Edge Computing for Legacy Applications

Mahadev Satyanarayanan, Thomas Eiszler, Jan Harkes, Haithem Turki, and Ziqiang Feng
Carnegie Mellon University

***Abstract*—Edge computing was motivated by the vision of new *edge-native applications* that are compute-intensive, bandwidth-hungry, and latency-sensitive. We show how infrastructure deployed for such futuristic applications can also benefit virtual machine (VM)-encapsulated Windows or Linux closed-source legacy applications. We present a new capability for legacy applications called *edge-based virtual desktop infrastructure (EdgeVDI)*, and discuss example use cases that it enables.**

The essence of Edge Computing, as shown in Figure 1, is a new tier (Tier-2) of server-class computers called *cloudlets* that sit between mobile and IoT devices (Tier-3) and the cloud (Tier-1). LAN-quality networking between Tier-3 and Tier-2 is essential in this architecture. By processing data on Tier-2 rather Tier-1, network round-trip time (RTT) is reduced for latency-sensitive applications such as augmented reality (AR). WAN bandwidth demand is also reduced from dense deployments of IoT devices such as video cameras. As cloudlet deployments grow, many new *edge-native applications*[14] will emerge. However, they will not displace all of today's applications—many old applications will remain valuable.

In this article, we explore how edge computing can benefit unmodified *legacy applications* that were originally written for a personal computing environment. We restrict this term to authoring tools that are used by professional workers. This is in contrast to information consumption tools that are used by the public at large, such as those for video viewing, web browsing, social media, etc. The term also excludes smartphone apps. Well-known examples of authoring tools include the Microsoft Office Suite, Adobe Acrobat, Adobe Photoshop, GIMP, AutoDesk Revit, and software development tools such as the gcc compiler. Modern versions of these tools are used today on Windows, MacOS, and Linux desktops and laptops.

Digital Object Identifier 10.1109/MPRV.2020.3026229
Date of publication 19 October 2020; date of current version 9 November 2020.

Published by the IEEE Computer Society

They are the primary vehicles through which many millions of professional workers, such as architects, engineers, lawyers, creative artists, and software developers, deliver their productive value into the global economy. Although the market for authoring tools is much smaller in relative terms than the market for information consumption tools (tens of millions of users, rather than billions), it is still a large market in absolute terms. It is also a market with high-profit margins because of the productive value of professional users.

The central message of this article is that cloudlets deployed for future edge-native applications can also benefit legacy applications. To achieve this benefit, *Virtual Desktop Infrastructure (VDI)* products, such as VMware Horizon, Citrix XenDesktop, and Microsoft Windows VDI need to be liberated from their current restriction to the LAN-connected campus of an enterprise. This restriction is imposed today because the low latency and high bandwidth required by the remote desktop protocols (RDPs) of these systems can only be met on a LAN. Edge computing enables this requirement to be met by moving the virtual desktop on demand from the enterprise private cloud (Tier-1) to a cloudlet close to a mobile user (Tier-2). A stateless Tier-3 device (possibly borrowed from the user's environment) can then be used to access the virtual desktop at low latency and high bandwidth. This extension of VDI, called edge-based virtual desktop infrastructure *(EdgeVDI),* creates the illusion of a user's desktop "following him around" on the Internet as he travels.

Today, reliable access to a legacy application during wide-area mobility can only be achieved on a laptop that is carried by the user. Instead of carrying hardware that adds weight, consumes space, and is vulnerable to loss, theft, and damage, the user can leverage EdgeVDI to rapidly instantiate a virtual desktop nearby. This alternative has many advantages, as discussed in this article.

TYRANNY OF LEGACY

The term "legacy" characterizes past choices that have an inordinate influence on the present and future. Those past choices may have been optimal at the time and context in which they were made. However, the growth of a rich and valuable ecosystem around those past choices makes change impractical. Perhaps the best-known example of the tyrannical influence of the past is the illogical layout of keyboards in modern computing systems. This layout was chosen for mechanical typewriters in the 19th century to reduce the likelihood of adjacent keys sticking together during fast typing. Although that rationale has long ceased to be relevant, the keyboard layout has remained the same. Once a critical mass of people had learned touch typing on this layout, their resistance to change proved to be an impenetrable barrier. Another example is which side of the road one drives on. Early in the history of automobiles, each country made a choice. Over time, that choice became harder to change because of investments in traffic and vehicular infrastructure, as well as the traffic-related safety habits learned by the public.

