Research Computing Desktops: Demystifying research computing for non-Linux users

Abhinav Thota[*]
Le Mai Weakley
Ben Fulton
H.E. Cicada Brokaw Dennis
Laura Huber
Scott Michael
Winona Snapp-Childs
{athota, llnguyen, befulton, hbrokaw, lamhuber, scamicha, wsnappch}@iu.edu
Pervasive Technology Institute
Indiana University
Bloomington, Indiana

Stephen Lien Harrell
Alexander Younts
Daniel T. Dietz
Christopher Phillips
Xiao Zhu
{sharrell, ayounts, ddietz, phill219, zhu472}@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana

ABSTRACT

Many members of the current generation of students and researchers are accustomed to intuitive computing devices and never had to learn how to use command-line based systems, which comprise the majority of high-performance computing environments in use. In the 2013-14 time frame, both Indiana University and Purdue university separately launched virtual desktop front-ends for their high performance computing clusters with the aim of offering an easier on-ramp to new users. In the last five years we iterated on and refined these approaches, and we now have over two thousand annual active users combined. Over 75% of those users say that the desktop services are either moderately or extremely important for their ability to use HPC resources. In this paper, we share our experience bootstrapping this new service framework, bringing in the end-users, dealing with runaway success, and making this service a sustainable offering. This paper offers a comprehensive picture of the driving motivations for desktops at each institution, reasons users like desktops, and ways of getting started.

CCS CONCEPTS

• **General and reference → General conference proceedings**; • **Social and professional topics → Management of computing and information systems**; **Systems development**; • **Applied computing →** *Life and medical sciences*; *Arts and humanities*; Physical sciences and engineering.

[*]Corresponding author

PEARC '19, July 28-August 1, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-7227-5/19/07…$15.00
https://doi.org/10.1145/3332186.3332206

KEYWORDS

supercomputing, high performance computing, remote desktops, VNC, GUI

ACM Reference Format:
Abhinav Thota, Le Mai Weakley, Ben Fulton, H.E. Cicada Brokaw Dennis, Laura Huber, Scott Michael, Winona Snapp-Childs, Stephen Lien Harrell, Alexander Younts, Daniel T. Dietz, Christopher Phillips, and Xiao Zhu. 2019. Research Computing Desktops: Demystifying research computing for non-Linux users . In *Practice and Experience in Advanced Research Computing (PEARC '19), July 28-August 1, 2019, Chicago, IL, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3332186.3332206

1 INTRODUCTION

Many members of the current generation of students and domain science researchers began computing exclusively using graphical interfaces and devices. Many of these individuals have rarely, or ever, had to use a terminal command line to do their work. However, in the age of increasing data volume, some researchers are finding themselves unable to do their analyses with desktop computers alone and are seeking more computing power, memory, and storage. When these researchers turn to high performance computing (HPC) systems, they typically encounter a barrier: HPC systems remain largely command-line based for a multitude of reasons. However, the research facilitation community is now in the early stages of adapting to user need. In 2013-2014, Indiana University and Purdue University began separate and simultaneous experiments in graphical computing to build services that would effectively bridge from researchers' desktops to HPC machines.

In 2014, Indiana University (IU) launched Karst Desktop (KD), a VNC based desktop frontend for Karst, its Linux-based cluster, based on the ThinLinc product by Cendio[1]. By the end of 2018, more than 1000 users had used KD, and more than 900 of those had used the service in that year, far exceeding the number of who ran jobs through the scheduler on Karst. The KD service has helped increase the total number of HPC users at IU by more than 50%.

During the same time period, Purdue University launched a dedicated graphical cluster based on the same ThinLinc product, which wraps VNC desktops for ease of use. This first installation had a capacity of 100 seats and was wildly popular from the beginning, often running out of concurrent licenses for new connections. In 2016, after a review of the ThinLinc cluster usage, it was evident that users were using the ThinLinc cluster as a jumping-off point to other clusters using SSH and X11 forwarding over SSH. Purdue moved to co-locate the software stack on top of each Community Cluster's set of frontend servers. By the end of 2017, all Community Clusters at Purdue—Carter, Conte, Rice, Halstead, Snyder, and Hammer—had ThinLinc deployed, and over a thousand of their users had taken the plunge into graphical HPC work.

