CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines

Timothy Wood Prashant Shenoy
University of Massachusetts Amherst
{twood,shenoy}@cs.umass.edu

K.K. Ramakrishnan Jacobus Van der Merwe
AT&T Labs - Research
{kkrama,kobus}@research.att.com

Abstract

Virtual machine technology and the ease with which VMs can be migrated within the LAN, has changed the scope of resource management from allocating resources on a single server to manipulating pools of resources within a data center. We expect WAN migration of virtual machines to likewise transform the scope of provisioning compute resources from a single data center to multiple data centers spread across the country or around the world. In this paper we present the CloudNet architecure as a cloud framework consisting of cloud computing platforms linked with a VPN based network infrastructure to provide seamless and secure connectivity between enterprise and cloud data center sites. To realize our vision of efficiently pooling geographically distributed data center resources, CloudNet provides optimized support for live WAN migration of virtual machines. Specifically, we present a set of optimizations that minimize the cost of transferring storage and virtual machine memory during migrations over low bandwidth and high latency Internet links. We evaluate our system on an operational cloud platform distributed across the continental US. During simultaneous migrations of four VMs between data centers in Texas and Illinois, CloudNet's optimizations reduce memory migration time by 65% and lower bandwidth consumption for the storage and memory transfer by 19GB, a 50% reduction.

Categories and Subject Descriptors C.2.4 [*Computer Communication Networks*]: Distributed Systems

General Terms Design, Performance

Keywords WAN migration, Virtualization, Cloud Computing

1. Introduction

Today's enterprises run their server applications in data centers, which provide them with computational and storage resources. Cloud computing platforms, both public and private, provide a new avenue for both small and large enterprises to host their applications by renting resources on-demand and paying based on actual usage. Thus, a typical enterprise's IT services will be spread across the corporation's data centers as well as dynamically allocated resources in cloud data centers.

VEE'11, March 9–11, 2011, Newport Beach, California, USA.

From an IT perspective, it would be ideal if both in-house data centers and private and public clouds could be considered as a flexible pool of computing and storage resources that are seamlessly connected to overcome their geographical separation. The management of such a pool of resources requires the ability to flexibly map applications to different sites as well as the ability to move applications and their data across and within pools. The agility with which such decisions can be made and implemented determines the responsiveness with which enterprise IT can meet changing business needs.

Virtualization is a key technology that has enabled such agility *within* a data center. Hardware virtualization provides a logical separation between applications and the underlying physical server resources, thereby enabling a flexible mapping of virtual machines to servers in a data center. Further, virtual machine platforms support resizing of VM containers to accommodate changing workloads as well as the ability to live-migrate virtual machines from one server to another without incurring application down-times. This same flexibility is also desirable *across* geographically distributed data centers. Such cross data center management requires efficient migration of both memory and disk state between such data centers, overcoming constraints imposed by the WAN connectivity between them. Consider the following use cases that illustrate this need:

Cloud bursting: Cloud bursting is a technique where an enterprise normally employs local servers to run applications and dynamically harnesses cloud servers to enhance capacity during periods of workload stress. The stress on local IT servers can be mitigated by temporarily migrating a few overloaded applications to the cloud or by instantiating new application replicas in the cloud to absorb some of the workload increase. These cloud resources are deallocated once the workload peak has ebbed. Cloud bursting eliminates the need to pre-provision for the peak workload locally, since cloud resources can be provisioned dynamically when needed, yielding cost savings due to the cloud's pay-as-you go model. Current cloud bursting approaches adopt the strategy of spawning new replicas of the application. This limits the range of enterprise applications that may use cloud bursting to stateless applications or those that include elaborate consistency mechanisms. Live migration permits *any* application to exploit cloud bursting while experiencing minimal downtime.

Enterprise IT Consolidation: Many enterprises with multiple data centers have attempted to deal with data center "sprawl" and cut costs by consolidating multiple smaller sites into a few large data centers. Such consolidation requires applications and data to be moved from one site to another over a WAN; a subset of these applications may also be moved to cloud platforms if doing so is more cost-effective than hosting locally. Typically such transformation projects have incurred application down-times, often spread over multiple days. Hence, the ability to implement these moves with

minimal or no down-time is attractive due to the corresponding reduction in the disruption seen by a business.

Follow the sun: "Follow the sun" is a new IT strategy that is designed for project teams that span multiple continents. The scenario assumes multiple groups spanning different geographies that are collaborating on a common project and that each group requires low-latency access to the project applications and data during normal business hours. One approach is to replicate content at each site—e.g., a data center on each continent—and keep the replicas consistent. While this approach may suffice for content repositories or replicable applications, it is often unsuitable for applications that are not amenable to replication. In such a scenario, it may be simpler to migrate one or more VM containers with applications and project data from one site to another every evening; the migration overhead can be reduced by transferring only incremental state and applying it to the snapshot from the previous day to recreate the current state of the application.

These scenarios represent the spectrum from pre-planned to reactive migrations across data centers. Although the abstraction of treating resources that span data centers and cloud providers as a single unified pool of resources is attractive, the reality of these resources being distributed across significant geographic distances and interconnected via static wide area network links (WANs) conspire to make the realization of this vision difficult. Several challenges need to be addressed to realize the above use-cases:

Minimize downtime: Migration of application VMs and their data may involve copying tens of gigabytes of state or more. It is desirable to mask the latency of this data copying overhead by minimizing application downtimes during the migration. One possible solution is to support live migration of virtual machines over a WAN, where data copying is done in the background while the application continues to run, followed by a quick switch-over to the new location. While live migration techniques over LAN are well known, WAN migration raises new challenges, such as the need to migrate disk state in addition to memory state.

Minimize network reconfigurations: Whereas VM migration over a LAN can be performed transparently from a network perspective (IP addresses remains unchanged, TCP connections move over, etc), doing so transparently is a major challenge over a WAN. Different data centers and cloud sites support different IP address spaces, so additional network support is necessary if WAN migration is to remain transparent from a user and application perspective.

Handle WAN links: Migration of virtual machines over a LAN is relatively simple since data center LANs are provisioned using high-speed low-latency links. In contrast, WAN links interconnecting data centers of an enterprise and the connection to cloud sites may be bandwidth-constrained and speed-of-light contraints dictate that inter-data center latencies are significantly higher than in a LAN environment. Even when data centers are inter-connected using well provisioned links, it may not be possible to dedicate hundreds of megabits/s of bandwidth to a *single* VM transfer from one site to another. Further, cloud sites charge for network usage based on the total network I/O from and to cloud servers. Consequently WAN migration techniques must be designed to operate efficiently over low bandwidth links and must optimize the data transfer volume to reduce the migration latency and cost.

In this paper we propose a platform called *CloudNet* to achieve the vision of seamlessly connected resource pools that permit flexible placement and live migration of applications and their data across sites. The design and implementation of CloudNet has resulted in the following contributions.

Network virtualization and Virtual Cloud Pools: We propose a Virtual Cloud Pool (VCP) abstraction which allows CloudNet to seamlessly connect geographically separate servers and provide the illusion of a single logical pool of resources connected over a LAN. VCPs can be thought of as a form of network virtualization where the network identity of a VM can be dynamically (re)bound to a server at any physical site. This minimizes the need for network reconfiguration during WAN migration. CloudNet uses existing VPN technologies to provide this infrastructure, but we present a new signaling protocol that allows endpoint reconfiguration actions that currently take hours or days, to be performed in tens of seconds. This capability is crucial, since scenarios such as Cloud Bursting require rapid reconfiguration of the VCP topology in order to offload local applications to newly instantiated cloud servers.

Live Migration over WANs: CloudNet supports live migration of virtual machines over WANs. There are two key differences between LAN-based live migration and WAN-based migration. First, live migration over LAN only moves memory state, since disk state is assumed to be stored on a storage area network. In contrast, WAN migration may need to move both memory and disk state of an application if the Storage Area Network (SAN) does not span multiple data center sites. Second, LAN VM migration is transparent to an application from a network standpoint. In contrast, WAN-based VM migration must coordinate with the network routers to implement a similar level of transparency. CloudNet includes a storage migration mechanism and leverages its dynamic VCP abstraction to support transparent VM migration over WANs.

WAN Optimizations: CloudNet implements several WAN optimizations to enable migration over low-bandwidth links. It implements an adaptive live migration algorithm that dynamically tailors the migration of memory state based on application behavior. It also implements mechanisms such as content-based redundancy elimination and page deltas into the hypervisor to reduce the data volume sent during the migration process. Collectively these optimizations minimize total migration time, application downtime, and volume of data transferred.

