CloudNet: A Platform for Optimized 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

Cloud computing platforms are growing from clusters of machines within a data center to networks of data centers with resources spread across the globe. Virtual machine migration within the LAN has changed the scale of resource management from allocating resources on a single server to manipulating pools of resources within a data center. We expect WAN migration to likewise transform the scope of provisioning from a single data center to multiple data centers spread across the country or around the world. In this paper we propose a cloud computing platform linked with a VPN based network infrastructure that provides seamless connectivity between enterprise and data center sites, as well as support for live WAN migration of virtual machines. We describe a set of optimizations that minimize the cost of transferring persistent storage and moving virtual machine memory during migrations over low bandwidth, high latency Internet links. Our evaluation on both a local testbed and across two real data centers demonstrates that these improvements can reduce total migration and pause time by over 30%. During simultaneous migrations of four VMs between Texas and Illinois, CloudNet's optimizations reduce memory migration time by 65% and lower bandwidth consumption for the storage and memory transfer by 20GB, a 57% reduction.

1 Introduction

Cloud computing enables both large and small enterprises to better manage their resources—some no longer need to invest in local IT resources and can instead lease cheaper, on-demand resources from providers, while others can utilize the flexibility of cloud resources to dynamically meet peak demand without having to over-provision in-house resources. Since cloud platforms typically rely on virtualization, new resources can be quickly and dynamically added within minutes. From a cloud computing service provider's perspective, server virtualization allows flexible multiplexing of resources among customers without the need to dedicate physical resources individually.

Current commercial solutions present cloud servers as isolated entities with their own IP address space outside the customer's control. This separation of cloud and enterprise resources increases software and configuration complexity when deploying services, and can lead to security concerns since enterprise customers must utilize IP addresses on the *public* Internet for their cloud resources. Cloud platforms leave the onus on the the customer to securely connect the cloud and enterprise resources and manage firewall rules. A more desirable architecture is for storage and compute resources in the cloud to be seamlessly connected to an enterprise's users and applications, acting as if they were secure, local resources within the enterprise LAN.

In such a scenario, we envision that an enterprise's IT services will be spread across the corporation's data center as well as dynamically set-up cloud data centers. Enterprises may choose to locate applications in provider cloud data centers for performance reasons, e.g., when the provider cloud is more optimally placed between customer sites than the enterprise' own data center, or it might utilize the provider cloud to handle "overflows" from local servers during periods of peak demands. Ideally, these cloud data centers could be located anywhere in the world to take advantage of costs like energy, infrastructure and labor, or workload metrics such as diurnal usage patterns. Further, cloud data centers in certain geographies can be exploited to move data and applications closer to end-users. These challenges increase when placement decisions can change, requiring applications to be dynamically moved between data centers in response to changing costs or workloads.

As a consequence, quickly and transparently migrating computing and storage from one data center to another (whether in the enterprise or in the cloud) will be necessary to break the boundaries between geographically separated data centers. WAN migration changes *the scale of provisioning from managing servers on a rack to optimizing pools of resources from multiple data centers.* It also greatly simplifies deployment into the cloud, allowing an enterprise to seamlessly move a live application from its own infrastructure into a cloud data center without incurring any downtime. Unfortunately, existing virtual machine migration techniques are designed for the LAN, and are not sufficiently optimized to perform well in low bandwidth, high latency settings that are

typical in WAN environments. Research prototypes and commercial products are only beginning to make WAN migration feasible, and the requirements in terms of storage and network configuration, as well as bandwidth and latency needs still prevent it from being practical.

We propose a platform called *CloudNet* in order to achieve the vision of securely connected enterprise and cloud sites that support dynamic migration of resources. CloudNet uses virtual private networks (VPNs) to provide secure communication channels and allow customers greater control over network provisioning and configuration between their sites and the cloud. CloudNet bridges the local networks of multiple data centers, making WAN-based cloud resources look like local LAN resources and allowing LAN-based protocols to seamlessly operate across these bridged WAN sites, albeit with increased network delay. As a consequence, LAN-based live virtual machine migration techniques [9] operate *unmodified* over CloudNet, allowing VMs to be moved across WAN sites. However, such a capability addresses only part of the problem, as LAN-based live migration techniques perform poorly in low-bandwidth high-latency WAN settings. To address this key challenge, CloudNet incorporates a set of optimizations to significantly improve performance of VM migration in WAN environments. Further, while traditional live migration methods assume a shared file system is available at both sites, CloudNet allows VM migration across data centers with or without shared storage by also migrating disk data when no shared storage is available. Our contributions include:

1. The design and implementation of a cloud computing platform that seamlessly connects resources at multiple data center and enterprise sites.
2. A holistic view of WAN migration that handles persistent storage, network connections, and memory state with minimal downtime.
3. Optimizations that minimize the total migration time, application downtime, and the volume of data transferred.
4. An extensive evaluation of how different application types impact migration performance under a variety of network conditions.

Our experiments using a set of realistic applications show CloudNet's optimizations decreasing memory migration and pause time by 30 to 70% in typical link capacity scenarios. We also evaluate application performance during migrations to show that CloudNet's optimizations reduce the window of decreased performance as VM state is transferred compared to existing techniques.

2 CloudNet Design Overview

In this section, we present some background and an overview of the CloudNet design, along with the motivation for why WAN migration is essential for managing resources across data centers.

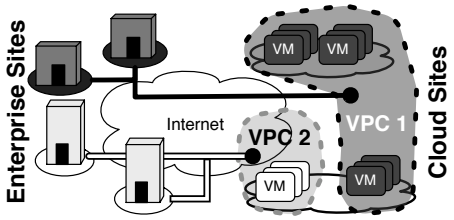


Figure 1: Two VPCs isolate resources within the cloud sites and securely link them to the enterprise networks.

2.1 Seamless, Secure Cloud Connections

Most cloud platforms allow cloud resources to have either private IP addresses that confine them within the cloud, or public IP addresses that allow them to be connected to the enterprise, but also potentially expose them to malicious Internet traffic. These cloud platforms rely on user configured firewalls, such as Amazon EC2's "security groups", to ensure that cloud and enterprise resources can be securely connected. These approaches are inadequate for enterprise needs as no effort is made to give the abstraction that cloud resources are seamlessly connected to the enterprise's existing infrastructure, and misconfiguration can easily leave resources unprotected.