Interestingly, as graphical services [16] have grown in popularity and been embraced by HPC users both new and old, IU and Purdue saw new avenues to help on-ramp users. Purdue began offering a dedicated cluster named Workbench at extremely cost effective rates to move users from their laptops into the data center. In 2018, IU launched the successor to the Karst Desktop called Research Desktop (RED). All of the above services have been incredibly well received and have experienced rapid, consistent growth. It is important to note that graphical desktops are widely used in the HPC community [20][15][18] and are well-understood from a technology perspective, however, there is a dearth of literature about supporting these desktops and their impact on users.

In this paper, we share our experience running our solutions from launch to retirement, and the lessons we learned. We hope this paper gives readers a road-map to launch similar services at their campuses. The rest of the paper is organized as follows. In section 2, we explain our motivation for exploring desktop frontends for HPC. In section 3, we describe a typical HPC remote desktop system architecture and the ThinLinc based systems that IU and Purdue use, in particular, including the architectural choices that were made, features that are supported, and how licensing works. In section 4, we discuss the support challenges that a desktop environment can create. In section 5, we share data showing the explosive growth of the services since their inception, reasons for it, and how we handled it. In section 6, we share the results of a series of surveys we have done with ThinLinc users, and other feedback we received from users on a range of topics. Finally, in 7, we describe some of the further steps we are planning and conclude the paper.

2 MOTIVATION: REDUCING THE BARRIERS TO HPC

As HPC application support and user-training staff at IU and Purdue, we frequently encounter researchers with use cases and needs that are beyond what their lab workstations can support. These researchers frequently face growing computational and storage needs; they are often new to HPC and come from disciplines that have not traditionally used HPC. At this point, when we determine that HPC and storage resources would address a researcher's needs, we introduce those researchers to the high performance computing and storage environments. We have observed that they typically reacted in one of the following ways.

If the researchers were Linux users, they enthusiastically learned about HPC and tried it out. They most likely complained about the scheduler, and about having to wait in the queue. They then made an estimation about whether learning HPC was worth all the trouble or if they could get their work done on their workstations one way or another. If the researchers were not Linux users, they most likely had all the same reservations as the Linux usres, and were also averse to learning the "command-line." If the researchers happened to have no choice but to move to HPC to make any progress, they committed to learning the command-line and gradually became users.

This told us that researchers often became HPC users because they had no other choice. Usually, down the road, after the researchers became more familiar with the command-line, and HPC in general, they came to appreciate their newfound additional compute power. These observations showed us that our new-user onboarding process did not communicate the value of HPC to researchers in a way that convinced them that learning command-line Linux and HPC was worth the effort. Being averse to learning Linux is understandable, and we are not disparaging the researchers. The command-line does not lend itself to intuitive learning, like a desktop or a smartphone.

For potential HPC users, the command-line and the batch job scheduling are major barriers. We were motivated to come up with a solution that created a more approachable, semi-intuitive, and somewhat forgiving environment for non-Linux users in which they could begin to use HPC resources. We were also interested in making this environment useful to existing users who struggled to use GUI-based applications on HPC machines using X forwarding and batch scheduling systems. After researching various implementations, we settled on a remote desktop based solution to address these challenges.

As we moved forward with implementation, we knew that we were in new territory and were careful to solicit frequent feedback from our users. Initially, we had face-to-face sessions to gather feedback, but last year, we conducted an online survey which solicited feedback on various aspects of the service. We share the results in more detail in section 6, but based on the results and based on the user growth we observed since launching these services, we are confident that the IU and Purdue implementations succeeded in reducing some of the major barriers to HPC adoption.

We were also aware that providing a desktop service would make HPC more accessible to potential non-traditional users on campus. We saw rapid user adoption of the desktop service by both new and existing users. At IU, within two years, the desktop service had as many users as the main campus cluster. This rapid growth expanded our user base beyond the more traditional research disciplines commonly associated with HPC. In section 5 we discuss how lowering the barrier to entry has expanded the user base, and how we work to match the users with the resources they need for their research. We also present additional data and trends that we observed during the last four years.