Prototyping and Experimentation across multiple data centers: We implement a prototype of Cloudnet using the Xen platform and a commercial layer-2 VPN implementation. We perform an extensive evaluation using three data centers spread across the continental United States. Our results show CloudNet's optimizations decreasing memory migration and pause time by 30 to 70% in typical link capacity scenarios; in a set of VM migrations over a distance of 1200km, CloudNet saves 20GB of bandwidth, a 50% reduction. We also evaluate application performance during migrations to show that CloudNet's optimizations reduce the window of decreased performance compared to existing techniques.

2. Cloudnet Overview

In this section, we present an overview of the key abstractions and design building blocks in CloudNet.

2.1 Resource Pooling: Virtual Cloud Pools

At the heart of CloudNet is a Virtual Cloud Pool (VCP) abstraction that enables server resources across data centers and cloud providers to be logically grouped into a single server pool as shown in Figure 1. The notion of a Virtual Cloud Pool is similar to that of a Virtual Private Cloud, which is used by Amazon's EC2 platform and was also proposed in our previous research [30]. Despite the similarity, the design motivations are different. In our case, we are concerned with grouping server pools across data centers, while Amazon's product seeks to provide the abstraction of a private cloud that is hosted on a public cloud. Both abstractions use virtual private networks (VPNs) as their underlying interconnec-

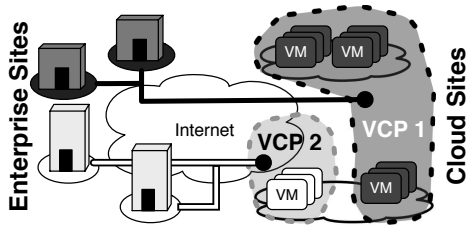


Figure 1. Two VCPs isolate resources within the cloud sites and securely link them to the enterprise networks.

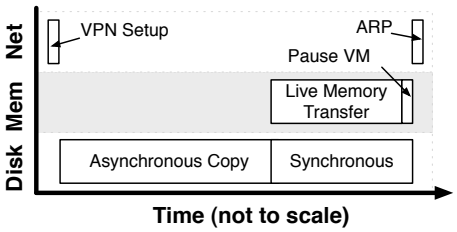


Figure 2. The phases of a migration for non-shared disk, memory, and the network in CloudNet .

tion technology—we employ Layer 2 VPNs to implement a form of network virtualization/transparency, while Amazon's VPC uses layer 3 VPNs to provide control over the network addressing of VM services.

The VCPs provided by CloudNet allow cloud resources to be connected to as securely and seamlessly as if they were contained within the enterprise itself. Further, the cloud to enterprise mappings can be adjusted dynamically, allowing cross data center resource pools to grow and change depending on an enterprise's needs. In the following sections we discuss the benefits of these abstractions for enterprise applications and discuss how this dynamic infrastructure facilitates VM migration between data centers.

2.2 Dynamic, Seamless Cloud Connections

CloudNet uses Multi-Protocol Label Switching (MPLS) based VPNs to create the abstraction of a private network and address space shared by multiple data centers. Since addresses are specific to a VPN, the cloud operator can allow customers to use any IP address ranges that they prefer without concern for conflicts between cloud customers. CloudNet makes the level of abstraction even greater by using *Virtual Private LAN Services (VPLS)* that bridge multiple MPLS endpoints onto a single LAN segment. This allows cloud resources to appear *indistinguishable* from existing IT infrastructure already on the enterprise's own LAN. VPLS provides transparent, secure, and resource guaranteed layer-2 connectivity without requiring sophisticated network configuration by end users. This simplifies the network reconfiguration that must be performed when migrating VMs between data centers.

VPNs are already used by many large enterprises, and cloud sites can be easily added as new secure endpoints within these existing networks. VCPs use VPNs to provide secure communication channels via the creation of "virtually dedicated" paths in the provider network. The VPNs protects traffic between the edge routers at each enterprise and cloud site. Within a cloud site, the traffic for a given enterprise is restricted to a particular VLAN. This provides a secure end-to-end path from the enterprise to the cloud and eliminates the need to configure complex firewall rules between the cloud and the enterprise, as all sites can be connected via a private network inaccessible from the public Internet.

As enterprises deploy and move resources between cloud data centers, it is necessary to adjust the topology of the client's VCP. In typical networks, connecting a new data center to a VPN endpoint can take hours or days, but these delays are administrative rather than fundamental to the network operations required. CloudNet utilizes a VPN Controller to automate the process of VPN reconfiguration, allowing resources at a new cloud data center to be connected to a VPN within seconds.

2.3 Efficient WAN Migration

Currently, moving an application to the cloud or another data center can require substantial downtime while application state is copied and networks are reconfigured before the application can resume operation. Alternatively, some applications can be easily replicated into the cloud while the original continues running; however, this only applies to a small class of applications (e.g. stateless web servers or MapReduce style data processing jobs). These approaches are insufficient for the majority of enterprise applications which have not been designed for ease of replication. Further, many legacy applications can require significant reconfiguration to deal with the changed network configuration required by current approaches. In contrast, the live VM migration supported by CloudNet provides a much more attractive mechanism for moving applications between data centers because it is completely application independent and can be done with only minimal downtime.

Most recent virtualization platforms support efficient migration of VMs within a local network [9, 21]. By virtue of presenting WAN resources as LAN resources, CloudNet's VCP abstraction allows these live migration mechanisms to function unmodified across data centers separated by a WAN. However, the lower bandwidth and higher latencies over WAN links result in poor migration performance. In fact, VMWare's preliminary support for WAN VM migration requires at least 622 Mbps of bandwidth dedicated to the transfer, and is designed for links with less than 5 msec latency [29]. Despite being interconnected using "fat" gigabit pipes, data centers will typically be unable to dedicate such high bandwidth for a *single* application transfer and enterprises will want the ability to migrate a group of related VMs concurrently. CloudNet uses a set of optimizations to conserve bandwidth and reduce WAN migration's impact on application performance.

Current LAN-based VM migration techniques assume the presence of a shared file system which enables them to migrate only memory data and avoid moving disk state. A shared file system may not always be available across a WAN or the performance of the application may suffer if it has to perform I/O over a WAN. Therefore, CloudNet coordinates the hypervisor's memory migration with a disk replication system so that the entire VM state can be transferred if needed.

Current LAN-based live migration techniques must be optimized for WAN environments, and cloud computing network infrastructure must be enhanced to support dynamic relocation of resources between cloud and enterprise sites; these challenges are the primary focus of this paper.

3. WAN VM Migration

Consider an organization which desires to move one or more applications (and possibly their data) between two data centers. Each application is assumed to be run in a VM, and we wish to live migrate those virtual machines across the WAN.

CloudNet uses these steps to live migrate each VM:

Step 1: Establish virtual connectivity between VCP endpoints.

Step 2: If storage is not shared, transfer all disk state.

Step 3: Transfer the memory state of the VM to a server in the destination data center as it continues running without interruption.

Step 4: Once the disk and memory state have been transferred,

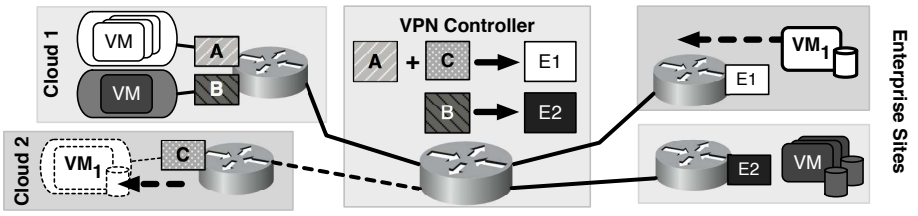


Figure 3. The VPN Controller remaps the route targets (A,B,C) advertised by each cloud data center to match the proper enterprise VPN (E1 or E2). To migrate VM_1 to Cloud Site 2, the VPN controller redefines E1's VPN to include route target A and C, then performs the disk and memory migration.

briefly pause the VM for the final transition of memory and processor state to the destination host. This process must not disrupt any active network connections between the application and its clients.

While these steps, illustrated in Figure 2, are well understood in LAN environments, migration over the WAN poses new challenges. The constraints on bandwidth and the high latency found in WAN links makes steps 2 and 3 more difficult since they involve large data transfers. The IP address space in step 4 would typically be different when the VM moves between routers at different sites. Potentially, application, system, router and firewall configurations would need to be updated to reflect this change, making it difficult or impossible to seamlessly transfer active network connections. CloudNet avoids this problem by virtualizing the network connectivity so that the VM appears to be on the same virtual LAN. We achieve this using VPLS VPN technology in step 1, and CloudNet utilizes a set of migration optimizations to improve performance in the other steps.