To address these transparency and security challenges, CloudNet uses the notion of a Virtual Private Cloud (VPC)[1]. A VPC is a combination of cloud computing resources with a VPN infrastructure to give users the abstraction of a private set of cloud resources that are transparently and securely connected to their own infrastructure. Figure 1 shows a pair of VPCs that span multiple cloud data centers, but present a unified pool of resources to each enterprise.

Seamless network connections: CloudNet uses MPLS-based VPNs to create the abstraction of a private network and address space shared by all VPN endpoints, connecting resources from different sites as if they were on a single network. Since addresses are specific to a VPN, the cloud operator can allow customers to use any IP address ranges that they like without worrying about conflicts between cloud customers. Another benefit of MPLS-based VPNs is that the level of abstraction can be made even greater with *Virtual Private LAN Services (VPLS)* that bridge multiple VPN endpoints onto a single LAN segment. This allows cloud resources to appear *indistinguishable* from existing IT infrastructure already on the enterprise's own LAN.

Secure any-to-any communication: VPNs are already used by many large enterprises, and cloud sites can be easily added as new secure endpoints within these existing networks. VPCs use VPNs to provide secure communication channels via the creation of "virtually dedicated" paths in the

[1] After proposing the virtual private cloud concept in [29], we have since found it also used on a blog post encouraging the use of VPNs and cloud computing [13], and it has subsequently been used for an Amazon product.

provider network. This 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.

2.2 Resource Pools that Span Data Centers

As enterprises increase their reliance on cloud computing for cheap and dynamic access to resources, it has become necessary to manage and optimize resources across multiple data centers. For instance, a single cloud provider may expose the presence of different geographically-separate data centers—e.g., as "availability regions" in EC2 [4]—enabling an enterprise to perform cross-geographic placement and optimizations. Similarly, an enterprise may lease resources from different cloud providers, each with their own data center, and perform cross-data center optimizations to ensure availability or to exploit dynamic prices. Today, jointly managing multiple data centers across the Internet is difficult because the lack of seamless connections between sites isolates resources, and there are only limited mechanisms for moving resources between locations.

CloudNet's VPC architecture simplifies cross-data center management, since its use of VPNs enables independent resource pools at each cloud site to be grouped into a single pool of resources transparently connected to the enterprise. Resources at new cloud data centers can be easily mapped into the VPC, and existing resources can be efficiently moved between enterprise and data center sites. Further, application-level considerations such as workloads or fault tolerance requirements can be used to dynamically decide where to place individual VMs.

2.3 Efficient WAN Migration

In order to dynamically manage and optimize resources across multiple data centers, an enterprise must have the ability to efficiently perform live migration of applications (and their data) across data centers. Several virtualization platforms support efficient migration of VMs within a local network [9, 20]. By virtue of presenting WAN resources as LAN resources, CloudNet's VPC 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 performance, as we show in Section 3.3. In fact, VMWare's recently announced support for WAN VM migration between nearby data centers requires at least 622 Mbps of bandwidth dedicated to the transfer, and is designed for links with less than 5 msec latency [3]. Despite being interconnected using "fat" gigabit pipes, data centers will typically be unable to dedicate such high bandwidth for a single application transfer, plus enterprises will want the ability to migrate a group of related VMs concurrently. Further, current live VM migration techniques assume the presence of a shared file system, which enables them to migrate only memory state 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, WAN migration techniques must be able to optionally migrate an application's disk state, in addition to migrating its memory state. Current LAN-based live migration techniques must be optimized for WAN environments before enterprises can fully exploit their benefits for cross data-center resource management, and this is the primary focus of this paper.

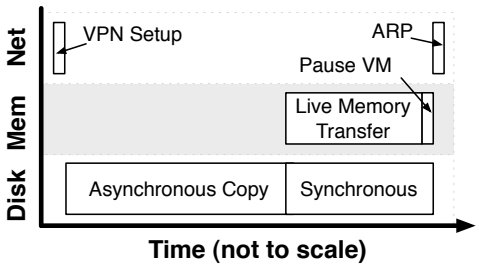


Figure 2: The phases of a migration for non-shared disk, memory, and the network in CloudNet.

3 WAN Migration in CloudNet

Consider an organization which desires to move one or more applications (and possibly their data) from Data Center A to Data Center B. 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 layer-2 connectivity between data centers, if needed.

Step 2: If storage is not shared, transfer the application's disk state.

Step 3: Transfer the memory state of the application to a server in Data Center B, as it continues running without interruption.

Step 4: Once the disk and memory state have been transferred, briefly pause the application for the final transition of memory and processor state to Data Center B. This process must also maintain 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, making it difficult or impossible to seamlessly transfer active network connections. CloudNet avoids this problem by using VPLS VPN technology in step 1, and utilize a set of migration optimizations to improve performance in the other steps.

3.1 VPLS-Driven Migration

Bridging sites A and B with a layer-2 connection simplifies network reconfiguration during a migration because it provides the abstraction of a single LAN across Data Centers A and B. While there are several technologies available to create such connections, CloudNet uses VPLS based VPNs since these are already commonly used by enterprises. In many cases, Data Center B will already be a part of the customer's virtual private cloud, 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 VPC into the new data center.

Creating a new VPLS endpoint involves configuration changes on the data center router in question. This is a process that can be readily automated via configuration interfaces on modern routers [2, 1]. Group membership in VPLS VPN is typically determined during this configuration phase. However, to facilitate more dynamic group changes, CloudNet uses a centralized VPN Controller to adjust which VPLS endpoints are grouped together to form each virtual private cloud. The VPN Controller maintains a ruleset indicating which endpoints should have connectivity; as all route control messages pass through the VPN Controller, it is able to control how the tunnels forming each VPLS are created. This ensures that each customer's resources are isolated within their own VPLS networks, providing CloudNet's virtual private cloud abstraction.

Maintaining Network Connections: Once disk and memory state have been migrated (as discussed in the subsequent sections), CloudNet must ensure that active network connections are redirected to Data Center B. In LAN migration, this is achieved by having the destination host transmit an unsolicited ARP message that causes the local 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 data centers A and B, causing the ARP message to be forwarded over the Internet to update the switch mappings at both sites. This allows open network connections to be seamlessly redirected to the VM's new location.