For software, both learning by users and the evolution of a rich ecosystem of compatible software and data formats constitute legacy. For many enterprises that use the software, incompatible change threatens to be highly disruptive. This is especially true in the case of authoring tools, as mentioned in the "Introduction" section. There is a high business value in creating new products that remain compatible with legacy software. Perhaps the best example of such business value is modern IBM mainframes, whose legacy reaches back half a century. Mainframes accounted for a substantial fraction of IBM's profits as recently as the fourth quarter of 2019.[†] In early 2020, the website for these products[5] states: "The IBM Z family maintains full backward compatibility. This means that current systems are the direct, lineal descendants of System/360 announced in 1964, and System/370 from the 1970s. Many applications written for these systems can still run unmodified on the newest IBM Z system over five decades later."

For authoring tools of the personal computing era, Microsoft Windows is the dominant legacy environment. While software created for Apple MacOS, Linux, and other flavors of Unix

[†]"Big Blue's fourth-quarter earnings boosted by Red Hat deal and mainframe computers," Wall Street Journal, January 23, 2020

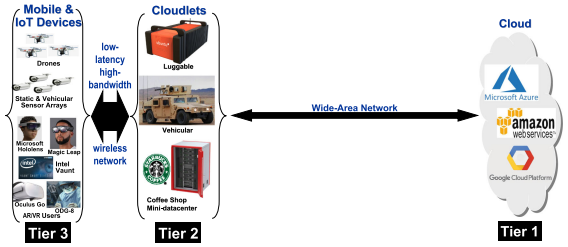

Figure 1. Three-tier model of edge computing.

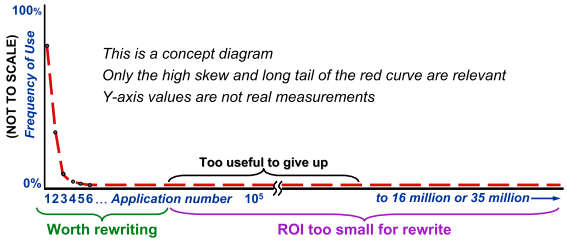

Figure 2. The long tail of legacy applications.

are also important, we will focus on Windows for brevity. The number of distinct Windows applications that have been created lies between 16 million[‡] and 35 million.[§] Even the smaller of these estimates is an enormous number. In the context of Figure 1, the simplest way to represent personal computing is to remove Tier-1 and Tier-2, and to replace the devices at Tier-3 by desktops and laptops. Cloud computing adds back Tier-1 to that figure.

There have been a number of efforts to rewrite widely used authoring tools in way that leverages cloud computing, but retains their user-facing and data-facing functionality and interfaces. Microsoft Office 365 is the best known of these transformations. Such rewriting requires crisp user interactions to be preserved, while splitting a monolithic Tier-3 application into a Tier-1 back-end separated by the high latency of the Internet from a Web browser at Tier-3. While this effort can be successful, it is very expensive and can only be profitable for the most widely used applications. For less widely used applications, the return on investment (ROI) is too low for the effort to be worthwhile.

Figure 2 illustrates the dilemma faced by the users and vendors of legacy authoring tools. Although we do not have sufficient empirical data to plot the exact shape of this curve (e.g., Zipf's Law, a Pareto distribution, or an exponential are all plausible), its salient features are high skew and an extremely long tail. In other words, a relatively small number of applications, possibly less than one percent of the 16 million or 35 million Windows applications mentioned earlier, are very heavily used. Only for these applications is the ROI of rewriting for a cloud-based implementation worthwhile. For the remaining millions of applications, many of which remain important to users and enterprises, rewriting is unlikely to be profitable. VDI, described in the following section, is the solution that brings the benefits of cloud computing to them.

[‡]2015 Microsoft blog entry https://blogs.windows.com/windowsexperience/2015/04/29/welcoming-developers-to-windows-10/
[§]2018 Microsoft blog entry https://blogs.windows.com/windowsexperience/2018/11/13/windows-10-quality-approach-for-a-complex-ecosystem/

VDI AND EDGE VDI

VDI uses virtual machine (VM) technology to bring the benefits of cloud computing to closed-source legacy applications. At its simplest, using VDI involves the following three steps:

- encapsulating an application and its operating system in a VM image;
- launching and managing an instance of this VM image in the cloud;
- interacting with the VM instance using an RDP from a Tier-3 device (e.g., a tablet or a desktop window).