3.1 Migrating Networks, Disk, and Memory

Here we discuss the techniques used in CloudNet to transfer disk and memory, and to maintain network connectivity throughout the migration. We discuss further optimizations to these approaches in Section 3.2.

3.1.1 Dynamic VPN Connectivity to the Cloud

A straightforward implementation of VM migration between IP networks results in significant network management and configuration complexity [14]. As a result, *virtualizing network connectivity is key in CloudNet for achieving the task of WAN migration seamlessly relative to applications*. However, reconfiguring the VPNs that CloudNet can take advantage of to provide this abstraction has typically taken a long time because of manual (or nearly manual) provisioning and configuration. CloudNet explicitly recognizes the need to set up new VPN endpoints quickly, and exploits the capability of BGP route servers [28] to achieve this.

In many cases, the destination data center will already be a part of the customer's virtual cloud pool because other VMs owned by the enterprise are already running there. However, if this is the first VM being moved to the site, then a new VPLS endpoint must be created to extend the VCP into the new data center.

Creating a new VPLS endpoint involves configuration changes on the data center routers, a process that can be readily automated on modern routers [8, 20]. A traditional, but naive, approach would require modifying the router configurations at each site in the VCP so they all advertise and accept the proper *route targets*. A route target is an ID used to determine which endpoints share connectivity. An alternative to adjusting the router configurations directly, is to dynamically adjust the routes advertised by each site within the network itself. CloudNet takes this approach by having data center routers announce their routes to a centralized VPN Controller. The VPN Controller acts as an intelligent route server and is connected via BGP sessions to each of the cloud and enterprise data centers. The controller maintains a ruleset indicating which endpoints should have connectivity; as all route control messages pass through this VPN Controller, it is able to rewrite the route targets in these messages, which in turn control how the tunnels forming each VPLS are created. Figure 3 illustrates an example where VM_1 is to be migrated from enterprise site E1 to Cloud Site 2. The VPN Controller must extend E1's VPLS to include route targets A and C, while Enterprise 2's VPLS only includes route target B. Once the change is made by the VPN Controller, it is propagated to the other endpoints via BGP. This ensures that each customer's resources are isolated within their own private network, providing CloudNet's virtual cloud pool abstraction. Likewise, the VPN Controller is able to set and distribute fine grained access control rules via BGP using technologies such as Flowspec (RFC 5575).

Our approach allows for fast VCP reconfiguration since changes only need to be made at a central location and then propagated via BGP to all other sites. This provides simpler connectivity management compared to making changes individually at each site, and allows a centralized management scheme that can set connectivity and access control rules for all sites.

In our vision for the service, the VPN Controller is operated by the network service provider. As the VPLS network in CloudNet spans both the enterprise sites and cloud data centers, the cloud platform must have a means of communicating with the enterprise's network operator. The cloud platform needs to expose an interface that would inform the network service provider of the ID for the VLAN used within the cloud data center so that it can be connected to the appropriate VPN endpoint. Before the VPN Controller enables the new endpoint, it must authenticate with the cloud provider to ensure that the enterprise customer has authorized the new resources to be added to its VPN. These security details are orthogonal to our main work, and in CloudNet we assume that there is a trusted relationship between the enterprise, the network provider, and the cloud platform.

3.1.2 Disk State Migration

LAN based live migration assumes a shared file system for VM disks, eliminating the need to migrate disk state between hosts. As this may not be true in a WAN environment, CloudNet supports either shared disk state or a replicated system that allows storage to be migrated with the VM.

If we have a "shared nothing" architecture where VM storage must be migrated along with the VM memory state, CloudNet uses the DRBD disk replication system to migrate storage to the destination data center [11]. In Figure 3, once connectivity is established to Cloud 2, the replication system must copy VM_1's disk to the remote host, and must continue to synchronize the remote disk with any subsequent writes made at the primary. In order to reduce the performance impact of this synchronization, CloudNet uses DRBD's *asynchronous* replication mode during this stage. Once the remote disk has been brought to a consistent state, CloudNet switches to a *synchronous* replication scheme and the live migration of the VM's

memory state is initiated. During the VM migration, disk updates are synchronously propagated to the remote disk to ensure consistency when the memory transfer completes. When the migration completes, the new host's disk becomes the primary, and the origin's disk is disabled.

Migrating disk state typically represents the largest component of the overall migration time as the disk state may be in the tens or hundreds of gigabytes. Fortunately, disk replication can be enabled well in advance of a planned migration. Since the disk state for many applications changes only over very long time scales, this can allow the majority of the disk to be transferred with relatively little wasted resources (e.g., network bandwidth). For unplanned migrations such as a cloud burst in response to a flash crowd, storage may need to be migrated on demand. CloudNet's use of asynchronous replication during bulk disk transfer minimizes the impact on application performance.

3.1.3 Transferring Memory State

Most VM migration techniques use a "pre-copy" mechanism to iteratively copy the memory contents of a live VM to the destination machine, with only the modified pages being sent during each iteration [9, 21]. At a certain point, the VM is paused to copy the final memory state. WAN migration can be accomplished by similar means, but the limited bandwidth and higher latencies can lead to decreased performance–particularly much higher VM downtimes–since the final iteration where the VM is paused can last much longer. CloudNet augments the existing migration code from the Xen virtualization platform with a set of optimizations that improve performance, as described in Section 3.2.

The amount of time required to transfer a VM's memory depends on its RAM allocation, working set size and write rate, and available bandwidth. These factors impact both the total time of the migration, and the application-experienced downtime caused by pausing the VM during the final iteration. With WAN migration, it is desirable to minimize both these times as well as the bandwidth costs for transferring data. While pause time may have the most direct impact on application performance, the use of synchronous disk replication throughout the memory migration means that it is also important to minimize the total time to migrate memory state, particularly in high latency environments.

As bandwidth reduces, the total time and pause time incurred by a migration can rise dramatically. Figure 4(a) shows the pause time of VMs running several different applications, as the available bandwidth is varied (assumes shared storage and a constant 10 msec round trip latency). Note that performance decreases non-linearly; migrating a VM running the SPECjbb benchmark on a gigabit link incurs a pause time of 0.04 seconds, but rises to 7.7 seconds on a 100 Mbps connection. This nearly 200X increase is unacceptable for most applications, and happens because a migration across a slower link causes each iteration to last longer, increasing the chance that additional pages will be modified and thus need to be resent. This is particularly the case in the final iteration. This result illustrates the importance of optimizing VM migration algorithms to better handle low bandwidth connections.

Migrations over the WAN may also have a greater chance of being disrupted due to network failures between the source and destination hosts. Because the switch-over to the second site is performed only after the migration is complete, CloudNet will suffer no ill effects from this type of failure because the application will continue running on the origin site, unaffected. In contrast, some pull or "post-copy" based migration approaches start running the application at the destination prior to receiving all data, which could lead to the VM crashing if there is a network disconnection.

3.1.4 Maintaining Network Connections

Once disk and memory state have been migrated, CloudNet must ensure that VM_1's active network connections are redirected to Cloud 2. In LAN migration, this is achieved by having the destination host transmit an unsolicited ARP message that advertises the VM's MAC and IP address. This causes the local Ethernet switch to adjust the mapping for the VM's MAC address to its new switch port [9]. Over a WAN, this is not normally a feasible solution because the source and destination are not connected to the same switch. Fortunately, CloudNet's use of VPLS bridges the VLANs at Cloud 2 and E1, causing the ARP message to be forwarded over the Internet to update the Ethernet switch mappings at both sites. This allows open network connections to be seamlessly redirected to the VM's new location.

In the Xen virtualization platform, this ARP is triggered by the VM itself after the migration has completed. In CloudNet, we optimize this procedure by having the destination host preemptively send the ARP message immediately after the VM is paused for the final iteration of the memory transfer. This can reduce the downtime experienced by the VM by allowing the ARP to propagate through the network in parallel with the data sent during the final iteration. However, on our evaluation platform this does not appear to influence the downtime, although it could be useful with other router hardware since some implementations can cache MAC mappings rather than immediately updating them when an ARP arrives.

3.2 Optimizing WAN VM Migration

In this section we propose a set of optimizations to improve the performance of VM migration over the WAN. The changes are made within the virtualization hypervisor; while we use the Xen virtualization platform in our work [9], they would be equally useful for other platforms such as VMWare which uses a similar migration mechanism [21].

3.2.1 Smart Stop and Copy

The Xen migration algorithm typically iterates until either a very small number of pages remain to be sent or a limit of 30 iterations is reached. At that point, the VM is paused, and all remaining pages are sent. However, our results indicate that this tends to cause the migration algorithm to run through many unnecessary iterations, increasing both the total time for the migration and the amount of data transferred.