3.2 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 the enterprise has access to a global SAN, then WAN migration can be achieved by simply granting the VM secure access to the SAN from both data centers.

Otherwise, we have a "shared nothing" architecture where VM storage must be migrated along with the VM memory state. CloudNet uses a disk replication system that migrates storage similar to how memory is transferred during a VM migration. Once a VM migration has been planned, the replication system must copy the VM'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 asynchronous replication 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 migration finishes and the VM becomes live on the remote host. When the migration completes, the new host's disk becomes the primary node in the replication scheme, and the origin's disk is disabled.

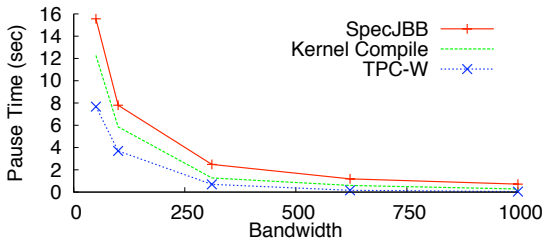


Figure 3: Low bandwidth Internet links can significantly increase the time required to migrate virtual machines.

3.3 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, 20]. At a certain point, the VM is paused to copy the final memory state. WAN migration can be accomplished by similar means, but the decreased bandwidth can lead to decreased performance–particularly much higher VM down times–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 4.

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. In a WAN migration, it is desirable to minimize both of these times and the bandwidth costs for transferring data.

As bandwidth decreases, the total time and pause time incurred by a migration can rise dramatically. Figure 3 shows the pause time of VMs running several different applications as the available bandwidth is varied (assumes shared storage

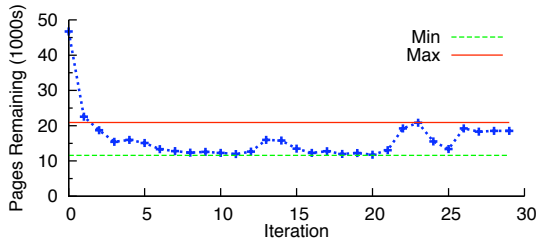


Figure 4: The number of pages to be sent initially decreases, but soon levels off. Intelligently deciding when to stop a migration eliminates wasteful transfers and can lower pause time.

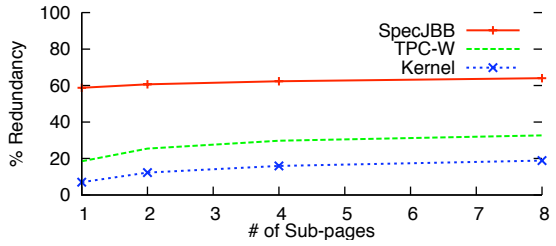


Figure 5: Different applications have different levels of redundancy. Using finer granularity finds more redundancy, but has diminishing returns.

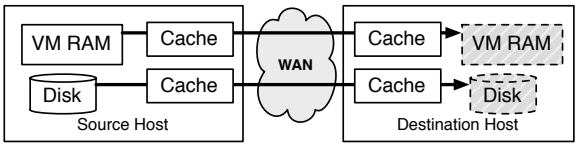


Figure 6: CloudNet maintains a cache within the VM and storage migration systems at each site to eliminate the transfer of redundant data.

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, particularly during the final iteration. This result illustrates the importance of optimizing VM migration algorithms to better handle low bandwidth connections.

4 Optimizing WAN Migration

In this section we propose a set of optimizations to improve the performance of migration over the WAN.

4.1 Smart Stop and Copy

The default Xen migration algorithm will iterate until either a very small number of pages remain to be sent, it has already sent more than three times the VM's total memory, or a limit of 30 iterations is reached. At that point, the VM is paused, and all remaininag pages are sent. In practice, for VMs with even a moderate memory dirty rate, the 30 iteration limit determines when a migration finishes. 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 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 at the end of each iteration. This indicates that (i) a large number of iterations only extends the total migration time and increases the total 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. We propose a heuristic that tracks the number of pages remaining to be sent over a short history to find a local minimum for expected pause time. When the migration begins, the VM is run through several iterations, recording the number of pages remaining to be sent into a sliding history buffer. After the history window has been filled, if the current number of pages remaining to be sent at the end of an iteration is lower than any previous entry in the history, then the VM is paused and the final iteration is begun. We have found that this greedy approach works well in practice, and use a window size of 5 iterations. Smart Stop could perform poorly if there is an increasing trend in remaining pages; if this is detected, the migration is terminated early.

4.2 Content Based Redundancy

Content based redundancy (CBR) elimination techniques have been used previously to save bandwidth between network routers [5], and we use a similar approach to eliminate the redundant data while transferring VM memory and disk state. Disks can have large amounts of redundant data caused by either empty blocks or similar files. Likewise, it has been shown in previous virtualization research that pairs of VMs often have identical pages in memory [28, 18], however, even within a single system there is often redundant pages or portions of pages.

Previous approaches for eliminating network redundancy use sliding window hashes called Rabin fingerprints to find

redundant strings of data within a packet [5, 24]. CloudNet can support either Rabin fingerprints, or a simpler, block based approach that detects identical, fixed size regions in either a memory page or disk block.

The block based CBR approach splits each memory page or disk block into a fixed number of blocks and generates hashes based on their content. If a hash matches an entry in caches maintained at the source and destination hosts, then a block with the same contents was sent previously. The migration algorithm can then simply send a 32bit index to the cache entry instead of the full block (4KB for a full disk or memory page).

Dividing a memory or disk page into smaller blocks allows redundant data to be found with finer granularity. Figure 5 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 blocks raises the level of redundancy that is found, but it can incur greater overhead since each block requires a hash table lookup.

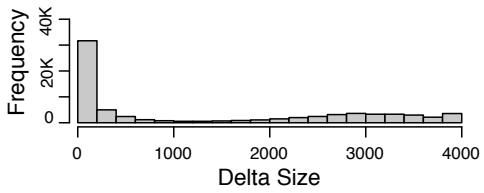
Figure 6 illustrates CloudNet's CBR system. While we have chosen to place the CBR caches within the Xen migration code and storage synchronization systems individually, it would be possible to use redundancy elimination within the network routers between sites [5]. We make our modifications within the Xen migration code because it requires no extra support from the network infrastructure, simplifies maintaining cache consistency, and we believe our optimization code will be a valuable contribution back to the Xen community.