3 REMOTE DESKTOP: ARCHITECTURE AND IMPLEMENTATION

Remote desktops are a well-understood technology and have been widely available since the 1990's. There are a wide variety of implementations such as X Windows [30], Citrix [2], VNC [28] and others (see section 3.1). This paper primarily focuses on desktops in use within the HPC community.

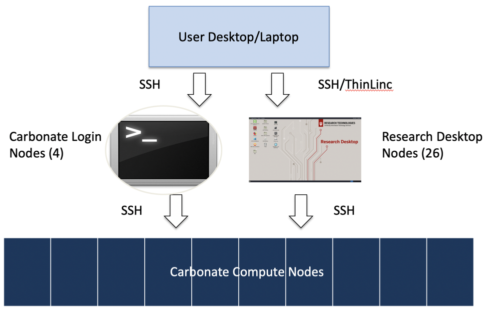

Figure 1: Architecture of the RED system

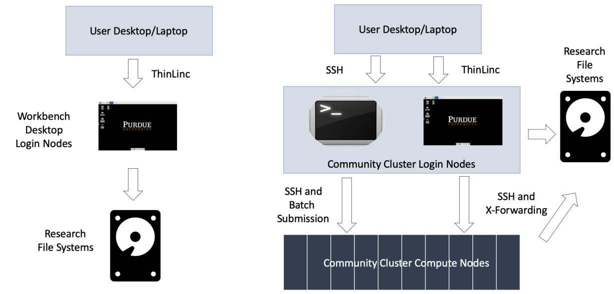

Figure 2: ThinLinc Architecture of Purdue's Data Workbench Cluster (left) and Community Clusters (right)

3.1 Remote Graphical Desktop Technologies Used in HPC

Remote graphical desktops exist in virtualized resource pools of compute and storage servers. The resource pool is shared across users, and resources are allocated on demand to user applications. In this paradigm, the expectation is reduced infrastructure, management, and maintenance costs through sharing of resources [32]. There are a number of technologies that can enable these desktops, as well as a number of different use cases.

Open solutions are very popular: (1) Open OnDemand [21] [26][29] enables a variety of web-based services for HPC clusters including remote desktops and allows the publishing of applications for finer control of a user's experience; (2) xrdp [10][12] provides remote X desktops over the Remote Desktop Protocol, a popular Microsoft Windows protocol; (3) Xvnc [11][25] gives users an X Desktop over the VNC protocol; (4) X2Go [8][17] is a graphical desktop server/client whose protocol runs over ssh, similar to ThinLinc , and allows the publishing of applications.

Non-open fee-based software packages are also widely used: (1) NoMachine [5][13] provides remote desktops over ssh and has some free components, however, large scale installations charge a fee; (2) FastX [6], another fee-based offering that can be used in the browser, also provides an API to interact with the desktops programmatically; (3) DCV [3] [22] enables remote desktops with the option to use NVidia GPUs for extra rendering power; (4) ThinLinc [33][19][14] is used extensively at IU and Purdue and will be the focus of sections 3.2 and 3.3.

ThinLinc is a product of the Swedish company Cendio AB. ThinLinc sessions communicate with the server using TigerVNC, an optimized implementation of the VNC protocol, and each session also has a dedicated X server. All client-server communication is done over a secure shell [34] tunnel, eliminating the need for additional layers of security such as VPN. The software also provides a front-end configuration that acts as a load balancer that can spread desktops among many nodes.

3.2 IU Research Desktop Architecture

RED provides graphical desktops at IU; it is also a computing cluster in its own right, consisting of 26 nodes and a separate gateway node. The gateway node maintains the list of sessions that are active on the remaining "agent" nodes. When users disconnect and reconnect to their existing session, the gateway knows where to send them. Each agent node has 256GB of memory and a mixture of two 8-core Intel Ivy Bridge processors and two 12-core Intel Haswell processors. The gateway holds a database of existing sessions and performs load-balancing by placing the sessions on the nodes with the least load or in a round-robin fashion as configured (Figure 1). Each node may handle a number of users, and there is a per-process memory limit set to 75GB to prevent a single user from overwhelming a node. The Mate desktop [4] is the default for users logging in for the first time.