Figure 4(b) shows the number of pages remaining to be sent at the end of each iteration during a migration of a VM running a kernel compilation over a link with 622 Mbps bandwidth and 5 msec latency. After the fourth iteration there is no significant drop in the number of pages remaining to be sent. This indicates that (i) a large number of iterations only extends the total migration time and increases the data transferred, and (ii) the migration algorithm could intelligently pick when to stop iterating in order to decrease both total and pause time. For the migration shown, picking the optimal point to stop the migration would reduce pause time by 40% compared to the worst stopping point.

CloudNet uses a *Smart Stop and Copy* optimization to reduce the number of unnecessary iterations and to pick a stopping point that minimizes pause time. Unfortunately, these two goals are potentially conflicting. Stopping after only a few iterations would reduce *total time*, but running for an extra few rounds may result in a lower *pause time*, which can potentially have a larger impact on application performance. The Smart Stop algorithm is designed to balance this trade-off by minimizing pause time without significantly increasing total time.

We note that in most cases (e.g. Figure 4(b)), after about five iterations the migration reaches a point of diminishing returns, where in a given iteration, approximately the same amount of data

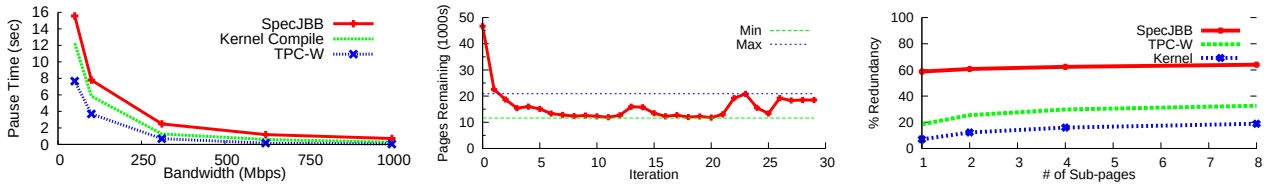


Figure 4. (a) Low bandwidth links can significantly increase the downtime experienced during migration. (b) The number of pages to be sent quickly levels off. Intelligently deciding when to stop a migration eliminates wasteful transfers and can lower pause time. (c) Each application has a different level of redundancy. Using finer granularity finds more redundancy, but has diminishing returns.

is dirtied as is sent. To detect this point, the first stage of Smart Stop monitors the number of pages sent and dirtied until they become equal. Prior to this point there was a clear gain from going through another iteration because more data was sent than dirtied, lowering the potential pause time.

While it is possible to stop the migration immediately at the point where as many pages are dirtied as sent, we have found that often the random fluctuations in how pages are written to can mean that waiting a few more iterations can result in a lower pause time with only a marginal increase in total time. Based on this observation, Smart Stop switches mode once it detects this crossover, and begins to search for a local minimum in the number of pages remaining to be sent. If at the start of an iteration, the number of pages to be sent is less than any previous iteration in a sliding window, Smart Stop pauses the VM to prevent any more memory writes and sends the final iteration of memory data.

3.2.2 Content Based Redundancy

Content based redundancy (CBR) elimination techniques have been used to save bandwidth between network routers [2], and we use a similar approach to eliminate the redundant data while transferring VM memory and disk state.[1] Disks can have large amounts of redundant data caused by either empty blocks or similar files. Likewise, a single virtual machine can often have redundant pages in memory from similar applications or duplicated libraries.

There are a variety of mechanisms that can be used to eliminate redundancy in a network transfer, and a good comparison of techniques is found in [1]. CloudNet can support any type of redundancy elimination algorithm; for efficiency, we use a block based approach that detects identical, fixed size regions in either a memory page or disk block. We have also tested a Rabin Fingerprint based redundancy elimination algorithm, but found it to be slower without substantially improving the redundancy detection rate.

CloudNet's block based CBR approach splits each memory page or disk block into fixed sized blocks and generates hashes based on their content using the Super Fast Hash Algorithm [16]. If a hash matches an entry in fixed size, FIFO caches maintained at the source and destination hosts, then a block with the same contents was sent previously. After verifying the pages match (in case of hash collisions), the migration algorithm can simply send a 32bit index to the cache entry instead of the full block (e.g. 4KB for a full memory page).

Dividing a memory page into smaller blocks allows redundant data to be found with finer granularity. Figure 4(c) shows the amount of memory redundancy found in several applications during migrations over a 100 Mbps link as the number of blocks per page was varied. Increasing the number of sub-pages raises the level of redundancy that is found, but it can incur greater overhead since each block requires a hash table lookup. In CloudNet we divide each page into four sub-pages since this provides a good trade-off of detection rate versus overhead.

Disk transfers can also contain large amounts of redundant data. Our redundancy elimination code is not yet fully integrated with DRBD, however, we are able to evaluate the potential benefit of this optimization by analyzing disk images with an offline CBR elimination tool.

We currently only detect redundancy within a single VM's memory or disk. Previous work has demonstrated that different virtual machines often have some identical pages in memory, e.g. for common system libraries [13, 31]. Likewise, different virtual machines often have large amounts of identical data on disk due to overlap in the operating system and installed applications. Some of this redundancy could be found by using a network based appliance to detect redundancy across the migration traffic of multiple virtual machines. However, a network based approach can only find a redundant disk or memory block if it matches a packet sent during a previous migration. In order to find redundancy in the disks or memories of VMs which are not being moved, such an approach could be complemented with a distributed, content addressable cache run across the hosts at each site [22]. Fortunately, the single VM redundancy detection technique used in CloudNet is still able to save a significant amount of bandwidth without this added complexity.

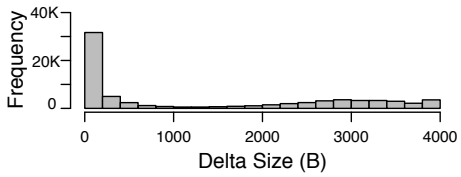
3.2.3 Using Page Deltas

After the first iteration, most of the pages transferred are ones which were sent previously, but have since been modified. Since an application may be modifying only portions of pages, another approach to reduce bandwidth consumption is to keep a cache of previously transmitted pages, and then only send the difference between the cached and current page if it is retransmitted. This technique has been demonstrated in the Remus high availability system to reduce the bandwidth required for VM synchronization [10] in a LAN. We enhance this type of communicating deltas in a unique manner by complementing it with our CBR optimization. This combination helps overcome the performance limitations that would otherwise constrain the adoption of WAN migration

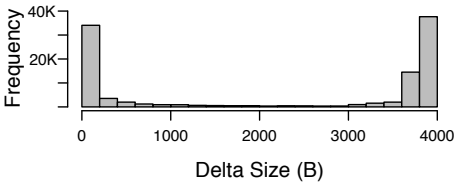
We have modified the Xen migration code so that if a page, or sub page block, does not match an entry in the cache using the CBR technique described previously, then the page address is used as a secondary index into the cache. If the page was sent previously, then only the difference between the current version and the stored version of the page is sent. This delta is calculated by XOR'ing the current and cached pages, and run length encoding the result.

Figure 5 shows histograms of delta sizes calculated during migrations of two applications. A smaller delta means less data needs

[1] Commercial products such as those from RiverBed Technologies can also perform CBR using a transparent network appliance. Such products may not be suitable in our case since memory and/or disk migration data is likely to use encryption to avoid interception of application state. In such cases, end-host based redundancy elimination has been proposed as an alternative [1]—an approach we use here also.



(a) Kernel Compile



(b) TPC-W

Figure 5. During a kernel compile, most pages only experience very small modifications. TPC-W has some pages with small modifications, but other pages are almost completely changed.

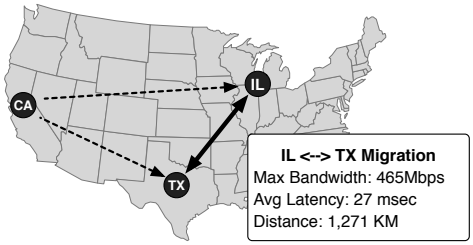


Figure 6. Our CloudNet testbed is deployed across three data centers. Migrations are performed between the data centers in IL and TX, with application clients running in CA.

to be sent; both applications have a large number of pages with only small modifications, but TPC-W also has a collection of pages that have been completely modified. This result suggests that page deltas can reduce the amount of data to be transferred by sending only the small updates, but that care must be taken to avoid sending deltas of pages which have been heavily modified.