Our block based approach can only find matching content if it occurs at the same offset within a block, but our evaluation in Section 6.6 indicates that it finds a similar level of redundancy to the more flexible Rabin based approach, and the fixed size blocks are much simpler to integrate into the Xen migration code.

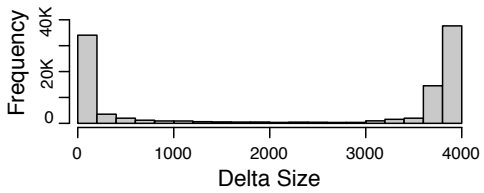
4.3 Using Page Deltas

After the first iteration, most of the pages transferred are pages which have been sent previously, but have since been modified. Since an application may be modifying only portions of pages, another approach to reduce the bandwidth consumed during migration 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 disaster recovery system to reduce the bandwidth required for VM synchronization [10].

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 the difference between the current version and the stored version of the page is calculated, and only the delta is sent.



(a) Kernel Compile



(b) TPC-W

Figure 7: 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 7 shows histograms of delta sizes calculated during migrations of two different applications. A smaller delta is better since it means less data needs 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.

5 CloudNet Implementation

We have implemented a prototype of CloudNet that uses three key building blocks: (i) the Xen virtualization platform, (ii) the DRBD storage replication protocol, and (iii) a commercial router-based VPLS/ layer-2 VPN impelmentation. Our CloudNet prototype assumes that each data center runs the Xen virtualization platform on its servers and runs applications inside Xen virtual machines. Application data is assumed to be either stored on a SAN (in which case, it is assumed to be accessible across data centers and not migrated) or stored on disks that are local to each data center (in which case it must be migrated along with an application). In the latter case, we use the DRBD storage replication software to migrate data from one data center to another. Last, we assume that each data center employs routers that provide layer-2 VPN support with VPLS; our current prototype relies on Juniper's commercial implementation of VPLS.

5.1 VPLS Router Reconfiguration

CloudNet must be able to dynamically manipulate the routers at each data center site in order to create VPN endpoints. To

do this, we use Juniper routers that have a remote API that can be used to programatically adjust the router configuration.

5.2 Storage Migration with DRBD

DRBD is a storage migration system that was recently integrated into the main line linux kernel [11]. CloudNet employs DRBD to migrate disk state of an application between data centers. The migration proceeds in two steps. First, a blank disk is created at the target. CloudNet then uses DRBD's asynchronous mode to perform an iterative pre-copy, transmitting all disk blocks to the destination and updating any subsequently modified blocks. Once the disks are synchronized, DRBD is switched to synchronous mode and initiates the memory state migration. This keeps both disks in sync by synchronously propagating all writes to both disks while Xen's memory state is migrated. Finally, DRBD is switched to dual primary mode during the switchover, allowing the VM to write to the disk at the destination host once the migration is complete. At this point, the source disk can be disconnected and its data deleted if appropriate.

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

5.3 Memory Optimizations

CloudNet extends Xen's live migration code as follows.

Smart Stop & Copy: We have adjusted the migration code to use Xen's dirty bitmap to calculate the number of pages remaining to be sent at the end of each iteration. This data is used to decide if the migration should continue through another iteration, or if it should be ended early. This change comprises only a few dozen lines added to Xen's migration code.

Content Based Redundancy: CloudNet adds a content indexed cache that is checked before sending each page or portion of a page. A fingerprint of the page's content is generated using the Super Fast Hash Algorithm and used as an index into a 100MB FIFO based cache. If the fingerprint is found in the cache, the cached and actual pages are compared to ensure the match was not caused by a hash collision. If all bytes match, then only the index into the cache is sent to the destination host. Otherwise, the full page is sent, and the page and fingerprint are added to the cache.

Alternatively, CloudNet can use a Rabin Fingerprint based redundancy elimination algorithm. This approach generates a set of fingerprints across the memory page which can be used to find arbitrary length strings of redundant data as described in [5]. However, in our current implementation, the block based CBR approach described previously incurs less overhead, so we use it for our experiments.

Page Deltas: To use page deltas, a second index into the page cache is created based on the page address. If the sender finds the page in the cache based on its address, then the current and cached pages are XOR'd to find the different bits. The XOR'd page is then run length encoded (RLE). Since the RLE algorithm can potentially increase the size of the data sent if a page has been significantly modified, only pages with a RLE size less than a threshold are sent in their compressed form.

Migration Signaling: CloudNet uses a per-page meta packet that indicates to the receiver whether the full page, a cache index, or an index plus a page delta is going to be sent. The meta packet contains a 32 bit cache index and a 16 bit delta length. A negative index indicates the page missed both caches and will be sent in full after the meta data; the value of the index is used by the destination host to know where to add the new page to its cache. This ensures that the caches maintained at the migration source and destination remain synchronized.

A positive index means the page can be found in the cache. If the delta length is zero, then it was a perfect CBR match, otherwise the receiver will wait to receive the run length encoded page delta and apply it to the cached page before copying it into the new VM's memory. If CBR is used at sub-page granularity, then one meta data block is generated per sub-page, but they can be aggregated and sent for each page to reduce the number of small transmissions.

6 Evaluation

Our evaluation explores the impact of network conditions on migration performance, and evaluates the benefits of each of our optimizations. We then compare the shared and replicated storage techniques. Finally, we study the performance of multiple simultaneous migrations between our real data center sites.

6.1 Evaluation Platform

6.1.1 Testbed Setup

We have evaluated our techniques both within a local testbed using a network emulator to mimic a WAN environment, and between two data center sites, spread across the United States, and interconnected via an operational network.

Local Testbed: Our local testbed consists of a pair of Sun servers with dual quad-core Xeon CPUs and 32GB of RAM. Each server is connected to a Juniper M7i router, and VPLS connectivity is established between the two routers. 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. This testbed allows us to explore VM migration over a variety of network conditions.