Even a highly optimized RDP has to precisely capture the tightly interleaved interactions between a stateful computing environment and a stateless keyboard, mouse, and display. These interactions are very frequent and fine-grain (i.e., "chatty") because graphical user interface (GUI) metaphors have coevolved with personal computing to take full advantage of extremely low latency and high bandwidth between the user and computer. Hence, VDI is only supported on LANs today. EdgeVDI overcomes this restriction by executing a VM instance at Tier-2 rather than Tier-1. Since LAN connectivity is guaranteed between Tier-3 and Tier-2, EdgeVDI remains usable even when its user is far from home (see Figure 3).

A key challenge for EdgeVDI is that delivering a virtual desktop to its optimal cloudlet can take many tens of minutes at typical WAN

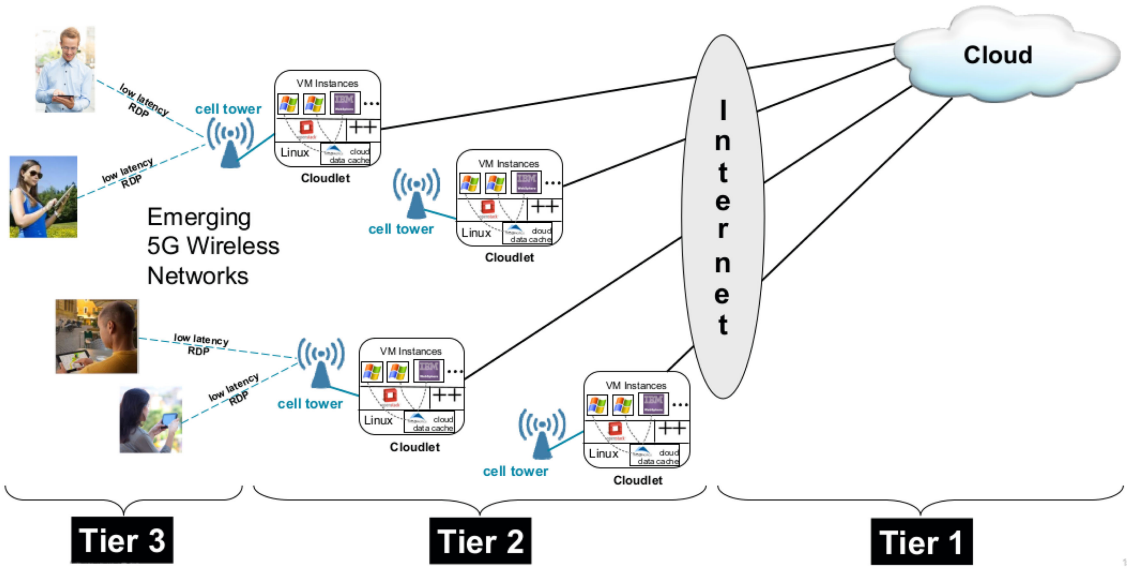

Figure 3. EdgeVDI: Leveraging edge computing for VDI.

bandwidths (few tens of megabit per second) and VM image sizes (few tens of gigabytes). This is an unacceptably long delay for a mobile user to wait before interactions begin. EdgeVDI keeps this delay to a minimum by launching the VM instance wherever the current VM image of the virtual desktop is available. This could be at the cloudlet where the user last ran his virtual desktop, or in the cloud. The initial interactions with this VM instance are suboptimal since the RDP is executed over a WAN. However, during these interactions, the VM instance is transparently moved to a cloudlet close to the user. When the move is complete, interactions are redirected to the nearby cloudlet. Interaction quality is now optimal since the RDP is executed over a LAN. A fast move shortens the period of suboptimal interactions.

EdgeVDI's mechanism for moving a VM instance without disrupting execution resembles *live migration*[3], which is extensively used within the cloud for load balancing and system maintenance. However, live migration requires high bandwidth (typically 10 Gb/s or better) that is only available within a data center or on dedicated long-distance fiber links. This is more than two orders of magnitude higher than the 5–25 Mb/s bandwidth available over typical WANs today. We have, therefore, completely redesigned and reimplemented the live migration concept in a mechanism called *VM Handoff*.