3.3 Purdue Remote Desktop Architecture

Purdue makes use of the same software as the IU solution: ThinLinc. Our first incarnation of a ThinLinc solution was the aptly named "ThinLinc" cluster, which included eight servers from another endeavor that had ended. All HPC users at Purdue were granted login privileges to this cluster and were encouraged to explore the service's capabilities. The initial license purchased was for 100 concurrent sessions; this has grown every year since deployment.

After widespread usage, the cluster was heavily loaded and became critical infrastructure for all of Purdue's computing resources, making an evolution of the service necessary. The solution was to replace the "ThinLinc" cluster by dispersing the technology to each computing resource run by Purdue, including six Community Clusters. A Community Cluster runs the ThinLinc software spread across the existing frontend servers. CPU sharing to keep sessions interactive on the frontends is made more complicated by adding ThinLinc (Figure 2). The cgroups_py [24] CPU monitor, initially developed by IU and extended by Purdue, is used for maintaining interactivity.

One of our stand-alone research desktops is Workbench, which serves users who purchase only into Purdue's Data Depot service, as well as users who are not in a place to commit to investing in a Community Cluster. These users tend to be outgrowing their lab's capacity to host USB hard drives, and their willingness to do heavy computations on their laptops. An investment in the Community Cluster program begins at around $5,000 for five years of service, whereas an entire lab can subscribe to Workbench for only $300 per year.

With the retirement of the "ThinLinc " cluster complete and the new architecture deployed, Purdue has seen a dramatic increase in users of the ThinLinc software, and growth into new faculty groups.

3.4 Picking a VNC Implementation

Before Purdue's first HPC ThinLinc cluster, the Research Computing department was putting together scripts and how-to guides for users to set up and connect to VNC daemons over SSH. At the same time, the Engineering Computing Network, another IT group on campus, was searching for a Sun Ray thin client replacement. After seeing a demo of ThinLinc , Research Computing worked to set up a proof of concept. Having a native client available for Windows was a positive feature over port forwarding with PuTTY. Other features that seemed appealing were the web client, audio support, and potential for 3D acceleration support.

When IU did a survey of available VNC offerings in 2014, ThinLinc and NoMachine were the main established products available. Purdue was already offering ThinLinc-based desktops when IU started evaluating the options. Based on our evaluation, ease of setup, features, licensing costs, and the feedback from Purdue, IU went forward with ThinLinc.

Cendio ThinLinc is a commercial product that requires the purchasing of licenses (the pricing is available here[1]). Each active concurrent user session consumes a license; if a user disconnects but is still running the session, a license is consumed. If the user logs out, the license is released back into the pool. There is a configuration option to set the maximum lifetime of a disconnected session since last login. IU allows 7 days of idle disconnected time before a session is terminated.

4 SUPPORTING DESKTOP FRONTENDS FOR HPC

With remote desktop systems, users are encouraged to consider the service as a frontend to HPC systems. Deploying such services has required additional layers of support, given the graphical and interactive nature of the service as well as the wide variety of users and types of research that we wished to target.

Support took on different forms, from making file and web browsers available, to choosing and placing icons in a sensible way, to reaching out and creating various tools to aid researchers, especially ones with less experience in advanced computing. In line with the spirit of making a user-friendly service for those new to HPC, to complement the initial marketing of the service, we did a lot of in-person outreach providing workshops, demonstrations, and personalized training sessions.

In addition to supporting communities that used popular GUI-based applications, software like Windows applications [19] were supported as seen in Figure 4. At IU, Wine was installed at the request of a research group who wished to run a Windows application called SIMION, which simulates particle trajectories.

Faculty members also found the services to be an avenue for teaching computational research to students when these desktops were added to Purdue's existing education-only cluster: Scholar[14]. Many professors began using this resource for in-classroom interactivity to aid their instruction. The first semester ThinLinc was available, four classes used the educational cluster; in the Spring of 2019, 36 classes used Scholar with over 2500 students enrolled.

Supporting a shared resource pool also means ensuring that a user's experience is not affected by the gamut of activity in the service. Maintaining the reliability and usability of desktop services required a variety of approaches such as monitoring usage, limiting resource utilization, and creating desktop alerts.