Data Center Prototype: We have also deployed CloudNet across two data centers in Illinois and Texas. 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 [8]. The servers and routers have the same specifications as those on the local testbed, but we have access to two servers at each site. Network characteristics between sites are variable since the data centers are connected over the WAN; we measured an average round trip latency of 27 msec and a throughput of 464 Mbps between the sites.

6.1.2 Applications and Workloads

Our evaluation studies three types of business applications. We run each application within a Xen VM granted 1GB of RAM, and allow each application to warm up for at least twenty minutes before performing a migration.

SPECjbb 2005 is a java server benchmark that emulates a client/server business application [23]. 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.

6.2 Migration in CloudNet

This section discusses the benefits provided by our optimizations. We first analyze migration performance using VMs allocated 1GB of RAM running each of our three applications. To focus on memory migrations in low bandwidth scenarios, we create the VMs on a shared storage device and configure the network emulator to mimic a 100 Mbps link with 20 msec round trip delay; this represents a reasonable expectation of the network capacity available to a single application in a well provisioned data center.

Figure 8 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 for each of the applications tested. For the kernel compile and TPC-W, the combination of optimizations also provides a reduction in VM pause time (Figure 8(a) and (b)). Table 1 lists the absolute performance of migrations with the default Xen code and with CloudNet's optimizations.

	Data Tx (GB)		Tot 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: Data transferred, total time, and pause time during migrations, with and without optimizations, over a 100Mbps link with shared storage.

We discuss the benefits of each optimization individually in the following sections.

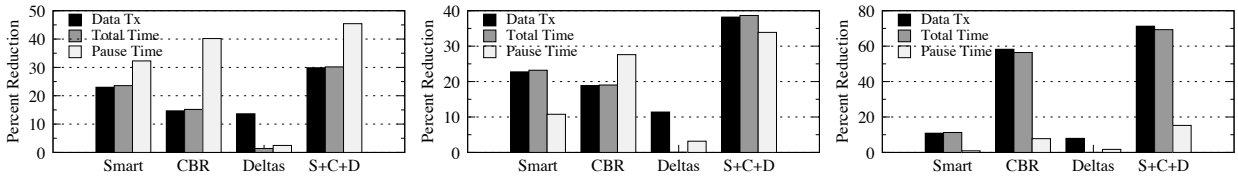
6.2.1 Smart Stop & Copy

CloudNet's Smart Stop optimization reduces the data transferred and total time in Kernel Compile and TPC-W by over 20%, but has a smaller impact on SPECjbb (Figure 8). To understand the difference in behavior of these applications, Figure 9 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 the TPC-W and SPECjbb migrations. 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.

The SPECjbb benchmark has a different memory behavior which limits the benefit of the Smart Stop optimization. The migration sends less than 5MB per iteration because SPECjbb has a relatively small working set which it very rapidly modifies. The Xen migration algorithm detects that these pages are being modified in consecutive iterations and delays sending them as it predicts that they will need to be retransmitted later. Since SPECjbb defers most of its sends until the final iteration, the Smart Stop optimization provides only a minor improvement when it eliminates the intermediate rounds.

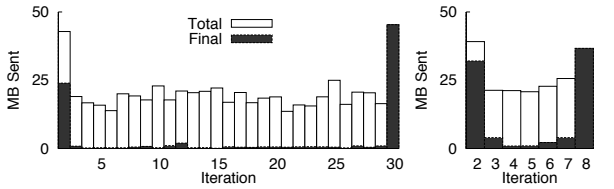
The Smart Stop optimization can reduce the pause time of the kernel compile by over 30% (Figure 8(a)). This is because the kernel compile exhibits a high variance in the rate at which memory is modified (Figure 4). The Smart Stop algorithm is thus able to pick a more intelligent iteration to conclude the migration at, minimizing the pause time since less data needs to be sent in the final iteration.

These results indicate that the large number of iterations used in the default Xen migration code is unnecessary and inefficient if bandwidth is limited. Smart Stop reduces total migration time by using fewer iterations and can reduce the

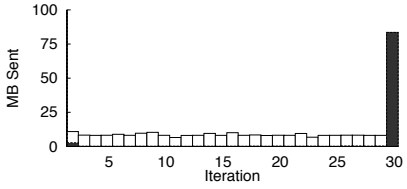


(a) Kernel Compile (b) TPC-W (c) SPECjbb

Figure 8: CloudNet's optimizations substantially reduce data sent, and lower both the total and pause time during migrations.



(a) TPC-W Default Xen (b) TPC-W Smart Stop



(c) SPECjbb Default Xen

Figure 9: 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. SPECjbb sends less data per iteration, reducing the benefit of Smart Stop.

application experienced downtime by choosing an intelligent time to pause.

6.2.2 Redundancy Elimination

The Content Based Redundancy optimization provides benefits to all of the tested applications by eliminating the redundant data during memory transfer. However, the applications have different levels and types of redundant memory. Figure 10 shows the redundancy found in each application when dividing each memory page into four 1KB blocks. SPECjbb exhibits the largest level of redundancy; however, the majority of the redundant data is from zero pages. In contrast, Kernel has about 13% redundancy, of which less than half is zero pages.

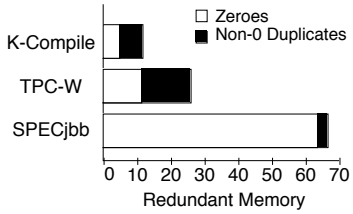


Figure 10: Each application has different types of redundancy.

The CBR optimization eliminates this redundancy, providing substantial reductions in the total data transferred and migration time (Figure 8). 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.

