CloudState: End-to-end WAN Monitoring for Cloud-based Applications

Aaron McConnell, Gerard Parr, Sally McClean, Philip Morrow, Bryan Scotney
School of Computing and Information Engineering
University of Ulster
Coleraine, Northern Ireland
Email: a.mcconnell@ulster.ac.uk, gp.parr@ulster.ac.uk, si.mcclean@ulster.ac.uk, pj.morrow@ulster.ac.uk, bw.scotney@ulster.ac.uk

***Abstract*—Modern data centres are increasingly moving towards more sophisticated cloud-based infrastructures, where servers are consolidated, backups are simplified and where resources can be scaled up, across distributed cloud sites, if necessary. Placing applications and data stores across sites has a cost, in terms of the hosting at a given site, a cost in terms of the migration of application VMs and content across a network, and a cost in terms of the quality of the end-to-end network link between the application and the end-user. This paper details a solution aimed at monitoring all relevant end-to-end network links between VMs, storage and end-users. With this knowledge at hand, it becomes easier to optimise the arrangement of VMs and content with a distributed cloud environment such that resident applications respond in a timely manner, both between cloud-based application components and in the delivery of the application to the end-user. Results show how this system provides network information which influences the choice of location for hosting applications and data.**

Keywords-Cloud Computing; WAN Monitoring; Cloud Networking

I. INTRODUCTION

Recent years have seen data centres, firstly adopting virtualisation solutions in order to consolidate servers, and then moving to Cloud environments where Cloud instances can be scaled across distributed resources depending on load [12][5]. Clouds allow applications to be migrated to remote hosts outside of the physical realm of the local data centre [1]. Applications may be statically hosted at a remote location, or it may happen dynamically in a ”fail-over” event, i.e., when local data centre resources are saturated and a resource-starved application is temporarily migrated to a remote location where available resources are such that it performs adequately. Depending on the network conditions between data centre locations, moving an application to another location may be ill-advised. It may be that the application (or service) communicates heavily with another application at its original location or with a particular data-store. It may also be the case that the proposed location is further away, in network terms, from the end-user of the application. If the network conditions between the proposed location and the end-user are sufficiently poor then the delivery of the application to the end-user will not be acceptable, despite the application having ample resources within the physical data centre.

It is therefore necessary to have a periodic, automated means of measuring the state of the WAN link between any two addresses relevant to the successful delivery of an application to end-users. This measurement should be taken periodically, with the time between polls being short enough that sudden changes in the quality of the WAN are observed, but far enough part so as not to flood the network with monitoring traffic. Data collected from polls should also be logged at a central location in order that decision-making about application performance and placement can be made, by a cloud management solution, with a full view of the distributed data centre available, including WAN metrics. Intelligent use of relevant WAN data can enable decision-making to occur which can pre-emptively and reactively lead to action which will ensure applications meet their Service Level Agreements (SLAs). It is also possible, given a WAN history between two addresses, to observe trends related to time of day and workload.

Providing live WAN information, specifically for network-aware placement of cloud-based applications and services, is of commercial importance to vendors of existing cloud vendors where hybrid cloud scenarios are used when private cloud resources are saturated. It is envisaged that network-awareness will be more important in future cloud topologies, where users may frequently migrate their content between cloud vendors in order to save money [11] or increase performance. It is with this in mind that a new solution, called CloudState, has been developed to provide real-time information on the state of the end-to-end WAN link between any two communicating entities related to the operation of the Cloud-based application and its delivery to the end-user. The end-user is defined, during the course of this work, as a corporate customer for a Cloud Provider. The end-user address communicated with is one at the edge of a customer’s LAN, typically a router with a WAN IP address.

A. Related Work

Commercial solutions exist which monitor WAN links and provide optimisation, both to the link and to the placement of virtual applications at the end of a link. Ipanema’s Ip—Engines [6] are placed at either end of a WAN link, one entity at the data centre and one between the customer’s edge

 ISBN: 978-1-61208-271-4

router and LAN. Ipanema's central management software then provides monitoring data for all WAN links relevant to an application component. Given that an Ip—Engine is required at each end of a WAN link, the cost of installing the Ip—Engines, and the intrinsic financial cost involved in scaling up such a WAN monitoring system, it is clear that an improvement may be made, in terms of a WAN monitoring solution, in the case where a Cloud implementation can be scaled up quickly across various locations (and cloud providers). Work has also been carried out with a focus on Application-Layer Traffic Optimisation (ALTO) [9] [4], which has been developed by the Internet Engineering Task Force (IETF). The ALTO protocol may well develop into a standardised means of acquiring network and traffic information, but at present it requires at least one ALTO server to be put in place and a number of ALTO clients, which should be integrated with end-user Web applications. The perspective of the work presented in this paper is that Cloud WAN monitoring system should ideally be simple, non-invasive (i.e., require no special hardware or servers to be configured), scalable (i.e., run inside a VM so that it can be migrated and cloned) and low-cost, both in terms of financial outlay and in terms of resource requirement.