4.1 Understanding Usage

Understanding usage is a key step to supporting a service. In order to get a picture of who users are and how they use the desktop service, IU put scripts monitoring usage and reliability in place along with e-mail alerts, and charted the data collected from the scripts. Usage metrics included, but were not limited to, number of users, number and duration of sessions, applications used, number of job submissions, and job submitters. In addition to shedding light on how the service was being used and who was using it, usage metrics and charts were, of course, helpful in planning and decision making, particularly in terms of illuminating peak license usage trends for purchasing additional ThinLinc licenses from Cendio in order to keep pace with growth.

IU also monitored load, network (in and out), and free memory, with Zabbix e-mail alerts set to be sent out for overload events. We also used Cgroups to limit resource usage among users and to create OOM desktop notifications within the user's session. We sent notices via e-mail to the user and to support team members. This monitoring helped us identify users who were over-using the CPUs and going out of memory and the processes they were running. Based on this information, we personally contacted users to inform them of their usage and see if we could provide solutions, perhaps by helping them move their work to the batch system or an interactive job.

Purdue offers two levels of interactive desktops as mentioned above. The Workbench service provides an on-ramp for users migrating from their laptops to centralized, powerful infrastructure. Our goal is to keep this service responsive; our major strategy for doing so involves keeping Workbench fed with enough hardware. When a user or research group grows too large, we upgrade them onto the Community Cluster that best suits their needs. The bulk of our usage tracking happens through Ganglia [23] and a watchful eye from the operations staff on the Community Cluster frontends. When a user who has upgraded their service misuses a cluster frontend without offloading work to a compute node, we email the user a review of our documentation and guidelines and invite them to one of our weekly in-person coffee hour consultations.

From a systems monitoring perspective, Purdue uses Sensu [9] for coarse service monitoring, Ganglia for overall system metrics, and Splunk [31] for additional dashboards. Additionally, we profile each user's job using Performance Co-Pilot [7] and XDMoD [27]. All of these tools allow us to catch systemic problems and understand them as they develop.

4.2 Challenges

The interactive nature of the service and the variety of experience among the users presented interesting challenges.

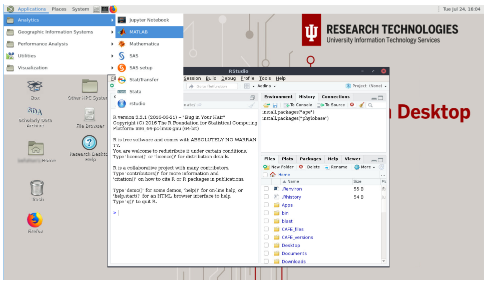

Figure 3: Research Desktop environment displaying the use of RStudio and various other applications available

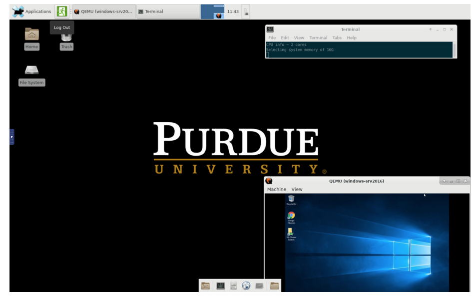

Figure 4: Purdue Remote Desktop environment displaying the use of Windows Server 2016 running on the frontend

The growth of usage turned out to be a challenge, as desktop services are shared resources. The data collection and analysis described above in section 4.1 helped us keep up with the increasing usage. Tracking daily license usage not only helped with planning for future purchasing, but also helped plan for workshops that used the desktop service where perhaps more licenses would be used. Cgroups and load data informed decisions about whether more nodes were going to be needed moving forward and how throttling should work.

The heterogeneity of the population using the services also presented different challenges that went beyond just introducing HPC novices to Linux or advanced computing. Issues like installing different languages for various GUI applications, and other issues related to the interactive nature of the service were hard to track down.

Other challenges related to the interactive nature of the service and diversity of research communities working on the desktops involved the variety of applications being run on the remote desktops. Among the top applications utilized on the services were applications like MATLAB, SIMION, RStudio, PhotoScan, ParaView, and COMSOL, which have a GUI or interactive component to them.

ThinLinc also gives the user the ability to disconnect from a session and reconnect to it later, as opposed to terminating all processes by logging out of the session completely. The convenience of being able to leave one machine and move to another without losing context is highly valued by users, especially when working interactively with applications; many have praised this option as a key feature (see section 6.3 for survey results). This, in turn, translates into a need to support long-running tasks and sessions.