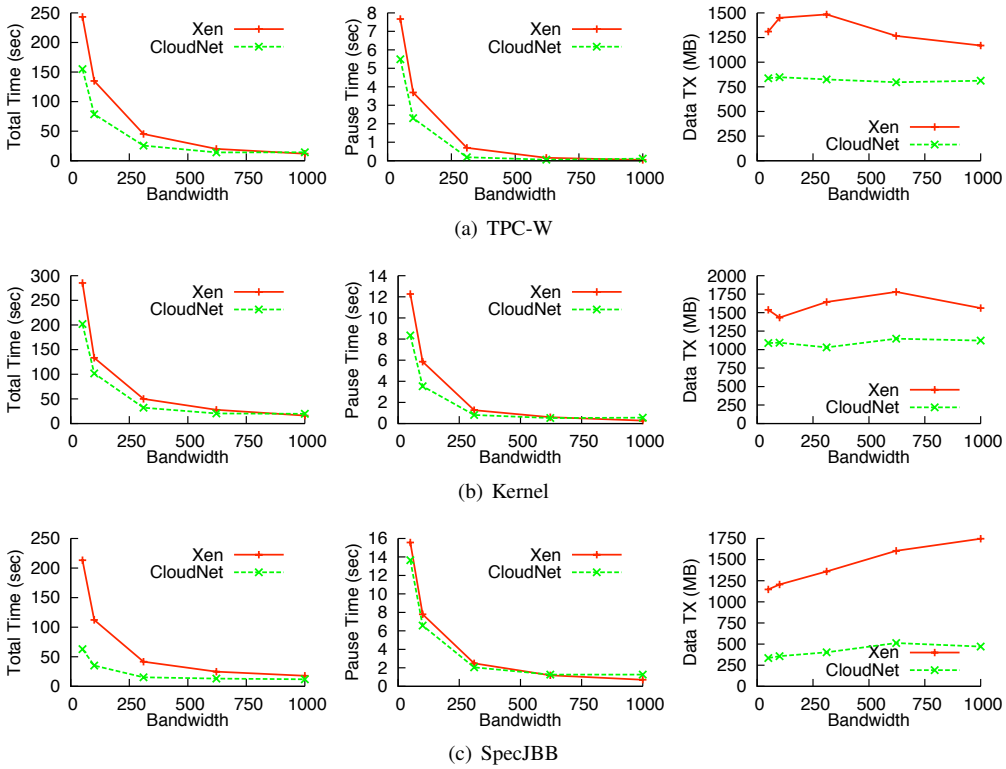
CloudNet's Content Based Redundancy optimization improves overall performance by eliminating redundant data transfers. This reduces total time and can also reduce pause time if the pages sent in the final iteration have similar content to previous iterations.

6.2.3 Page Deltas

In Figure 8, the use of Page Deltas provides a smaller improvement compared to the other optimizations because the address based cache can only be used from the second iteration onwards. The first iteration makes up a large portion of the total data transferred since during this iteration the majority of a VM's memory–containing less frequently touched pages–is transferred.

Table 2 shows the amount of memory data transferred during the first and remaining iterations during migrations of each application. All applications transfer at least 877 MB of data during the first iteration when the cache cannot be used. During iterations 2 to 30, the Page Delta optimization significantly reduces the amount of data that needs to be sent. For example, TPC-W sees a reduction from 487 MB to 315 MB, 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 provide both bandwidth and time reductions.



(a) TPC-W

(b) Kernel

(c) SpecJBB

Figure 11: Decreased bandwidth has a large impact on migration time, however, CloudNet's optimizations reduce the data transferred, lowering total and pause time.

	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.

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, the Smart Stop optimization limits the number of useless iterations and combines with both the CBR and Page delta techniques to minimize the pause time during the final iteration.

6.3 Impact of Network Conditions

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

6.3.1 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 use a network emulator to 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 11 compares the performance of default Xen to CloudNet's optimized migration system. Decreased bandwidth lowers performance for both applications, but our optimizations provide significant benefits, particularly in low bandwidth scenarios.

CloudNet's optimizations reduce the amount of data that needs to be transferred during the migration because of both caching and the lower number of iterations. As bandwidth decreases, TPC-W sends a steady amount of data in CloudNet. In contrast, SpecJBB's data transfer increases because its memory writes are restricted to a relatively small portion of memory that is repeatedly dirtied. As bandwidth decreases, a larger portion of the SPECjbb working set is dirtied before being sent, causing the migration algorithm to refrain from sending those pages until the final iteration. In both cases, CloudNet's optimizations still provide substantial reductions in the amount of data sent.

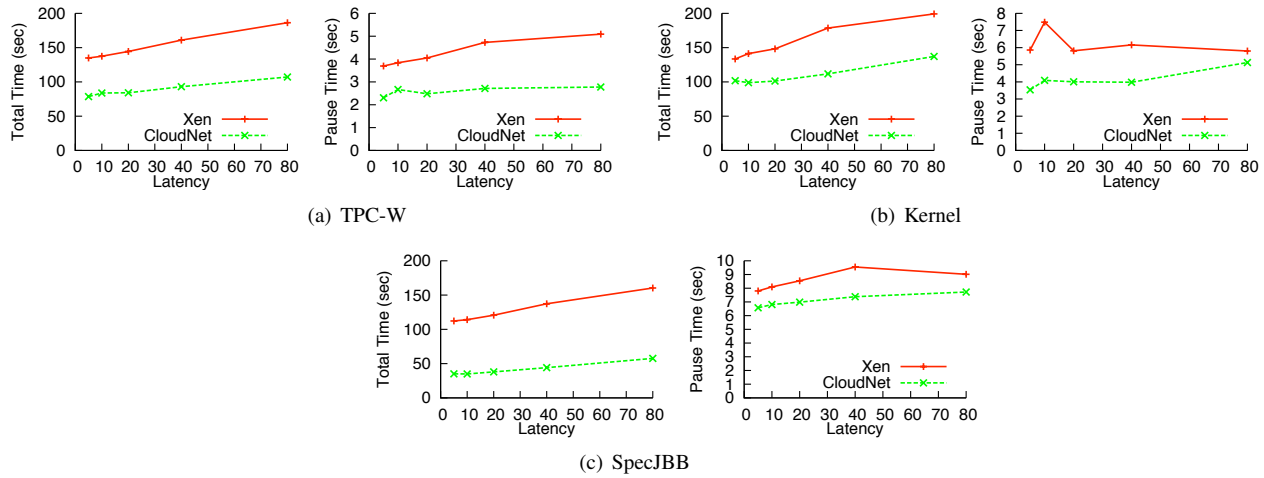


Figure 12: Increased latency has only a minor impact on CloudNet migration performance.

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 provides reductions in data transferred, but does not significantly affect the total 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.