GEANT's perfSONAR provides a range of software to monitor networks and report performance measurements [3]. PerfSONAR aims to provide monitoring data from networking entities, e.g., routers, along the end-to-end network path. This system assumes that, although the entities may be in different domains of ownership, sufficient performance data will be made available, by the commercial vendors involved, in order that end-to-end network performance can be quantified. It is also unlikely that detailed per-hop performance data is required in the case where a decision is to be made about where to place a virtual instance of an application or service. The decision about placing the application or service is only concerned with the quality of the end-to-end link, between the host server and the end-user, and not with the specific performance of each entity en route. PerfSONAR would offer a comprehensive network monitoring solution in the case where multiple cloud providers agree to implement the system and where interfaces for acquiring performance data is shared (as is the case with the OPTIMIS project [15]). An assumption cannot be made that this communal arrangement exists in a cloud computing scenario. Therefore, there is scope for the design and development of a simple solution which monitors only the end-to-end network path.

The Network Weather Service (NWS) [14] is another distributed system for monitoring network performance, with a focus on dynamically forecasting the performance or networking entities. Like perfSONAR, this system requires that, in the case where the end-to-end network path crosses different domains of ownership, performance data and forecasting information are made available to various commercial cloud vendors. The NWS is quite complex in the regard that it requires a name server, memory host, sensor host and forecaster host. As is the case with perfSONAR, there remains scope for a simple end-to-end monitoring solution. The next section provides a description of the CloudState model, followed by a section describing the prototype solution. Experimentation and results are described in the subsequent section and the final section discusses conclusions and further work.

II. Architecture Description

CloudState is aimed at providing a software-based VM-embedded, scalable, migratable WAN analyser for Cloud Computing. A full, up-to-date view of all LAN/WAN links is possible with CloudState, with instances of the application strategically placed throughout a cloud.

The aim of this work is to provide a means of monitoring network capabilities across LANs and WANs, for distributed applications on privately-owned hardware, for elements running on third-party equipment and for a continuous assessment of the link to the end-user. The resultant data, combined with the other cloud performance metrics, provides the means by which QoS guarantees can be ensured, via SLA-compliance, and for optimisation mechanisms to ultimately ensure that the VPC is making best use of available resources.

A. A VM-Embedded Solution

CloudState is designed to reside inside a VM. The central reason for this is because it reduces the amount of work required to install CloudState, configure it and place it within the VPC. Current network assessment tools require hardware installation or special servers and clients, as mentioned in section I-A. It is necessary, for a dynamically-changing distributed topology, that the level of installation, configuration and engineering required is minimised. The ultimate aim with CloudState is that it can be cloned and migrated to a remote location and run with minimal setup. It is designed to require a list of IP addresses, representing the other end of the links to test, its own IP address and connectivity to a centralised Cloud Management Database (CMDB) in order to log data.

B. Multiple Link Monitoring

CloudState is designed to communicate with a list of IP addresses if required. These addresses may represent a number of end-user locations which can potentially use the cloud location where the CloudState instance is resident. These addresses are repeatedly polled at a defined time interval and the results stored in the CMDB. Figure 1 illustrates a typical topology scenario. The centralised CMDB is used in order that requests for network performance data can be made from a single source in order to assess the suitability of numerous sites for placing an application or service.