The combination of having long running sessions with interactive and GUI applications in a shared resource pool can contribute to issues that are hard for the user to describe in addition to being hard to pinpoint and support. Fortunately, ThinLinc features the option to enable shadowing of a session, a useful tool that allows us to see issues that a user experiences within their sessions.

5 GROWING THE HPC COMMUNITY

As stated in section 2, one of the primary aims in offering a desktop service was to make HPC systems and services more approachable and accessible to a larger number of researchers and research disciplines. To do this, we identified a number of benefits that would motivate new users who did not have any experience with HPC or the command line. We found that, over time as our user base for the desktop systems grew, we needed to devise strategies to steer researchers to the correct resource for their workload. Helping new desktop users move on to the clusters' batch systems was important in order to prevent the desktop systems from being misused.

To support HPC novices at IU, we developed tools with the goal of helping users learn about the job submission process and perhaps help them "graduate" from using a session on the Desktop system to submitting jobs to the queues, and possibly other systems, should a need arise. For example, we created an icon on the desktop that, when opened, submits an interactive job for the user. Feedback to the user includes information about the kind of interactive job they have requested and what command could be run in a terminal to get an interactive job instead of clicking on the icon.

When the first HPC desktop service at IU was offered in 2015, it had 141 users, compared to 913 users in 2018. Among them, 300+ users are active on average each month. At IU, desktop services are used by roughly 65% of the all HPC users, and among all the HPC systems at IU, RED has the most users. Purdue has seen similar growth over the years and also averages 300+ users every month.

5.1 Benefits

Providing such a resource pool to high performance computing users makes a variety of possibilities available. Applications with graphical interfaces can be offered; links to server-based applications such as Galaxy can be provided as links on the desktop; and terminals are available to run traditional command-line based software, which is especially useful when that software has a X-windows component. Additionally, some of the features that new users have found to be beneficial to their work flows include:

- File browsers and file editors
- Graphical access to the compute nodes (indirectly)
- Convenient data transfer options
- Support for long-running tasks
- The ability to disconnect and reconnect to the current session

Placing file editors like Emacs and popular GUI-based applications like RStudio, Matlab, and QGIS in the menu bars makes such applications visible and easy for users to find. Combined with the file browser feature, the desktop gives users the confidence to look for things on their own as they point and click through a seemingly familiar interface. Some of these features and the desktops are shown in figures 3 and 4.

At IU, the current deployment of the desktop service is called Research Desktop (RED) and is an evolution of the original Karst Desktop service. Purdue employs a fleet of research desktops, one connected to each community cluster as well as a few one-off research desktops for specific communities of users. In Research Computing at Purdue, the move toward maintaining a desktop service was born out of a need for reliable X-Windows (GUI) applications on the community clusters. The response was overwhelming, and within a year a cluster of ThinLinc based frontends was created. Since then, the desktops have supported a myriad of use-cases within research computing including classroom usage, on-demand windows virtual machines, and bridging between laptop and cluster computation. In both instances, lowering the barrier to entry, and adding functionality enabled by desktops, helped previously-reluctant users to make great use of HPC resources.

It is difficult to get users to adopt new technologies. In order to do so, the new technologies must (1) offer advantages over current practices, (2) positively change the way that work can be performed, and (3) be easy to implement and straightforward to use [35]. Both the Karst Desktop and Purdue ThinLinc offerings address each of these issues: the convenience of a familiar graphical interface makes the transition simple and straightforward; simple file transfers and persistent sessions are big advantages over a command line system; and larger amounts of available memory provides an advantage over a laptop machine. The simplicity and familiarity of graphical interfaces are a positive change for each user.

5.2 Balancing the Mix of Traditional HPC and Desktop Use

One of the challenges of an expanding user base for a desktop service is preventing poor performance and a bad end-user experience due to resource contention. Although both Purdue and IU have put safeguards in place via cgroups to prevent users from monopolizing system resources, there are still instances in which a user will attempt to use more resources than a desktop node can support given the number of other users on that node. A common example of this is a user running parallel Matlab processes. In these cases, Purdue and IU have taken slightly different approaches to mitigating desktop "power users." IU has developed a set of monitoring scripts to detect desktop nodes that are under high load and have users who are using a large amount of computational resources. As mentioned in section 4.1, users who are in this state are personally contacted and, in the case of parallel applications, advised on running their workflow in the batch system. Purdue's approach involves identifying "power users" and placing them in a cgroup partition with other "power users." In this way, the more general-purpose users are not impacted by the "power users," and the "power users" will be motivated to use batch submission due to poor performance.