While it is possible to perform some WAN optimizations such as redundancy elimination in network middleboxes [2], the Page Delta optimization relies on memory page address information that can only be obtained from the hypervisor. As a result, we make all of our modifications within the virtualization and storage layers. This requires no extra support from the network infrastructure and allows a single cache to be used for both redundancy elimination and deltas. Further, VM migrations are typically encrypted to prevent eavesdroppers from learning the memory contents of the VM being migrated, and network level CBR generally does not work over encrypted streams [1]. Finally, we believe our optimization code will be a valuable contribution back to the Xen community.

4. Evaluation

This section evaluates the benefits of each of our optimizations and studies the performance of several different application types during migrations between data center sites under a variety of network conditions. We also study migration under the three use case scenarios described in the introduction: Section 4.4 illustrates a cloud burst, Section 4.8 studies multiple simultaneous migrations as part of a data center consolidation effort, and Section 4.9 looks at the cost of disk synchronization in a follow-the-sun scenario.

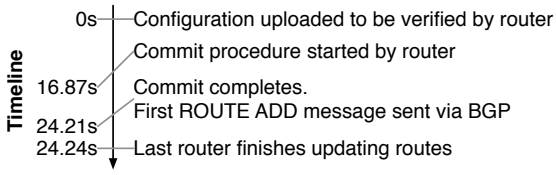


Figure 7. Timeline of operations to add a new endpoint.

4.1 Testbed Setup

We have evaluated our techniques between three data center sites spread across the United States, and interconnected via an operational network, as well as on a laboratory testbed that uses a network emulator to mimic a WAN environment.

Data Center Prototype: We have deployed CloudNet across three data centers in Illinois, Texas, and California as shown in Figure 6. Our prototype is run on top of the ShadowNet infrastructure which is used by CloudNet to configure a set of logical routers located at each site [7]. At each site we have Sun servers with dual quad-core Xeon CPUs and 32GB of RAM. We use Juniper M7i routers to create VPLS connectivity between all sites. We use the California site to run application clients, and migrate VMs between Texas and Illinois. Network characteristics between sites are variable since the data centers are connected via the Internet; we measured an average round trip latency of 27 msec and a max throughput of 465 Mbps between the sites used for migrations.

Lab Testbed: Our lab testbed consists of multiple server/router pairs linked by a VPLS connection. The routers are connected through gigabit ethernet to a PacketSphere Network Emulator capable of adjusting the bandwidth, latency, and packet loss experienced on the link. We use this testbed to evaluate WAN migrations under a variety of controlled network conditions.

4.2 Applications and Workloads

Our evaluation studies three types of business applications. We run each application within a Xen VM and allow it to warm up for at least twenty minutes prior to migration.

SPECjbb 2005 is a Java server benchmark that emulates a client/server business application [24]. The majority of the computation performed is for the business logic performed at the application's middle tier. SPECjbb maintains all application data in memory and only minimal disk activity is performed during the benchmark.

Kernel Compile represents a development workload. We compile the Linux 2.6.31 kernel along with all modules. This workload involves moderate disk reads and writes, and memory is mainly used by the page cache. In our simultaneous migration experiment we run a compilation cluster using *distcc* to distribute compilation activities across several VMs that are all migrated together.

TPC-W is a web benchmark that emulates an Amazon.com like retail site [26]. We run TPC-W in a two tier setup using Tomcat 5.5 and MySQL 5.0.45. Both tiers are run within a single VM. Additional servers are used to run the client workload generators, emulating 600 simultaneous users accessing the site using the "shopping" workload that performs a mix of read and write operations. The TPC-W benchmark allows us to analyze the client perceived application performance during the migration, as well as verify that active TCP sessions do not reset during the migration.

4.3 VPN Endpoint Manipulation

Before a migration can begin, the destination site may need to be added to the customer's VPN. This experiment measures the time required for CloudNet's VPN Controller to add the third data center site to our Internet-based prototype by manipulating route targets.

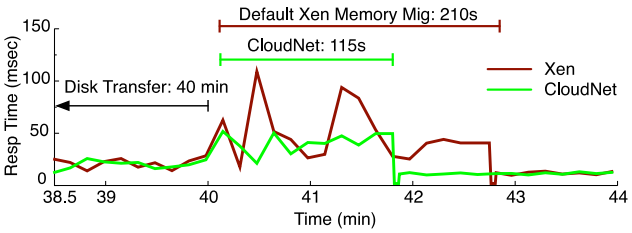


Figure 8. Response times rise to an average of 52 msec during the memory migration, but CloudNet shortens this period of reduced performance by 45%. Response time drops to 10msec once the VM reaches its destination and can be granted additional resources.

Figure 7 shows a timeline of the steps performed by the VPN Controller to reconfigure its intelligent route server. The controller sends a series of configuration commands followed by a commit operation to the router, taking a total of 24.21s to be processed on our Juniper M7i routers; these steps are manufacturer dependent and may vary depending on the hardware. As the intelligent route server does not function as a general purpose router, it would be possible to further optimize this process if reduction in VPN reconfiguration time is required.

Once the new configuration has been applied to the router maintained by the VPN controller, the updated information must be propagated to the other routers in the network. The information is sent in parallel via BGP. On our network where three sites need to have their routes updated, the process completes in only 30 milliseconds, which is just over one round trip time. While propagating routes may take longer in larger networks, the initial intelligent route server configuration steps will still dominate the total cost of the operation.

4.4 Cloud Burst: Application Performance

Cloud Bursting allows an enterprise to offload computational jobs from its own data centers into the cloud. Current cloud bursting techniques require applications to be shut down in the local site and then restarted in the cloud; the live WAN migration supported by CloudNet allows applications to be seamlessly moved from an enterprise data center into the cloud.

We consider a cloud bursting scenario where a live TPC-W web application must be moved from an overloaded data center in Illinois to one in Texas without disrupting its active clients; we migrate the VM to a more powerful server and increase its processor allocation from one to four cores once it arrives at the new data center location. In a real deployment a single VM migration would not have access to the full capacity of the link between the data centers, so we limit the bandwidth available for the migration to 85Mbps; the VM is allocated 1.7GB of RAM and has a 10GB disk. We assume that CloudNet has already configured the VPN endpoint in Texas as described in the previous section. After this completes, the DRBD subsystem begins the initial bulk transfer of the virtual machine disk using asynchronous replication; we discuss the disk migration performance details in Section 4.5 and focus on the application performance during the memory migration here.

The full disk transfer period takes forty minutes and is then followed by the memory migration. Figure 8 shows how the response time of the TPC-W web site is affected during the final 1.5 minutes of the storage transfer and during the subsequent memory migration when using both default Xen and CloudNet with all optimizations enabled. During the disk transfer period, the asynchronous replication imposes negligible overhead; average response time is 22 msec compared to 20 msec prior to the transfer. During the VM migration itself, response times become highly variable, and the average rises 2.5X to 52 msec in the default Xen case. This overhead is primarily caused by the switch to synchronous disk replication—any web request which involves a write to the database will see its response time increased by at least the round trip latency (27 msec) incurred during the synchronous write. As a result, it is very important to minimize the length of time for the memory migration in order to reduce this period of lower performance. After the migration completes, the response time drops to an average of 10 msec in both cases due to the increased capacity available for the VM.

	Data Tx (GB)	Total Time (s)	Pause Time (s)
TPC-W	1.5 → 0.9	135 → 78	3.7 → 2.3
Kernel	1.5 → 1.1	133 → 101	5.9 → 3.5
SPECjbb	1.2 → 0.4	112 → 35	7.8 → 6.5

Table 1. CloudNet reduces bandwidth, total time, and pause time during migrations over a 100Mbps link with shared disk.

While both default Xen and CloudNet migrations do suffer a performance penalty during the migration, CloudNet's optimizations reduce the memory migration time from 210 to 115 seconds, a 45% reduction. CloudNet also lowers the downtime by half, from 2.2 to 1 second. Throughout the migration, CloudNet's memory and disk optimizations conserve bandwidth. Using a 100MB cache, CloudNet reduces the memory state transfer from 2.2GB to 1.5GB. Further, the seamless network connectivity provided by the CloudNet infrastructure prevents the need for any complicated network reconfiguration, and allows the application to continue communicating with all connected clients throughout the migration. This is a significant improvement compared to current cloud bursting techniques which typically cause lengthy downtime as applications are shutdown, replicated to the second site, and then rebooted in their new location.