C. Communication Protocols

The current architecture of CloudState uses the Internet Control Message Protocol (ICMP) protocol to send echo-request packets to and from the destination IP address. This approach is used in order that any IP address can be queried

 ISBN: 978-1-61208-271-4

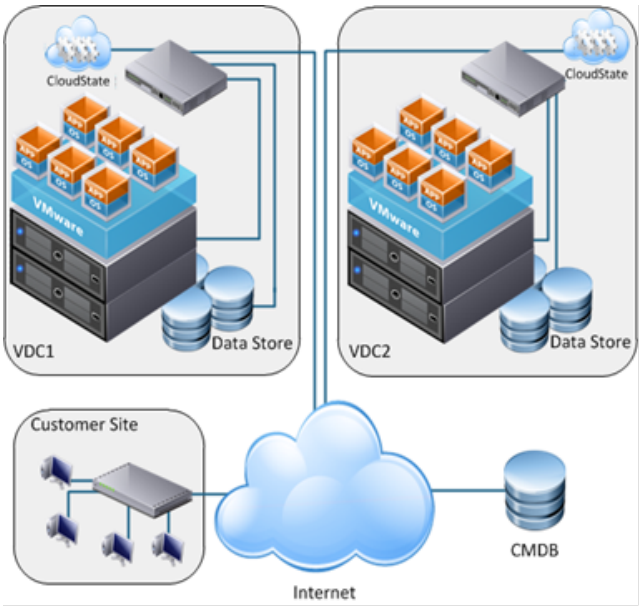

Figure 1. CloudState locations within the cloud

and so that no specialised hardware or software is required at the destination address.

III. CloudState Prototype

CloudState uses BWPing [2] as a library to test each link for bandwidth, latency and packets dropped. BWPing is used to send a number of user-sized ICMP echo-request packets to a destination IP address. The response from the destination, and the volume of packets transmitted, allows bandwidth, latency and packets dropped to be calculated.

The network metrics acquired using BWPing provides the basic functionality of CloudState. The native BWPing C class was converted to C++, an object of which was created in the main CloudState class. Nokias QT signal slot libraries [13] were used to allow BWPing to emit signals each time a ping process was completed. It is necessary for the destination address being pinged by CloudState to be configured so that it responds to ICMP echo-requests, otherwise CloudState will not provide metrics. This is an existing limitation of CloudState, but one which will be addressed in future versions. At present it is possible to provide an IP address to CloudState for which ICMP packets are permitted through a firewall.

CloudState runs on a host at the same physical location as the hosts running VM-based user applications. This means that WAN metrics, returned by CloudState for each address, should match those experienced by each application. From this location, CloudState can communicate with any address relevant to the delivery of each application, e.g., databases, Web Services and end-users. The destination address focused on in this work is the end-user. CloudState is used to gather link metrics between the host, resident at the CloudState location, and the user.

The CloudState user is presented with an interface, show in Figure 2. From this interface the user may define operational parameters, e.g., packet size, transfer speed, transferred volume, for the underlying BWPing echo-request operation.

CloudState provides the administrator with an interface by which a connectivity parameters can be defined so that the CMDB can be reached. CloudState connects to the remote CMDB and writes the results of each poll to the CloudState database table. Poll metrics, as shown in table I, are stored in the CMDB, along with the address of the CloudState agent, the destination address and a time stamp for the ping operation. Connectivity is achieved using the QT QMySQL Linux driver.

IV. Experimentation

A controlled test environment was created in order to validate the metrics returned by CloudState, and to assess the impact of CloudState on both the source host, the destination addresses and the network. The test environment comprised of a Dell R515 Server with two AMD 6-core processors, 16 GB of main memory and twelve 1 Gb network interface cards (NIC). The VMWare ESX 4.1 hypervisor was installed on this server (with load balanced across the twelve NICs) and a CloudState VM placed on it using VMWare vCenter 4.1, which was installed on another networked machine. CloudState was installed within a Ubuntu Linux VM with 1 virtual processor, 512 MB of RAM and a 4 GB virtual thin-provisioned hard drive.

CloudState was used to assess the bandwidth and latency of a known 100 Mbps link. A VM was polled, running on another identical Dell R515 host, connected by a single 100 Mbps switch. Results for the bandwidth returned varied depending on the parameters used for the underlying BWPing operation. Packet sizes ranging from 500 bytes to 1500 bytes (the largest allowed by Ethernet at the network layer) were tested as well as a range of transmission data volumes. The aim is to momentarily saturate the network so that the bandwidth can be quantified, but for the monitoring load to be create minimal intrusion on the network and destination address.