VM Handoff was originally conceived as a cross-cloudlet load balancing tool for computations offloaded from Tier-3 devices. This article focuses on how this mechanism can be leveraged for a very different goal, namely Internet-wide mobility of virtual desktops. A detailed description and performance evaluation of VM Handoff was presented in earlier work.[4] We present a summary of its implementation in the next paragraph, and follow with a more detailed description of how it has been extended for use in EdgeVDI.

VM Handoff relies on three broad design principles. First, we use deduplication, compression, and delta-encoding to ruthlessly eliminate avoidable data transfers. Second, since network bandwidth is a precious resource, we strive to keep its utilization at the highest possible level. Third, we adapt at fine time granularity to network bandwidth and cloudlet compute resources. VM Handoff allows continued use of a migrating VM instance at the source cloudlet. The VM state that is modified during handoff is tracked and transferred in a second pass. This process is repeated for several iterations until the final pause of the source and switch to the destination (see Figure 4). This complexity is completely invisible to the VM instance, and, hence, to the user and to legacy applications.

EdgeVDI uses a command line program called nephele-client to control the placement and

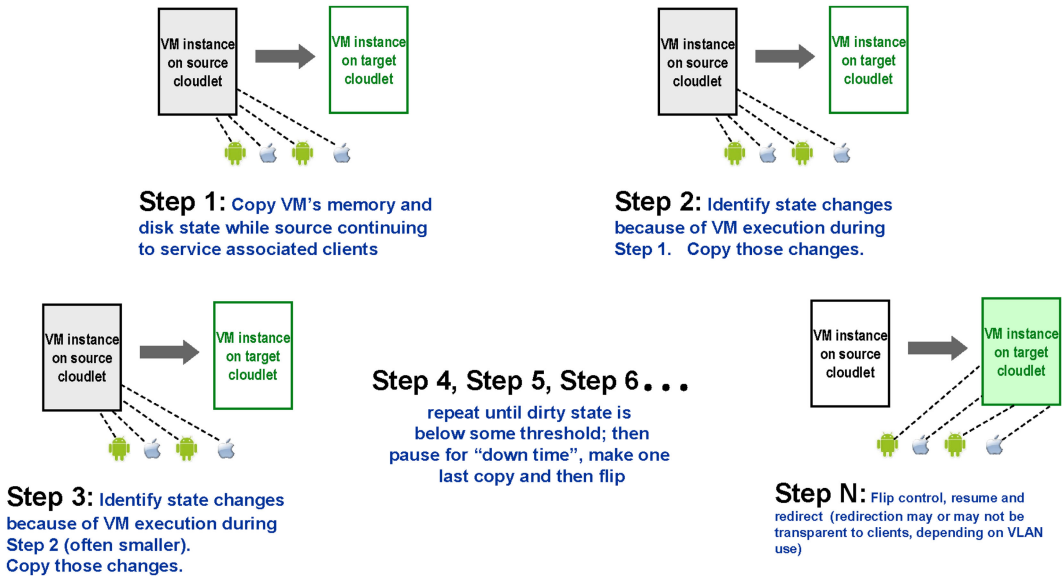

Figure 4. Successive iterations of VM handoff.

movement of VM instances via VM Handoff. Linux already provides a program called virt-manager for defining VM images, managing the life cycle of VM instances created from them and integrating RDP support (see Figure 5). nephele-client can be viewed as an extension of virt-manager that manages the distributed aspects of life cycle management, such as moving a VM instance from one machine to another. Our long-term plan is to integrate the functionality of nephele-client into virt-manager. Table 1 describes a sample of the commands available via nephele-client.

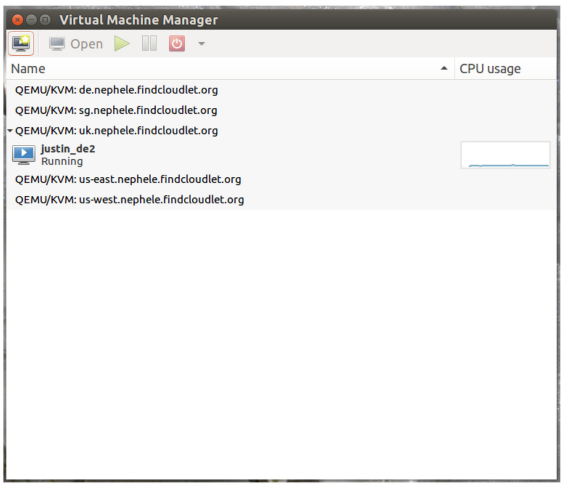

Figure 5. Screenshot of VirtManager in Linux.