4.5 Disk Synchronization

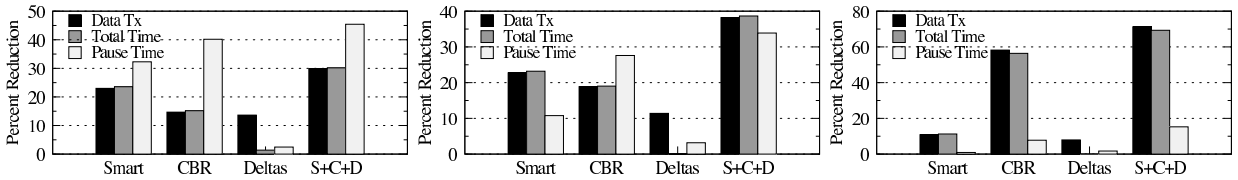
Storage migration can be the dominant cost during a migration in terms of both time and bandwidth consumption. The DRBD system used by CloudNet transfers disk blocks to the migration destination by reading through the source disk at a constant rate (4MB/s) and transmitting the non-empty blocks. This means that while the TPC-W application in the previous experiment was allocated a 10GB disk, only 6.6GB of data is transferred during the migration.

The amount of storage data sent during a migration can be further reduced by employing redundancy elimination on the disk blocks being transferred. Using a small 100MB redundancy elimination cache can reduce the transfer to 4.9GB, and a larger 1GB cache can lower the bandwidth consumption to only 3.6GB. Since the transfer rate is limited by the disk read speed, disk migration takes the same amount of time with and without CloudNet's optimizations; however, the use of content based redundancy significantly reduces bandwidth costs during the transfer.

4.6 Memory Transfer

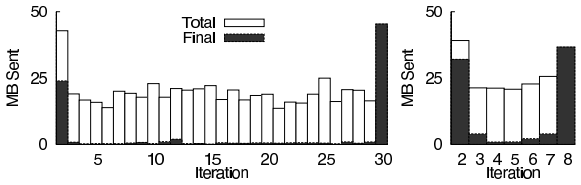
Here we discuss the benefits provided by each of our optimizations for transferring memory state. To understand each optimization's contribution, we analyze migration performance using VMs allocated 1GB of RAM running each of our three test applications; we create the VMs on a shared storage device and perform the migrations over a 100 Mbps link with 20 msec RTT in our local testbed.

Figure 9 shows each of CloudNet's optimizations enabled individually and in combination. We report the average improvement in total time, pause time, and data transferred over four repeated migrations for each optimization. Overall, the combination of all optimizations provides a 30 to 70 percent reduction in the amount of data transferred and total migration time, plus up to a 50% reduction in pause time. Table 1 lists the absolute performance of migrations with the default Xen code and with CloudNet's optimizations.



(a) Kernel Compile (b) TPC-W (c) SPECjbb

Figure 9. CloudNet's optimizations affect different classes of application differently depending on the nature of their memory accesses. Combining all optimizations greatly reduces bandwidth consumption and time for all applications.



(a) Default Xen (b) Smart Stop

Figure 10. Smart Stop reduces the iterations in a migration, significantly lowering the number of "useless" page transfers that end up needing to be retransmitted in the default case.

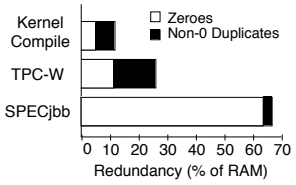


Figure 11. Different applications have different levels of redundancy, in some cases mostly from empty zero pages.

Smart Stop: The Smart Stop optimization can reduce the data transferred and total time by over 20% (Figure 9). Using Smart Stop lowers the number of iterations from 30 to an average of 9, 7, and 10 iterations for Kernel Compile, TPC-W, and SPECjbb respectively. By eliminating the unnecessary iterations, Smart Stop saves bandwidth and time.

Smart Stop is most effective for applications which have a large working set in memory. In TPC-W, memory writes are spread across a database, and thus it sees a large benefit from the optimization. In contrast, SPECjbb repeatedly updates a smaller region of memory, and these updates occur fast enough that the migration algorithm defers those pages until the final iteration. As a result, only a small number of pages would have been sent during the intermediate iterations that Smart Stop eliminates.

Figure 10 shows the total number of pages sent in each iteration, as well as how much of the data is *final*–meaning it does not need to be retransmitted in a later iteration–during a TPC-W migration. After the second iteration, TPC-W sends over 20MB per iteration, but only a small fraction of the total data sent is final–the rest is resent in later iterations when pages are modified again. Smart Stop eliminates these long and unnecessary iterations to reduce the total data sent and migration time.

Smart Stop is also able to reduce the pause time of the kernel compile by over 30% (Figure 9(a)). This is because the compilation exhibits a variance in the rate at which memory is modified (Figure 4(b)). The algorithm is thus able to pick a more intelligent iteration to conclude the migration, minimizing the amount of data that needs to be sent in the final iteration.

	Data Transfer (MB)		Page Delta
	Iter 1	Iters 2-30	Savings (MB)
TPC-W	954	315	172
Kernel	877	394	187
SPECjbb	932	163	127

Table 2. The Page Delta optimization cannot be used during the first iteration, but it provides substantial savings during the remaining rounds.

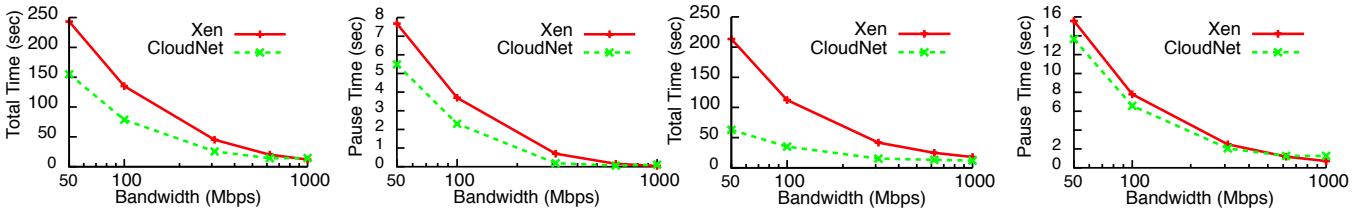
Redundancy Elimination: Figure 11 shows the amount of memory redundancy found in each applications during migrations over a 100 Mbps link when each memory page is split into four blocks. SPECjbb exhibits the largest level of redundancy; however, the majority of the redundant data is from empty "zero" pages. In contrast, a kernel compilation has about 13% redundancy, of which less than half is zero pages. The CBR optimization eliminates this redundancy, providing substantial reductions in the total data transferred and migration time (Figure 9). Since CBR can eliminate redundancy in portions of a page, it also can significantly lower the pause time since pages sent in the final iteration often have only small modifications, allowing the remainder of the page to match the CBR cache. This particularly helps the kernel compile and TPC-W migrations which see a 40 and 26 percent reduction in pause time respectively. SPECjbb does not see a large pause time reduction because most of the redundancy in its memory is in unused zero pages which are almost all transferred during the migration's first iteration.

Page Deltas: The first iteration of a migration makes up a large portion of the total data sent since during this iteration the majority of a VM's memory–containing less frequently touched pages–is transferred. Since the Page Delta optimization relies on detecting memory addresses that have already been sent, it can only be used from the second iteration onward, and thus provides a smaller overall benefit, as seen in Figure 9.

Table 2 shows the amount of memory data transferred during the first and remaining iterations during migrations of each application. While the majority of data is sent in the first round, during iterations 2 to 30 the Page Delta optimization still significantly reduces the amount of data that needs to be sent. For example, TPC-W sees a reduction from 487MB to 315MB, a 36 percent improvement.

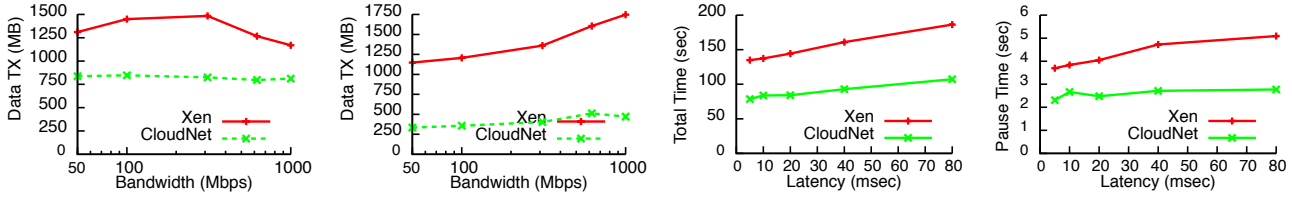
Currently, the Page Delta optimization does not reduce migration time as much as it reduces data transferred due to inefficiencies in the code. With further optimization, the Page Delta technique could save both bandwidth and time.