Two methods were used to validate the results returned by CloudState: IPerf [7] was used to ensure the bandwidth values returned by Cloudstate were similar to those of IPerf, when no emulated network degradation was forced, and a

TABLE I
CloudState CMDB fields

Parameter	Description
Source	The source IP address of the CloudState application
Host	IP address of the represented host
Target	The target IP address of the ping operation
PacketSize	The size of each packet transmitted to the target
Totalpkts-tx	The number of packets transmitted to the target
Totalpkts-rx	The number of packets received from the target
Vol-tx	The number of bytes transmitted (packetSize x totalpkts-tx)
Vol-rx	The number of bytes received from the target
time-secs	The time taken for the complete operation
Speed-kbps	The bandwidth of the link
Rtt-min	The minimum round-trip-time taken
Rtt-max	The maximum round-trip-time taken
Rtt-mean	The mean round-trip-time taken
Date/Time	Date and time of ping

 ISBN: 978-1-61208-271-4

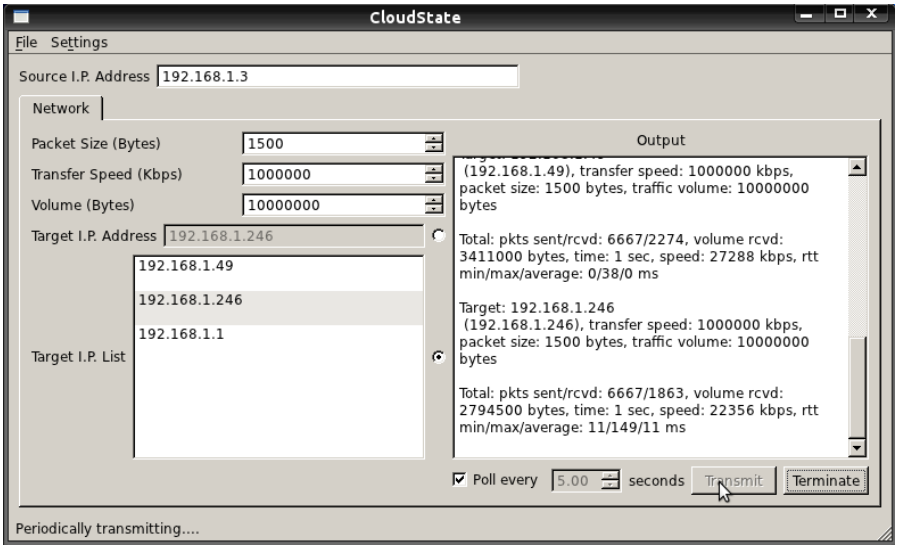

Figure 2. The CloudState user interface

simple Linux ping was used to assess the latency. Over the native 100 Mbps link, IPerf returned a mean (of 5 polls) bandwidth rating of 82.34 Mbps, with an average volume of 98.2 MB of data transferred at each poll. The mean (of 5 polls) latency returned by ping was 1 ms. CloudState returned a slightly lower average bandwidth value of 78.99 Mbps with a packet size of 1500 bytes and a volume of 10 MB. It is acknowledged that further work is required in order to ensure the bandwidth values returned match those returned by other bandwidth measurement tools, but at this point there is an argument that a less accurate measurement is acceptable given that only 10.18% (of the volume of data IPerf used) was loaded onto the network to return a bandwidth value which was 95.93% accurate. The inaccuracy is probably due to the fact that the volume of data transmitted does not fill the bandwidth available. An algorithm is required that increases the volume at each poll until a small percentage of packets is dropped, indicating bandwidth limitations.

The fixed values for packet size and volume currently used proved to be most accurate in assessing bandwidth and latency for each link tested. Figure 3 illustrates the bandwidth returned as the packet size was changed from 500 bytes to 2000 bytes in steps of 500 bytes. A traffic volume of 10 MB was used for each test. The graph shows a bandwidth increase up to a packet size of 1500 bytes followed by a steep decline in bandwidth when a packet size of more than 1500 bytes was used. More packets are sent when the packet size is small, e.g., 500 bytes, in order to achieve the same transfer volume. This increases the per-packet delay because each packet must be processed. The level of throughput in a given timeframe is therefore reduced. Large packets can also reduce throughput because of the time required to process the amount of data in each packet. This is evident in Figure 3 when the packet size is increased beyond 1500 bytes. Figure 4 shows the bandwidth returned as the volume was increased from 5 MB to 25 MB,