6.3.2 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 latency introduced by the network emulator over a 100Mbps link. We vary the delay in each direction from 5 to 80msec. The default TCP settings installed with Linux perform *very* poorly under high latency; we have adjusted the TCP settings by switching to the "scalable" control algorithm [19], and increasing the maximum send and receive buffer sizes to 32 MB.

Even with optimized TCP settings, the slow start in TCP causes performance to decrease as latency rises. CloudNet's optimizations still provide a consistent improvement regardless of link latency. Although our optimizations increase the number of small packets sent as meta data during a migration, CloudNet does not require any additional application level acknowledgements which would cause performance to suffer as latency rises.

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, our optimizations enable efficient VM migrations in much lower bandwidth and higher latency scenarios.

6.4 Storage Migration

This section evaluates how the different storage synchronization techniques impact application performance, as well as optimizations for migrating disk state.

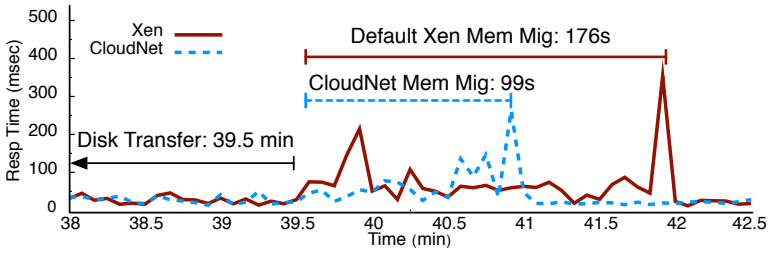
6.4.1 Application Performance

CloudNet seeks to minimize the total migration time both to increase the flexibility with which migration decisions can be made and to decrease the period of time where VM performance is decreased by migration overheads. This experiment studies application performance when migrating VMs with storage over a low bandwidth link.

We migrate a VM running the TPC-W application and configure the network to 100 Mbps with a 20 msec roundtrip latency. When the migration is initially scheduled, the DRBD subsystem begins the initial bulk transfer of the VM's disk using asynchronous replication. To prevent the disk transfer from impacting application performance, the synchronization rate is limited to 4MB per second, resulting in a storage migration period of 39 minutes to transfer the VM's 10GB disk.

Figure 13(a) shows how the response time of the TPC-W application is affected during the final two minutes of the storage transfer and during the subsequent memory migration. The response time includes the queuing and processing time within the TPC-W VM, but does not include the WAN latency to and from the client. During the disk transfer period, the asynchronous replication imposes only a modest overhead; average response time is 31 msec compared to 20 msec in the non-replicated case.

During the migration itself, response time increases to 86 msec, 4.25 times the response time during ordinary operation. This is in part due to the switch to synchronous replication, but also because of the additional memory overhead caused by the Xen migration code's use of shadow page tables to detect which pages need to be resent. While both de-



(a) Application Performance

Optimization	Tx Savings
Empty Blocks	3.4 GB
CBR-100	5.1 GB
CBR-1024	6.4 GB

(b) Disk Transfer Optimizations

Figure 13: (a) CloudNet minimizes the window of reduced performance during migrations. (b) Eliminating redundancy in disk state transfers can significantly reduce the amount of data sent. Data transfer savings are out of a 10GB disk.

	Mem TX (MB)	Time (s)	Paused (s)
Xen	13,308	245	6.1
CloudNet	4,155	87	3.1

Table 3: CloudNet significantly reduces the cost of running four simultaneous migrations because its optimizations reduce the amount of memory data that needs to be sent.

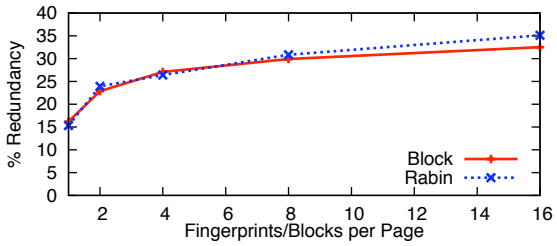


Figure 14: CloudNet's block based redundancy elimination performs similarly to a Rabin fingerprint based technique.

fault Xen and CloudNet migrations suffer this performance penalty, CloudNet's optimizations reduce the window of decreased performance from 176 to 99 seconds. After CloudNet's full storage and memory migration completes at 40.9 minutes, the disk replication is completely disabled, bringing the response time back to the original 20 msec.

6.4.2 Storage Migration Optimizations

Storage migration can be the dominant cost during a migration. The DRBD replication system used by CloudNet already performs some optimizations during the migration by only sending deltas of blocks that are updated, and by not sending blocks that are completely empty. 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. The table in Figure 13(b) shows the benefits of eliminating empty blocks from the storage migration, as well as applying the CBR optimization with different cache sizes. We find that even a small 100MB cache increases the savings from redundancy elimination to 5.1GB, and the use of a 1GB cache provides a further 1.3GB in data transfer savings.

6.5 Simultaneous Migrations between Data Centers

While our optimizations are targeted at low bandwidth scenarios, they still provide benefits during data center to data center migrations by (i) reducing the data transferred (lowering bandwidth costs) and (ii) allowing multiple migrations to occur simultaneously.

In this experiment, we demonstrate a migration of a small cluster of four VMs running a distributed kernel compilation from the CloudNet data center in Texas to the data center in Illinois. The total bandwidth available between the two sites is 465Mbps with a 27 msec round trip latency. Each of the VMs has a 10GB disk (of which 6GB is in use) and is allocated 1.7GB of RAM, similar to a "small" VM instance on Amazon EC2[2].

We use CloudNet's DRBD storage system to simultaneously migrate the storage of all VMs, transferring a total of 24.1 GB of data after DRBD skips the empty disk blocks. The disk transfers takes 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 VM migration code, which reduces the memory migration time to only 87 seconds, and halves the pause time from 6.1 to 3.1 seconds. Table 3 compares the performance of the memory migration in each case, showing that CloudNet reduces the data transferred during the memory migrations from 13GB to 4GB. If the disk based CBR optimizations were also used, the total data transferred to move all four VMs, including their storage and memory data, would fall from 37.3GB in the default Xen case to 16.1GB when using CloudNet's optimizations.

[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 proportionally.

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 57%, saving 20GB worth of bandwidth costs.