EXAMPLE USE CASES

Zero-Pound Laptop

Business travelers carry laptops today because a smartphone or tablet is rarely sufficient for serious work via authoring tools. Alas, laptops contribute nontrivial weight and space to carry-on luggage. They are also vulnerable to loss, theft, and physical damage during travel. EdgeVDI offers a way to avoid carrying a laptop. Stateless thin client hardware that runs an RDP could be made available anywhere that a business traveler may access his virtual desktop, including inside aircraft and in hotel rooms. In other words, the *ubiquity* of EdgeVDI can replace the portability of a laptop. While the business models and thin client products to support this still need to be developed, its technical feasibility is not in doubt.

What kind of user experience can a traveler expect from EdgeVDI? Table 2 illustrates its performance for a Windows 7 VM with a latency-sensitive application (Microsoft Paint). The network between source and destination cloudlets had a RTT of 200 ms and a bandwidth of 20 Mb/s. VM Handoff was triggered after roughly 50 s of interaction with the distant cloudlet. Table 2 describes that interactions continued for roughly 50 more seconds, after which the VM instance was unresponsive for roughly a minute. The user then enjoyed LAN-quality RDP

Table 1. Examples of nephele-client commands.

```
nephele-client <host> <command> <arguments>
```

`<command>` is one of

`handoff`	Handoff a running instance to another `nephele` node
`image`	Manage base VM images
`images`	List base images tracked by `nephele`
`ps`	List running VM instances known to `nephele`
`run`	Instantiate a VM from an existing snapshot
`snapshot`	Manage VM snapshots
`snapshots`	List snapshots known to `nephele`

(a) Top-level Commands

```
nephele-client localhost snapshots
SNAPSHOT                        IMAGE ID            CREATED
/root/t2.zip                    edbe05049453        2020-02-28 15:23:42.956169
/root/rpcbuild.zip              edbe05049453        2020-03-03 08:49:12.872102
/root/rpcbuild_1.zip            edbe05049453        2020-03-03 14:22:10.750467
```

(b) Listing `nephele` Snapshots on Local Machine

```
nephele-client localhost run [-d] [-p PORTS] snapshot title
```

`snapshot`	Disk/memory snapshot to instantiate from
`title`	Title of the VM instance to display in `virt-manager`
`-d`	(Optional) Ignore memory, create delta from disk blocks only. (default = `False`)
`-p PORTS`	(Optional) Comma separated list of ports to forward from host to guest (e.g., `-p 80,443,8080,8443`)

(c) Launching a `nephele` Instance on Local Machine

```
nephele-client <source-host> handoff <title-of-running-instance> <destination-host>
```

(d) Moving a `nephele` Instance from Source Machine to Destination Machine

Table 2. EdgeVDI for Microsoft Paint on Windows 7.

	Run 1	Run 2	Run 3
Time before trigger	50	48	56
Total VM Handoff time	101	103	104
Down time	57	56	60
Memory+Disk sent on wire	101	111	103
Memory: Dirtied	1067	696	1084
Memory: Sent by value	380	411	398
Memory: Sent by reference	214	206	211
Memory: Zeros	473	80	477
Disk: Dirtied	13	17	17
Disk: Sent by value	5	6	6
Disk: Sent by reference	7	11	11
Disk: Zeros	0	0	0

All times in seconds, rounded to nearest second
All sizes in MB, rounded to nearest MB
Network RTT = 200 ms Bandwidth = 20 Mb/s
VM configuration: 4 GB RAM, 32 GB virtual disk, 4 vCPUs.

interactions. Only about 100 MB of data was transmitted over the network, even though 696–1084 MB of memory was dirtied by user interaction. The table describes how deduplication, compression, and detection of zero-filled pages each contributed to this substantial savings.