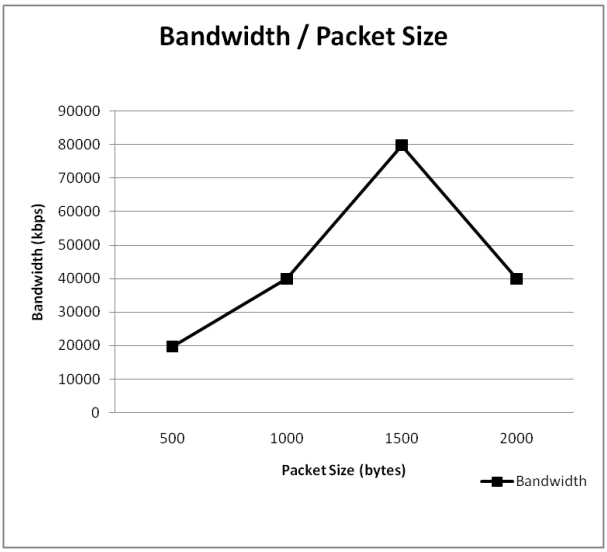

Figure 3. The bandwidth values returned as the packet size is increased

with a constant packet size of 1500 bytes [8].

Figure 6 shows the latency values returned as the latency for a given link was artificially increased using WANem [16]. Latency values returned were an average of 2 ms higher than those set, the extra being introduced by the processing of the WANem gateway. Similarly, Figure 7 shows the reported bandwidth compared with the artificially-set bandwidth using WANem. The reported bandwidth is slightly lower than that set, except for when the set bandwidth is higher than 80 Mbps. At this point the limitations of the 100 Mbps physical link prevent the actual bandwidth reaching that set with WANem.

Given that a CloudState instance runs at each data centre location within a cloud, it is important to calculate the likely impact on a destination node when it is polled by numerous

 ISBN: 978-1-61208-271-4

TABLE II
BANDWIDTH AND LATENCY VALUES RETURNED AS THE NUMBER OF POLLED ADDRESSES IS SCALED UP

No. Targets	Min Lat (ms)	Max Lat (ms)	Mean Lat (ms)	BW (kbps)	Time Taken (secs)
1	< 1	13	< 1	79956	1
2	< 1	13	< 1	79956	1
4	< 1	16	< 1	79887	2
6	< 1	13	< 1	79896	2
8	< 1	14	< 1	79920	1
10	< 1	12	< 1	79992	1
12	< 1	15	< 1	79860	1
14	< 1	13	< 1	79932	1

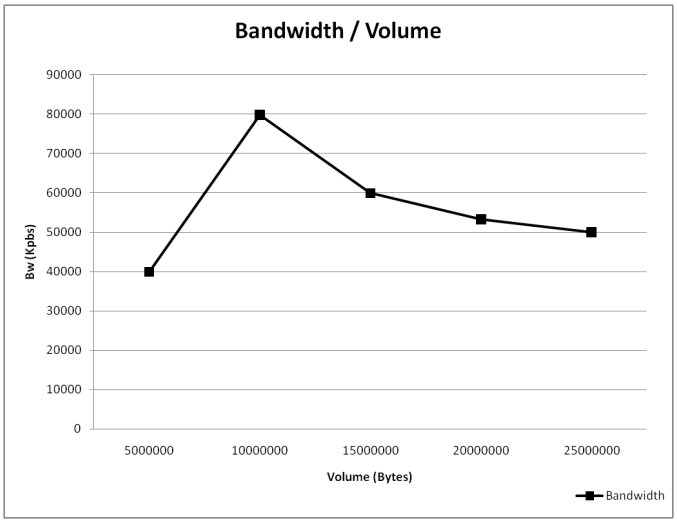

Figure 4. The bandwidth values returned as the data volume is increased

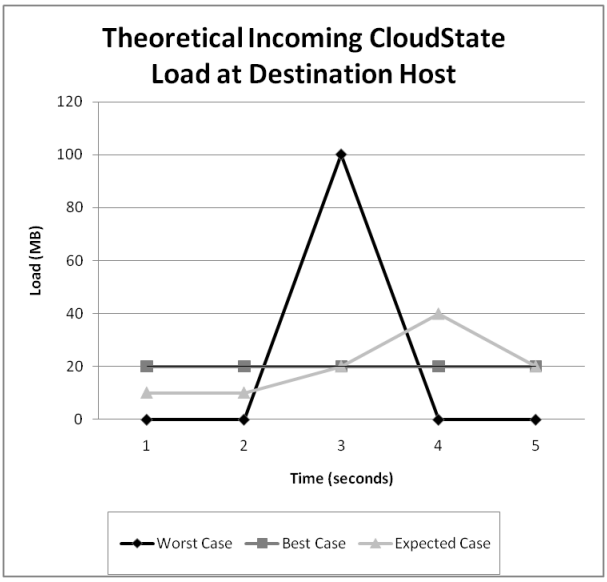

Figure 5. Theoretical impact of CloudState load at a destination address

CloudState instances. The amount of traffic in bytes received by a destination address for each CloudState poll will be:

$$U = \sum_{i=1}^{N} S_i P_i \quad (1)$$

where U is the amount of traffic received by a destination address given i number of CloudState instances communicating with it, each with a given packet size S_i and number of packets transmitted P_i. Figure 5 illustrates theoretical incoming CloudState load trends at a destination host with which 10 CloudState instances communicate, with each CloudState instance transmitting 10 MB of data at intervals of 5 seconds each. Three different situations are possible, one where there is a momentary spike in the incoming load where all of the CloudState instances transmit at the same time. This is the worst case scenario because it is possible that the destination host will encounter a momentary I/O outage. This also will affect both the delivery of the application running on the host as well as the results returned by each CloudState instance. The available bandwidth value will not accurately reflect the average state of the link to that destination host. The best case scenario is one where 2 of the 10 CloudStates poll at the same time and the load is kept almost constant at the destination. The expected case is that there are some spikes in CloudState load but not an "all or nothing" scenario. The effect of the possible CloudState trends on application performance, at the destination host, will differ depending on the amount and the profile of application traffic. It is clear that the poll period for each CloudState should be reduced if it is found to impair the delivery of an application at any host.

V. CONCLUSIONS

Cloud Computing has become a key paradigm in the area of distributed computing. The underlying virtualisation means that, unlike Grid Computing, distributed host node resources are both fragmented for application placement and consolidated to offer more resources to facilitate a resource-hungry application when needed. It is this dynamic, virtual arrangement of resources that causes both a problem in terms of the quality of the WAN link between them at any given time, and an opportunity for intelligent placement of applications such that their WAN needs are satisfied. CloudState provides a low-cost, highly-scalable solution to this problem. Instances can be easily deployed on base systems or within VMs. VM resource usage is low - approximately 712.69 Mhz is used for the CloudState VM under a load of 14 destination addresses. Memory usage equates to 348.16 MB of main memory for the entire VM under the same load of destination addresses. No

 ISBN: 978-1-61208-271-4

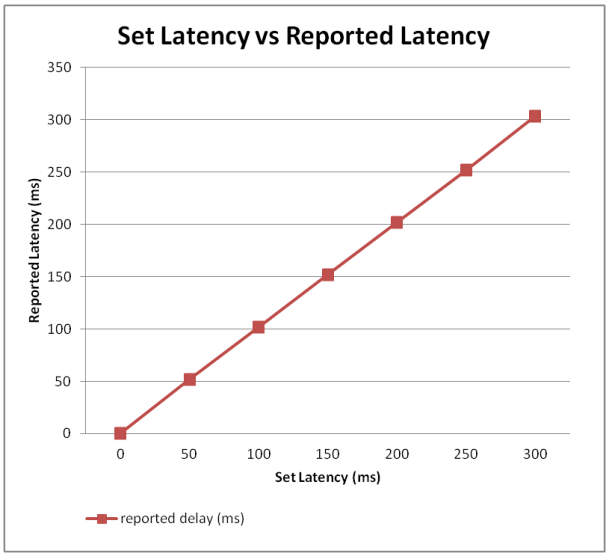

Figure 6. CloudState set latency vs reported latency (ms)

special hardware, servers or clients are required throughout the distributed cloud, except for the requirements that all addresses respond to ICMP echo requests.

A. Further Work

CloudState is currently installed within a Ubuntu 10.10 Linux VM which contains the Gnome desktop environment and a range of applications which are installed with Ubuntu by default. A bare-bones Linux install, e.g., Ubuntu JeOS, would be more suitable for hosting CloudState where unnecessary programs are not installed and the footprint, both in terms of hard-drive space and run-time resource requirements, are minimised. There is an argument for removing the GUI from CloudState and having operational parameters passed as run-time arguments and/or stored in settings files. This would mean that a Linux desktop environment is not required and would make CloudState a very lightweight VM.