Results Summary: The combination of all optimizations improves the migration performance more than any single technique. While the Page Delta technique only comes into effect after the first iteration, it can provide significant reductions in the amount of data sent during the remainder of the migration. The CBR based approach, however, can substantially reduce the time of the first iteration during which many empty or mostly empty pages are transferred. Finally, Smart Stop eliminates many unnecessary iterations



(a) TPC-W (b) SpecJBB

Figure 12. Decreased bandwidth has a large impact on migration time, but CloudNet's optimizations reduce the effects in low bandwidth scenarios.



(a) TPC-W Bandwidth Usage (b) SPECjbb Bandwidth Usage (c) TPC-W Latency Impact

Figure 13. (a-b) CloudNet's optimizations significantly reduce bandwidth consumption. (c) Increased latency has only a minor impact on the migration process, but may impact application performance due to synchronous disk replication.

and combines with both the CBR and Page Delta techniques to minimize the pause time during the final iteration.

4.7 Impact of Network Conditions

We next use the network emulator testbed to evaluate the impact of latency and bandwidth on migration performance.

Bandwidth: Many data centers are now connected by gigabit links. However, this is shared by thousands of servers, so the bandwidth that can be dedicated to the migration of a single application is much lower. In this experiment we evaluate the impact of bandwidth on migrations when using a shared storage system. We vary the link bandwidth from 50 to 1000 Mbps, and maintain a constant 10 msec round trip delay between sites.

Figure 12 compares the performance of default Xen to CloudNet's optimized migration system. We present data for TPC-W and SPECjbb; the kernel compile performs similar to TPC-W. Decreased bandwidth increases migration time for both applications, but our optimizations provide significant benefits, particularly in low bandwidth scenarios. CloudNet also substantially reduces the amount of data that needs to be transferred during the migration because of redundancy elimination, page delta optimization and the lower number of iterations, as seen in Figure 13(a-b).

CloudNet's code presently does not operate at linespeed when the transfer rate is very high (e.g. about 1Gbps or higher *per VM transfer*). Thus in high bandwidth scenarios, CloudNet reduces the data transferred, but does not significantly affect the total migration or pause time compared to default Xen. We expect that further optimizing the CloudNet code will improve performance in these areas, allowing the optimizations to benefit even LAN migrations.

Latency: Latency between distant data centers is inevitable due to speed of light delays. This experiment tests how latency impacts migration performance as we adjust the delay introduced by the network emulator over a 100Mbps link. Even with TCP settings optimized for WAN environments, slow start causes performance to decrease some as latency rises. CloudNet's optimizations still provide a consistent improvement regardless of link latency as shown in Figure 13(c). While latency has only a minor impact on total migration and pause time, it can degrade application performance due to the synchronous disk replication required during the VM migration. Fortunately, CloudNet's optimizations can significantly reduce this period of lowered performance.

Results Summary: CloudNet's optimized migrations perform well even in low bandwidth (50 to 100Mbps) and high latency scenarios, requiring substantially less data to be transferred and reducing migration times compared to default Xen. In contrast to commercial products that require 622 Mbps per VM transfer, CloudNet enables efficient VM migrations in much lower bandwidth and higher latency scenarios.

4.8 Consolidation: Simultaneous Migrations

We next mimic an enterprise consolidation where four VMs running a distributed development environment must be transitioned from the data center in Texas to the data center in Illinois. Each of the VMs has a 10GB disk (of which 6GB is in use) and is allocated 1.7GB of RAM and one CPU, similar to a "small" VM instance on Amazon EC2[2]. The load on the cluster is created by repeatedly running a distributed kernel compilation across the four VMs. The maximum bandwidth available between the two sites was measured as 465Mbps with a 27 msec round trip latency; note that bandwidth must be *shared* by the four simultaneous migrations.

We first run a baseline experiment using the default DRBD and Xen systems. During the disk synchronization period a total of 24.1 GB of data is sent after skipping the empty disk blocks. The disk transfers take a total of 36 minutes. We then run the VM memory migrations using the default Xen code, incurring an additional 245 second delay as the four VMs are transferred.

Next, we repeat this experiment using CloudNet's optimized migration code and a 1GB CBR cache for the disk transfer. Our optimizations reduce the memory migration time to only 87 seconds, and halves the average pause time from 6.1 to 3.1 seconds. Figure 14 compares the bandwidth consumption of each approach. CloudNet reduces the data sent during the disk transfers by 10GB and lowers the memory migrations from 13GB to 4GB. In total, the

[2] Small EC2 instances have a single CPU, 1.7GB RAM, a 10GB root disk, plus an additional 150GB disk. Transferring this larger disk would increase the storage migration time, but could typically be scheduled well in advance.

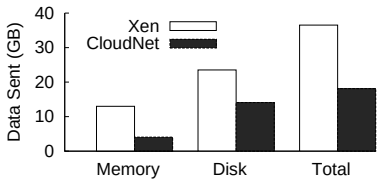


Figure 14. CloudNet saves nearly 20GB of bandwidth when simultaneously migrating four VMs.

data transferred to move the memory and storage for all four VMs falls from 37.4GB in the default Xen case to 18.5GB when using CloudNet's optimizations.

Results Summary: CloudNet's optimizations reduce pause time by a factor of 2, and lower memory migration time–when application performance is impacted most–by nearly 3X. The combination of eliminating redundant memory state and disk blocks can reduce the total data transferred during the migration by over 50%, saving nearly 20GB worth of network transfers.

4.9 Follow-the-Sun: Disk Synchronization

In a follow-the-sun scenario, one or more applications are moved between geographic locations in order to be co-located with the workforce currently using the application. In this experiment we consider moving an application with a large amount of state back and forth between two locations. We focus on the disk migration cost and demonstrate the benefits of using incremental state updates when moving back to a location which already has a snapshot from the previous day.

We use the TPC-W web application, but configure it with a much larger 45GB database. The initial migration of this disk takes 3.6 hours and transfers 51GB of data to move the database and root operating system partitions. We then run a TCP-W workload which lasts for 12 hours to represent a full workday at the site. After the workload finishes, we migrate the application back to its original site. In this case, only 723MB of storage data needs to be transferred since the snapshot from the previous day is used as a base image. This reduces the migration time to under five minutes, and the disk and memory migrations can be performed transparently while workers from either site are accessing the application. This illustrates that many applications with large state sizes typically only modify relatively small portions of their data over the course of a day. Using live migration and incremental snapshots allows applications to be seamlessly moved from site to site for relatively little cost and only minimal downtime.

5. Related Work

Cloud Computing: Armbrust et al provide a thorough overview of the challenges and opportunities in cloud computing [3]. There are several types of cloud platforms, but we focus on Infrastructure as a Service (IaaS) platforms which rent virtual machine and storage resources to customers. InterCloud explores the potential for federated cloud platforms to provide highly scalable services [6]; CloudNet seeks to build a similar environment and uses WAN migration to move resources between clouds and businesses.

Private Clouds & Virtual Networks: The VIOLIN and Virtuoso projects use overlay networks to create private groups of VMs across multiple grid computing sites [23, 25]. VIOLIN also supports WAN migrations over well provisioned links, but does not have a mechanism for migrating disk state. Overlay network approaches require additional software to be run on each host to create network tunnels. CloudNet places this responsibility on the routers at each site, reducing the configuration required on end hosts.

Our vision for Virtual Private Clouds was initially proposed in [30]. Subsequently, Amazon EC2 launched a new service also called "Virtual Private Clouds" which similarly uses VPNs to securely link enterprise and cloud resources. However, Amazon uses IPSec based VPNs that operate at layer-3 by creating software tunnels between end hosts or IPSec routers. In contrast, CloudNet focuses on VPNs provided by a network operator. Network based VPNs are typically realized and enabled by multiprotocol label switching (MPLS) provider networks, following the "hose model" [12] and are commonly used by enterprises. Provider based VPNs can be either layer-3 VPNs following RFC 2547, or layer-2 virtual private LAN Service (VPLS) VPNs according to RFC 4761. CloudNet relies on network based VPLS as it simplifies WAN migration, has lower overheads, and can provide additional functionality from the network provider, such as resource reservation.

LAN Migration: Live migration is essentially transparent to any applications running inside the VM, and is supported by most major virtualization platforms [9, 18, 21]. Work has been done to optimize migration within the LAN by exploiting fast interconnects that support remote memory access technology [17]. Jin et al. have proposed using memory compression algorithms to optimize migrations [19]. Breitgand et al. have developed a model based approach to determine when to stop iterating during a memory migration [5], similar to Smart Stop. Their approach can allow them to more precisely predict the best time to stop, but it requires knowledge of the VM's memory behavior, and it is not clear how the model would perform if this behavior changes over time. CloudNet's CBR and Page Delta optimizations are simple forms of compression, and more advanced compression techniques could provide further benefits in low bandwidth WAN scenarios, although at the expense of increased CPU overhead. The Remus project uses a constantly running version of Xen's live migration code to build an asynchronous high availability system [10]. Remus obtains a large benefit from an optimization similar to CloudNet's Page Delta technique because it runs a form of continuous migration where pages see only small updates between iterations.