Moving Legacy Code to Big Data

EdgeVDI can also be valuable in use cases that have nothing to do with the physical mobility of users. The ability to efficiently move legacy code and relevant execution state to any point on the Internet is a valuable asset in processing large, dispersed datasets. In that setting, bringing data to the compute (for example, via caching in a distributed file system) may involve much larger amounts of data transfer than moving a VM instance to each of the datasets. An application-specific solution is possible, but it would require rewriting the applications to handle remote communication, data compression, fault tolerance, etc. EdgeVDI offers a low-friction,

off-the-shelf solution that transparently makes legacy programs "distributed." We describe two examples below.

Legacy Scientific Applications *Scientific reproducibility* is a topic of increasing importance. Today, many publications encourage authors to open source and freely share their research software. EdgeVDI can be a valuable tool in this context. For example, a program written in MATLAB or Python can be encapsulated in a VM image, along with specific versions of the interpreter, workspaces, libraries, and operating system. EdgeVDI can be used to move an instance of this VM image to the sites of multiple large datasets that are distributed across the Internet. Moving compute to data, rather than vice versa, may be desirable for reasons beyond reducing the volume of network data transfer. For example, privacy regulations such as GDPR may restrict the movement of a dataset from its current location.

To illustrate the performance win from this approach, we benchmarked a simulation based on the *SeFO* weather prediction package[7]. The simulation reads 3.6 GB of weather data from each of two different network locations and then computes the visualization of a forecast for a specified month. The datasets were external to the VM image and were accessed via NFS. EdgeVDI made NFS access much more efficient by moving the VM instance close to the data. Table 3 summarizes our results. In the baseline case, the VM instance is fixed at one site and accesses the other site over a network with an RTT of 200 ms and a bandwidth of 20 Mb/s. In the EdgeVDI case, the VM instance moves to the second site to process its data. Total runtime with EdgeVDI is only one-fourth that of the baseline, in spite of EdgeVDI generating much more dirty state (roughly 12 GB) than the amount of distant data accessed in the baseline case (3.6 GB).

Table 3. SeFO execution on distributed data.

	Baseline	EdgeVDI
Read from first partition	68 (3)	67 (13)
VM Handoff		939 (4)
Read from second partition	3 979 (53)	114 (38)
Total time	4 047 (55)	1 120 (51)
Dirty memory state		8 304 (13)
Dirty disk state		3 746 (2)
Total data transmitted	3 600 (0)	1 835 (4)

All times in seconds, rounded to nearest second
All sizes in MB, rounded to nearest MB
VM image configuration: 4 vCPUs, 8 GB memory, 50 GB disk
Host configuration: 36 Intel Xeon 2.3 GHz CPUs, 64 GB memory, 2TB disk
Results are means of 3 runs, with standard deviations shown in parens

Legacy Database Joins Unless carefully optimized, relational database joins across datasets dispersed over a WAN can be inefficient. Modern database systems embody these optimizations, but legacy database systems may not. EdgeVDI enables applications built around these legacy database systems to be used efficiently on distributed data. Specifically, the VM instance performing the join can be moved to the optimal location for each step of the join. This can yield considerable reduction in network data transmission over a WAN and, hence, significant speedup.

To illustrate the speedup achievable, we run experiments using SQLite and the TPC-H database benchmark at scale factor 10 GB[16]. We run the following select-join query on two tables that are separated by a WAN with the same attributes as before (200 ms RTT, 20 Mb/s bandwidth):

```
SELECT avg(l_extendedprice),
     sum(l_quantity),
     count(*)
FROM lineitem as l, orders as o
WHERE l.l_orderkey = o.o_orderkey and
    o.o_orderdate <= date(''1992-01-25'')
```

In the baseline, the entire query is executed at the site of the larger table (lineitem, 7.6 GB). The data of the smaller table (orders, 1.6 GB) are fetched through NFS. In contrast, the EdgeVDI approach starts by running the SELECT part on the orders table locally, after which it triggers VM handoff. This moves the VM-encapsulated intermediate state to the site of the lineitem table, where it finishes the join.

As Table 4 describes, EdgeVDI reduces total query time from 860 to 483 s (43% reduction). It only ships 655 MB of data over the WAN, in spite

Table 4. Distributed joins in SQLite.