CloudState currently transmits a user-defined data volume at each poll. An algorithm should be included that starts off with a small amount of data to be transmitted, which increases with each poll until the bandwidth of the link is established. The volume of data to be transmitted over that link should be recorded and then only periodically checked thereafter. There may be some scope in examining a correlation between the volume of data sent, the link bandwidth and the number of packets dropped at each poll. It is speculated that an overloaded link will discard packets and the volume should be set at the point where a small percentage of packets are dropped. This requires further research.

CloudState may be improved by incorporating a different approach to gathering network performance statistics. Gathering performance data at the application level is possible [10] using Web client probes, especially with a focus on ALTO. This approach would ensure that firewalls are not an issue in gathering performance data and would provide data that may be more accurate, given that it is gathered at the end-user and not at other points on the path between the VM-hosted application and the end-user, e.g., at a router or switch.

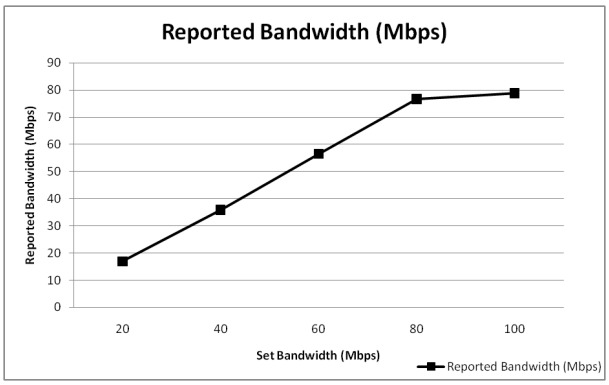

Figure 7. CloudState set bandwidth vs reported bandwidth (Mbps)

REFERENCES

[1] R. Buyya, Chee Shin Yeo, and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference on, pages 5 –13, sept. 2008.

[2] Oleg Derevenetz. Bwping - open source bandwidth measurement tool. http://bwping.sourceforge.net/, June 2011. [retrieved: March. 2013].

[3] GEANT. perfsonar. http://www.perfsonar.net/, 2012. [retrieved: March. 2013].

[4] V.K. Gurbani, M. Scharf, T.V. Lakshman, V. Hilt, and E. Marocco. Abstracting network state in software defined networks (sdn) for rendezvous services. In Communications (ICC), 2012 IEEE International Conference on, pages 6627–6632, june 2012.

[5] Wei Hao, I-Ling Yen, and B. Thuraisingham. Dynamic service and data migration in the clouds. In Computer Software and Applications Conference, 2009. COMPSAC '09. 33rd Annual IEEE International, volume 2, pages 134 –139, july 2009.

[6] Ipanema. Ip engine. http://www.ipanematech.com/en/ip-engine, June 2011. [retrieved: March. 2013].

[7] Iperf. Iperf. online, http://iperf.sourceforge.net/, December 2008. [retrieved: March. 2013].

[8] M. Katevenis, G. Passas, D. Simos, I. Papaefstathiou, and N. Chrysos. Variable packet size buffered crossbar (cicq) switches. In Communications, 2004 IEEE International Conference on, volume 2, pages 1090 – 1096 Vol.2, june 2004.

[9] S. Kiesel, L. Popkin, S. Previdi, R. Woundy, and Y.R. Yang. Application-layer traffic optimization (alto) requirements. Internet Engineering Task Force, Internet-Draft draft-ietf-alto-reqs-00, 2009.

[10] Myung-Sup Kim, Young J Won, and James Won-Ki Hong. Application-level traffic monitoring and an analysis on ip networks. volume 27, pages 22–42, 2005.

[11] Jeffrey S. Klaus. Follow-the-moon scheduling to lower energy costs. http://www.datacenterknowledge.com/archives/2012/10/22/follow-the-moon-scheduling-to-lower-energy-costs/, 2012. [retrieved: March. 2013].

[12] Lijun Mei, W.K. Chan, and T.H. Tse. A tale of clouds: Paradigm comparisons and some thoughts on research issues. In Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE, pages 464 –469, dec. 2008.

[13] Nokia. Qt sdk. http://qt.nokia.com/products/qt-sdk, June 2011.

[14] NPACI. Network weather service. http://nws.cs.ucsb.edu/, 2004. [retrieved: March. 2013].

[15] Optimis. Optimis. http://www.optimis-project.eu/, 2013. [retrieved: March. 2013].

[16] WANem. Wanem - the wide area network emulator. online, http://wanem.sourceforge.net, December 2009. [retrieved: March. 2013].

 ISBN: 978-1-61208-271-4