6.6 Comparison to Rabin Fingerprints

While all our results thus far have used the block-based CBR technique, CloudNet also supports Rabin fingerprints. To compare these approaches, we record a trace of all the memory pages sent during a migration of our TPC-W virtual machine, without any of our optimizations, over a 100Mbps link with 10 msec latency. We then analyze this trace offline to compare the amount of redundancy detected with CloudNet's block based CBR to the Rabin fingerprint based approach implemented as described in [5].

Both approaches have a parameter which affects the granularity at which redundancy is found: the number of blocks that a page is divided into in the block based approach, and the number of fingerprints stored per page when using Rabin fingerprints. Figure 14 shows how redundancy detection changes as we vary the size of blocks in CloudNet or the number of Rabin fingerprints stored per page. Both approaches perform similarly at four fingerprints per page, the parameter used in our experiments. However, the Rabin approach scales better when using a larger number of fingerprints because it can find redundancy at arbitrary offsets within a memory page.

An effective redundancy elimination technique must both find a large amount of redundant data and incur only a small processing overhead. Our block based tool takes 7.3 seconds to parse the 1.5GB memory trace. Our Rabin tool takes 119 seconds to analyze the same trace. However, we believe this is partly due to inefficiencies in the Rabin library used [21]; CloudNet's architecture can easily be transitioned to use the Rabin based redundancy elimination once the implementation has been optimized.

7 Related Work

Cloud Computing: Armbrust et al provide a thorough overview of the challenges and opportunities in cloud computing [6]. There are several types of cloud computing platforms, but we focus on Infrastructure as a Service (IaaS) platforms which rent virtual machine and storage resource to customers. Microsoft Azure, VMware vCloud, and the Amazon Elastic Compute Cloud are some of the IaaS platforms from major vendors.

Private Clouds & Virtual Networks: The VIOLIN and Virtuoso projects use overlay networks to create private groups of VMs across multiple grid computing sites [22, 25]. VIOLIN also supports simultaneous 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 each end host.

We initially proposed our vision for Virtual Private Clouds in [29]. 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 "hoses model" [12] and are commonly used by enterprises. Provider based VPNs can provide 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 they simplify WAN migration, have lower overheads, and can provide additional services 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 [20, 9, 16]. Work has been done to optimize migration within the LAN by exploiting fast interconnects that support remote memory access technology [15]. Jin et al. have proposed using memory compression algorithms to optimize migrations [17]. 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 recently announced limited support for WAN migration, but only under very constrained conditions: 622 MBps link bandwidth and less than 5 msec network delay [3]. 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 the network supports [7]. The VM Turntable Demonstrator showed a VM migration over intercontinental distances with latencies of nearly 200 msec; they utilize giga-

bit 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 [14]; 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.

8 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 local testbed and in a prototype deployed across two data centers separated by over 1,200KM. During simultaneous migrations of four VMs between these data centers, CloudNet's optimizations reduce memory transfer time by 65%, and can save 20GB in bandwidth for storage and memory migration.

References

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

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

[3] 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, September 2009.

[4] Amazon elastic computing cloud. `http://aws.amazon.com/ec2`.

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

[6] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

[7] Robert Bradford, Evangelos Kotsovinos, Anja Feldmann, and Harald Schiberg. 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.

[8] Xu Chen, Z Morley Mao, and Jacobus Van der Merwe. ShadowNet: a platform for rapid and safe network evolution. In *USENIX Annual Technical Conference*, 2009.

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

[10] Brendan Cully, Geoffrey Lefebvre, Dutch Meyer, and Anoop Karollil. Remus: High availability via asynchronous virtual machine replication. In *NSDI 08*, 2008.

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

[12] N. G. Duffield, Pawan Goyal, Albert Greenberg, Partho Mishra, K. K. Ramakrishnan, and Jacobus 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] Elasticvapor blog: Virtual private cloud. `http://www.elasticvapor.com/2008/05/virtual-private-cloud-vpc.html`.

[14] Eric Harney, Sebastien Goasguen, Jim Martin, Mike Murphy, and Mike Westall. The efficacy of live virtual machine migrations over the internet. In *Proceedings of the 3rd international workshop on Virtualization technology in distributed computing*, pages 1–7, Reno, Nevada, 2007. ACM.

[15] Wei Huang, Qi Gao, Jiuxing Liu, and Dhabaleswar 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.

[16] Microsoft hyper-v server. `www.microsoft.com/hyper-v-server`.

[17] Hai Jin, Li Deng, Song Wu, Xuanhua Shi, and Xiaodong Pan. Live virtual machine migration with adaptive memory compression. In *Cluster*, 2009.

[18] G Milos, DG Murray, S Hand, and M Fetterman. Satori: Enlightened page sharing. In *USENIX Annual Technical Conference*, 2009.

[19] R. Morris. Scalable TCP congestion control. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1176–1183 vol.3, 2000.

[20] Michael Nelson, Beng-Hong Lim, and Greg Hutchins. Fast transparent migration for virtual machines. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, 2005.

[21] Sliding window based rabin fingerprint computation library. `www.cs.cmu.edu/~hakim/software`.

[22] P. Ruth, J. Rhee, D. Xu, R. Kennell, and S. Goasguen. Autonomic live adaptation of virtual computational environments in a multi-domain infrastructure. In *ICAC '06: Proceedings of the 2006 IEEE International Conference on Autonomic Computing*, Washington, DC, USA, 2006.

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

[24] Neil T. Spring and David Wetherall. A protocol-independent technique for eliminating redundant network traffic. *SIGCOMM Comput. Commun. Rev.*, 30(4):87–95, 2000.

[25] Ananth I. Sundararaj and Peter 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] Franco Travostino, Paul Daspit, Leon Gommans, Chetan Jog, Cees de Laat, Joe Mambretti, Inder Monga, Bas van Oudenaarde, Satish Raghunath, and Phil Yonghui Wang. Seamless live migration of virtual machines over the MAN/WAN. *Future Generation Computer Systems*, 22(8):901–907, October 2006.

[28] Carl A Waldspurger. Memory resource management in VMware ESX server. In *OSDI '02: Proceedings of the 5th symposium on Operating systems design and implementation*, page 181194, New York, NY, USA, 2002. ACM.

[29] 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 Topicsin Cloud Computing (HotCloud), San Diego, CA*, June 2009.