	Baseline	EdgeVDI
SELECT part of query	757 (19)	21 (9)
VM Handoff		371 (22)
JOIN part of query	102 (63)	91 (4)
Total time	860 (82)	483 (29)
Dirty memory state		3 346 (98)
Dirty disk state		486 (60)
Total data transmitted	1 600 (0)	655 (6)

Experimental setup identical to Table 3.

of dirtying over 3.8 GB. The baseline approach has to ship the entire smaller table, which is 1.6 GB in size.

Creating Dynamic Web Vantage Points

The ability to move encapsulated VM state efficiently over the Internet can be a valuable capability when studying regional differences in the properties of the World Wide Web.* By observing a website from a VM instance at one location, then moving the VM instance to a different location within a very short time and then repeating the observation there, an investigator can expose differences that are solely attributable to location. For example, because of differences in privacy laws in different parts of the world, the same website may use cookies differently depending on the location of the browser. Building on the work of Libert[8], Murgia and Harlow[9] report how different health websites share information from user queries with third parties. Although that work predates EdgeVDI, future studies of this genre can benefit from its precise control of software versions and configurations through VM encapsulation, and the temporal correlation made possible by fast VM Handoff.

RELATED WORK

The vision of replacing portable thick client hardware with a virtual desktop that is VM-encapsulated and delivered anywhere on the Internet was first described in 2002 in the Internet Suspend/Resume (ISR) project[6]. This vision was explored in series of implementations and deployments (ISR-1, ISR-2, ISR-3 and Open-ISR) between 2002 and 2011[13]. Their key difference from EdgeVDI is that they were all based on a *postcopy* VM movement strategy that demand-paged VM state over the Internet. In contrast, EdgeVDI uses a *precopy* strategy that ensures that all state is available at the target cloudlet before switching to it. This ensures rapid launch. After the switch to the target cloudlet, it also ensures uniform LAN-quality performance that is undisturbed by page faults. The tradeoff is that EdgeVDI suffers an initial period of suboptimal RDP interactions over a WAN.

The SoulPad work[1] explored the use of portable USB devices to transport VM-encapsulated virtual desktops. This approach is particularly valuable in contexts such as developing countries with poor Internet connectivity. The SoulPad and ISR approaches can be combined to achieve faster moves of a virtual desktop, as shown by Smaldone *et al.*[15].

As mentioned earlier, VM Handoff was inspired by live migration,[3] which is used in cloud computing. The ability to use the same underlying approach for many different purposes (system administration, load balancing, EdgeVDI) makes for a powerful abstraction with a common code base.

Delivering archival VM-encapsulated legacy software environments on demand over the Internet was the focus of the Olive project[12]. A key difference from EdgeVDI was that VM instances were discarded after use. While Olive aimed to perfectly recreate the same archival state on each use, EdgeVDI allows a user to continue from the point of last use.

The Collective[2,11] and Moka5[17] aimed to restructure software environments as collections of VM-encapsulated appliances. Neither wide-area mobility nor support for legacy software were explicit goals.

Using stateless thin-client devices for user interaction in personal computing dates back to devices like SunRay[18] and the X-Terminal[19] from the 1990s. Although these RDP-based hardware products have long been obsolete, their past existence points to possible future resurgence with EdgeVDI.

*We are indebted to Jonathan M. Smith of the University of Pennsylvania for suggesting this idea.

Edge computing has also been used to support legacy scientific instruments whose continued use remains important to the relevant scientific communities. Nguyen *et al.*[10] report on the motivation and approach to this important use case.

CLOSING THOUGHTS

Edge computing is driven today by the vision of exciting new edge-native applications, such as AR, real-time video analytics, data-driven drone control, and wearable cognitive assistance. In this article, we have shown how cloudlet infrastructure that is being deployed for such futuristic applications can also benefit bread-and-butter legacy applications. In particular, we have shown how unmodified VM-encapsulated legacy applications can benefit in a variety of ways from edge computing. As we have shown in the "Example Use Cases" section, even a WAN bandwidth value of 20 Mb/s is adequate to achieve usable performance. This is a conservative figure even today, and it will rise significantly over time. That will further improve usability and, hence, strengthen the argument for using Edge-VDI. It will also enable support for much large VM instances.