WAN Migration: VMware has announced limited support for WAN migration, but only under very constrained conditions: 622 MBps link bandwidth and less than 5 msec network delay [29]. CloudNet seeks to lower these requirements so that WAN migration can become an efficient tool for dynamic provisioning of resources across data centers. Past research investigating migration of VMs over the WAN has focused on either storage or network concerns. Bradford et al. describe a WAN migration system focusing on efficiently synchronizing disk state during the migration; they modify the Xen block driver to support storage migration, and can throttle VM disk accesses if writes are occurring faster than what the network supports [4]. Shrinker uses content based addressing to detect redundancy across *multiple* hosts at the destination site during VM migrations [22]. This could allow it to reduce bandwidth costs compared to CloudNet, but exposes it to security concerns due to hash collisions, although the likelihood of this can be bounded. The VM Turntable Demonstrator showed a VM migration over intercontinental distances with latencies of nearly 200 msec; they utilize gigabit lightpath links, and like us, find that the increased latency has less impact on performance than bandwidth [27]. Harney et al. propose the use of Mobile IPv6 to reroute packets to the VM after it is moved to a new destination [15]; this provides the benefit of supporting layer-3 connections between the VM and clients, but the authors report a minimum downtime of several seconds due to the Mobile IP switchover, and the downtime increases further with network latency. In this work, we leverage existing mechanisms to simplify storage migration and network reconfiguration, and propose a set of optimizations to reduce the cost of migrations in low bandwidth and high latency environments.

6. Conclusions

The scale of cloud computing is growing as business applications are increasingly being deployed across multiple global data centers. We have built CloudNet, a prototype cloud computing platform that coordinates with the underlying network provider to create seamless connectivity between enterprise and data center sites, as well as supporting live WAN migration of virtual machines. CloudNet supports a holistic view of WAN migration that handles persistent storage, network connections, and memory state with minimal downtime even in low bandwidth, high latency settings.

While existing migration techniques can wastefully send empty or redundant memory pages and disk blocks, CloudNet is optimized to minimize the amount of data transferred and lowers both total migration time and application-experienced downtime. Reducing this downtime is critical for preventing application disruptions during WAN migrations. CloudNet's use of both asynchronous and synchronous disk replication further minimizes the impact of WAN latency on application performance during migrations. We have demonstrated CloudNet's performance on both a prototype deployed across three data centers separated by over 1,200km and a local testbed. During simultaneous migrations of four VMs between operational data centers, CloudNet's optimizations reduced memory transfer time by 65%, and saved 20GB in bandwidth for storage and memory migration.

Acknowledgements: This work was supported in part by NSF grants CNS-0916972, CNS-0720616, CNS-0855128, and a VURI award from AT&T.

References

[1] B. Aggarwal, A. Akella, A. Anand, P. Chitnis, C. Muthukrishnan, A. Nair, R. Ramjee, and G. Varghese. EndRE: An end-system redundancy elimination service for enterprises. In *Proceedings of NSDI*, 2010.

[2] A. Anand, V. Sekar, and A. Akella. SmartRE: an architecture for coordinated network-wide redundancy elimination. *SIGCOMM Comput. Commun. Rev.*, 39(4):87–98, 2009.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A Berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

[4] R. Bradford, E. Kotsovinos, A. Feldmann, and H. Schiöberg. Live wide-area migration of virtual machines including local persistent state. In *Proceedings of the 3rd international conference on Virtual execution environments*, pages 169–179, San Diego, California, USA, 2007. ACM.

[5] D. Breitgand, G. Kutiel, and D. Raz. Cost-aware live migration of services in the cloud. In *Proceedings of the 3rd Annual Haifa Experimental Systems Conference*, SYSTOR '10, New York, NY, USA, 2010. ACM.

[6] R. Buyya, R. Ranjan, and R. N. Calheiros. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In *International Conference on Algorithms and Architectures for Parallel Processing*, 2010.

[7] X. Chen, Z. M. Mao, and J. Van der Merwe. ShadowNet: a platform for rapid and safe network evolution. In *USENIX Annual Technical Conference*, 2009.

[8] Cisco Active Network Abstraction. `http://www.cisco.com`.

[9] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In *Proceedings of NSDI*, May 2005.

[10] B. Cully, G. Lefebvre, D. Meyer, M. Feeley, N. Hutchinson, and A. Warfield. Remus: High availability via asynchronous virtual machine replication. In *NSDI*, 2008.

[11] Drbd. `http://www.drbd.org/`.

[12] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. Van der Merwe. Resource management with hoses: point-to-cloud services for virtual private networks. *IEEE/ACM Transactions on Networking*, 10(5), 2002.

[13] D. Gupta, S. Lee, M. Vrable, S. Savage, A. C. Snoeren, G. Varghese, G. M. Voelker, and A. Vahdat. Difference engine: harnessing memory redundancy in virtual machines. *Commun. ACM*, 53(10):85–93, 2010.

[14] M. Hajjat, X. Sun, Y. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. In *Proceedings of SIGCOMM*, 2010.

[15] E. Harney, S. Goasguen, J. Martin, M. Murphy, and M. Westall. The efficacy of live virtual machine migrations over the internet. In *Proceedings of the 3rd VTDC*, 2007.

[16] Hsieh. hash functions. `http://www.azillionmonkeys.com/qed/hash.html`.

[17] W. Huang, Q. Gao, J. Liu, and D. K. Panda. High performance virtual machine migration with RDMA over modern interconnects. In *Proceedings of the 2007 IEEE International Conference on Cluster Computing*, pages 11–20. IEEE Computer Society, 2007.

[18] Microsoft Hyper-V Server. `www.microsoft.com/hyper-v-server`.

[19] H. Jin, L. Deng, S. Wu, X. Shi, and X. Pan. Live virtual machine migration with adaptive memory compression. In *Cluster*, 2009.

[20] Juniper Networks, Configuration and Diagnostic Automation Guide. `http://www.juniper.net`.

[21] M. Nelson, B.-H. Lim, and G. Hutchins. Fast transparent migration for virtual machines. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, 2005.

[22] P. Riteau, C. Morin, and T. Priol. Shrinker: Efficient Wide-Area Live Virtual Machine Migration using Distributed Content-Based Addressing. Research Report RR-7198, INRIA, 02 2010.

[23] P. Ruth, J. Rhee, D. Xu, R. Kennell, and S. Goasguen. Autonomic live adaptation of virtual computational environments in a multi-domain infrastructure. In *Proceedings of ICAC*, 2006.

[24] The SPEC java server benchmark. `http://spec.org/jbb2005/`.

[25] A. I. Sundararaj and P. A. Dinda. Towards virtual networks for virtual machine grid computing. In *VM'04: Proceedings of the 3rd conference on Virtual Machine Research And Technology Symposium*, 2004.

[26] TPC. the tpcw benchmark. Website. `http://www.tpc.org/tpcw/`.

[27] F. Travostino, P. Daspit, L. Gommans, C. Jog, C. de Laat, J. Mambretti, I. Monga, B. van Oudenaarde, S. Raghunath, and P. Y. Wang. Seamless live migration of virtual machines over the MAN/WAN. *Future Generation Computer Systems*, Oct. 2006.

[28] J. Van der Merwe, A. Cepleanu, K. D'Souza, B. Freeman, A. Greenberg, D. Knight, R. McMillan, D. Moloney, J. Mulligan, H. Nguyen, M. Nguyen, A. Ramarajan, S. Saad, M. Satterlee, T. Spencer, D. Toll, and S. Zelingher. Dynamic connectivity management with an intelligent route service control point. In *Proceedings of the 2006 SIGCOMM workshop on Internet network management*.

[29] Virtual machine mobility with VMware VMotion and Cisco Data Center Interconnect Technologies. http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns836/white_paper_c11-557822.pdf, Sept. 2009.

[30] T. Wood, A. Gerber, K. Ramakrishnan, J. Van der Merwe, and P. Shenoy. The case for enterprise ready virtual private clouds. In *Proceedings of the Usenix Workshop on Hot Topics in Cloud Computing (HotCloud), San Diego, CA*, June 2009.

[31] T. Wood, G. Tarasuk-Levin, P. Shenoy, P. Desnoyers, E. Cecchet, and M. Corner. Memory buddies: Exploiting page sharing for smart colocation in virtualized data centers. In *2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE 2009)*, Washington, DC, USA, March 2009.