESEARCH DEMONSTRATION

A. *Demonstration Setup*

For the sake of simplicity and portability, we designed our demonstration system with minimum complexity and small enough to fit in a suitcase. We emulate each distributed data center with a single laptop computer (a data center with one server, if you will). Our communications network is a single router connecting the data centers and the user's computer. The following sections describe the research demonstration in more detail.

As specific use case, we show a virtual desktop application served from data centers. Remote desktops are a promising use case for outsourcing enterprise infrastructure, and they offer many advantages to both the employees and the corporate IT organization. However, remote desktops are very latency-sensitive. If the latency rises, users will abandon remote desktops in lieu of dedicated laptops. Therefore this is a good example of an application that is better provided by a distributed, network-aware cloud, rather than a traditional, centralized IaaS cloud.

B. *Monitoring and Abstraction*

One key focus of our prototype is to demonstrate how ALTO allows us to optimize resource management in a distributed cloud. In particular, we emphasize the use of real-time network information.

To be effective, resource management requires accurate information about the network status. However, that requires measuring and processing huge amounts of data, and that in turn requires compromises regarding the detail-level and the timescale of data.

Our demonstration shows how dynamic networking information can be gathered, how it can be abstracted and coupled with other data such as data center load, how it can be exposed, and, finally, how it can be used by a cloud management system to optimize the system performance.

C. *Intelligent Network-Aware VM Placement*

The other main focus of our demonstration is to show how network monitoring information can be used to balance load and to optimize placement decisions. Our Web portal includes a placement decision algorithm that assigns a score to each data center, orchestrating the information about available network and compute resources. Virtual machines are placed at the site with the best score.

Furthermore, the demonstration illustrates how a cloud management system can react to changes in the network, e.g., an increased delay. Figures 5 and 6 depict how our Web portal detects and visualizes an increase of the delay on a link and advertises this information, in addition to updating the corresponding ALTO cost.

We also show that in case of network problems such as delay or loss, migrating virtual machines to other data centers can mitigate performance issues in the network. This is a key value proposition of a distributed cloud: a service can be offered from different locations, and the location can be selected based on the network conditions.

VI. CONCLUSION

Cloud computing gives network providers the opportunity to improve their existing operations while generating revenue from new services. Network providers have the unique advantage of controlling both network and data center resources, and they can deliver a wide range of services from a single, elastic environment. Carrier clouds thus enable providers to extend the reach of clouds beyond today's centralized and transaction-oriented architectures.

This paper explains how Application Layer Traffic Optimization (ALTO) protocol can be used to orchestrate and expose information in distributed clouds, and how this enables effective delivery of high quality services. The demonstration shows a distributed cloud system with dynamic service placement algorithms. The prototype accesses network and data center information using ALTO. Real-time information enables smart placement and migration of latency-sensitive services in order to optimize the user experience.

As a result, the demonstration setup highlights that the ALTO protocol is well-suited to provide infrastructure-to-application information [8] in networked clouds.

REFERENCES

[1] J. Seedorf, E. Burger, “Application-Layer Traffic Optimization (ALTO) Problem Statement”, RFC 5693, Oct. 2009

[2] S. Kiesel, S. Previdi, M. Stiemerling, R. Woundy, Y R. Yang, “Application-Layer Traffic Optimization (ALTO) Requirements”, Internet Draft, work in progress, 2012

[3] R. Alimi, R. Penno, Y. Yang, “ALTO Protocol”, Internet Draft, work in progress, 2012

[4] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, RFC 2616, 1999

[5] D. Crockford, “The application/json Media Type for JavaScript Object Notation (JSON)”, RFC 4627, 2006

[6] J. Livingood, L. Popkin, C. Griffiths, R. Woundy, Y. Yang, “Comcast’s ISP Experiences in a Proactive Network Provider Participation for P2P (P4P) Technical Trial”, RFC 5632

[7] B. Niven-Jenkins, G. Watson, N. Bitar, J. Medved, S. Previdi, “Use Cases for ALTO within CDNs”, Internet Draft, work in progress, 2011

[8] E. Marocco, V.K. Gurbani, “Extending the Application-Layer Traffic Optimization (ALTO) Protocol,” Internet-Draft, work in progress, 2012

[9] V. Gurbani, M. Scharf, T. V. Lakshman, V. Hilt, “Abstracting network state in Software Defined Networks (SDN) for rendezvous services”, Workshop on Software Defined Networks, co-located with IEEE ICC, June 2012

[10] S. Randriamasy, N. Schwan, “Multi-Cost ALTO”, Internet-Draft, work in progress, 2012

[11] N. Schwan, W. D. Roome, “ALTO Inremental Updates”, Internet-Draft, work in progress, 2012

[12] S. Randriamasy, N. Schwan, “ALTO Cost Schedule”, Internet-Draft, work in progress, 2012

[13] “ALTO interoperability results in IETF-81”, http://www.ietf.org/mail-archive/web/alto/current/msg01181.html

[14] CloudStack - Open Source Cloud Computing, http://cloudstack.org

[15] M. Alicherry, T. V. Lakshman, “Network Aware Resource Allocation in Distributed Clouds”, Proc. IEEE INFOCOM 2012

[16] D. Stiliadis, F. Balus, W. Henderickx, N. Bitar, M. Pisica, “Software Driven Networks: Use Cases and Framework”, Internet-Draft, work in progress, 2012

[17] Bryan Ford, “Icebergs in the Clouds: the Other Risks of Cloud Computing”, 4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '12), June 12-13, 2012