Extending VDI infrastructure to work in WAN environments is the critical capability needed to achieve the vision described in this article. To encourage experimentation, innovation, and commercial efforts in this space, we have released our implementation of EdgeVDI open source on GitHub[**]

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their feedback, which helped to improve this article. This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under Contract HR001117C0051, in part by the National Science Foundation (NSF) under Grant CNS-1518865, and in part by Intel, Vodafone, Deutsche Telekom, Crown Castle, InterDigital, Seagate, Microsoft, VMware, and the Conklin Kistler family fund. The work of R. Iyengar was supported by an NSF Graduate Research Fellowship under Grant DGE1252522 and Grant DGE1745016. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the view(s) of their employers or the abovementioned funding sources.

[**]https://github.com/cmusatyalab/nephele

REFERENCES

1. R. Caceres, C. Carter, C. Narayanaswami, and M. Raghunath, "Reincarnating PCs with portable soulpads," in *Proc. 3rd Int. Conf. Mobile Syst., Appl. Serv.*, Jun. 2005, pp. 65–78.
2. R. Chandra, N. Zeldovich, C. Sapuntzakis, and M. S. Lam, "The collective: A cache-based system management architecture," in *Proc. 2nd Symp. Netw. Syst. Des. Implementation*, May 2005, pp. 259–272.
3. C. Clark *et al.*, "Live migration of virtual machines," in *Proc. 2nd ACM/USENIX Symp. Netw. Syst. Des. Implementation*, 2005, pp. 273–286.
4. K. Ha *et al.*, "You can teach elephants to dance: Agile VM handoff for edge computing," in *Proc. 2nd ACM/IEEE Symp. Edge Comput.*, 2017, pp. 1–14.
5. IBM, "IBM Z Platform," Feb. 2020, Accessed: Feb. 3, 2020. [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/linuxonibm/liaag/wkvm/wkvm_c_overview.htm
6. M. Kozuch and M. Satyanarayanan, "Internet suspend/resume," in *Proc. 4th IEEE Workshop Mobile Comput. Syst. Appl.*, 2002, pp. 40–46.
7. N. Krakauer, "SeFo: A package for generating probabilistic forecasts from NMME predictive ensembles," *J. Open Res. Softw.*, 2016.
8. T. Libert, "An automated approach to auditing disclosure of third-party data collection in website privacy policies," in *Proc. Web Conf.*, Apr. 2018, pp. 207–216.
9. M. Murgia and M. Harlow, "How top health websites are sharing sensitive data with advertisers," *Financial Times*, London, U.K., Nov. 12, 2019.
10. P. Nguyen *et al.*, "Bracelet: Edge-cloud microservice infrastructure for aging scientific instruments," in *Proc. IEEE Int. Conf. Comput., Netw., Commun.*, Feb. 2019, pp. 692–696.
11. C. Sapuntzakis *et al.*, "Virtual appliances for deploying and maintaining software," in *Proc. 17th Large Installation Syst. Admin. Conf.*, Oct. 2003.
12. M. Satyanarayanan, "Saving software from oblivion," *IEEE Spectr.*, vol. 55, no. 10, pp. 36–41, Oct. 2018.

13. M. Satyanarayanan *et al.*, "Pervasive personal computing in an Internet suspend/resume system," *IEEE Internet Comput.*, vol. 11, no. 2, pp. 16–25, Mar./Apr. 2007.
14. M. Satyanarayanan, G. Klas, M. Silva, and S. Mangiante, "The seminal role of edge-native applications," in *Proc. IEEE Int. Conf. Edge Comput.*, Jul. 2019, pp. 33–40.
15. S. Smaldone, B. Gilbert, N. Bila, L. Iftode, E. de Lara, and M. Satyanarayanan, "Leveraging smart phones to reduce mobility footprints," in *Proc. MobiSys*, Krakow, Poland, Jun. 2009, pp. 109–122.
16. Transaction Processing Performance Council (TPC), "TPC-H is a decision support benchmark," 2020, Accessed: Feb. 3, 2020. [Online]. Available: http://www.tpc.org/tpch/
17. Wikipedia, "Moka 5," Jul. 2020. [Online]. Available: https://en.wikipedia.org/wiki/Moka5
18. Wikipedia, "Sun Ray," Aug. 2020. [Online]. Available: https://en.wikipedia.org/wiki/SunRay
19. Wikipedia, "X Terminal," Sept. 2020. [Online]. Available: https://en.wikipedia.org/wiki/Xterminal