While IU supports users running serial workloads directly on the RED desktop nodes, when they have parallel workloads, IU has had reasonable success in transitioning users to submitting batch jobs to the compute nodes. None of these users, to our knowledge, have become exclusive batch job users, but at least 20% of the total RED users have submitted at least one job to the batch system in 2018. This is a big jump compared to the handful of users who submitted batch jobs when the service was first introduced in 2014.

In a similar vein Purdue has had success migrating users to batch who are outgrowing the Remote Desktops. However, with cgroups containing any high CPU usage, Purdue's users decide for themselves when they feel that they need something more. This difference from IU support is directly correlated to the difference in offerings and research computing resource pricing, i.e. Purdue's remote desktop service does not have a batch system and Purdue's Remote Desktop computing service is 7x to 10x cheaper than batch computing.

6 USER SURVEYS

In order to evaluate these services, two user surveys were conducted. In December 2017, IU performed a survey of remote desktop users to determine their levels of satisfaction with the service. In January 2019, IU and Purdue conducted a similar joint survey. The surveys were designed to reveal what users of the systems found to be important features of the desktop service, and how the desktop service was impacting the research of various research communities. We report on the 2019 survey, as it is the first survey to include results from multiple institutions.

6.1 Participants

Participants in the 2019 survey were users of the IU or Purdue Research Computing Desktop services in 2018, a total of almost 1700 individuals. 324 individuals, 180 with IU institutional affiliations and 144 Purdue institutional affiliations, agreed to participate and did so with informed consent. Thus, there was a total 19% response rate.

6.2 Survey Instrument and Administration

Users were asked to evaluate the service at their institution, and for feedback on how the service could be improved. This survey was approved by the Human Subjects Office at Indiana University. The survey instrument, including types of responses, can be viewed here: https://www.doi.org/10.5967/re7v-p284.

After potential participants were identified, they were sent an invitation to participate over email. A follow-up e-mail was sent one week after the initial invitation. A final e-mail was sent out two weeks after the initial request. This survey was unusual in that participants were able to share their identities and all but one did so.

6.3 Results

User feedback from the survey was informative and positive, with 81% of respondents reporting that they were satisfied or extremely satisfied with the services, and 75% of respondents reporting that the remote desktop services are extremely or moderately important to their ability to use HPC resources.

Several items asked respondents to indicate the importance of various features of the desktop interface. Several of these features were common to both the IU and Purdue surveys. The respondents were given a five point Likert scale ranging from "not at all important" to "extremely important." The features were "point and click interface," "web browser," "file browser," "GUI/graphical text

Table 1: Summary statistics for features of importance.

Item	Mean	St. Dev.	N
Long Session	4.53	0.80	311
Jobs	4.14	1.06	309
Point-and-Click	4.09	1.11	307
File Browser	3.91	1.18	306
GUI	3.85	1.15	306
Thindrives	3.70	1.19	307
Web Browser	3.33	1.41	306

editor," "ability to run long-running sessions," "ability to mount local laptop file system using Thindrives," and "ability to run interactive and batch jobs." The mean values for the combined IU and Purdue responses are presented in table 1. Pairwise Wilcoxon tests showed that both the ""web browser" item and the "ability to run long-running sessions" items were statistically different from all other items with a 95% confidence interval. The remaining items had at least one other item for which there was no statistically significant difference. The ability to run long uninterrupted sessions was rated most important by both university populations, while the ability to use a web browser had the lowest importance rating.

When inter-comparing results between the populations at the two universities, a Mann-Whitney test with a 95% confidence interval was used to test for differences between the two populations for each of the 5-point Likert items. Of the items rating the importance of features, only the "point and click interface," "file browser," and "ability to mount local laptop file system using Thindrives" showed differences with significance. Table 2 shows the summary statistics for each university for these three features. In all cases, the mean value of the importance is ranked less for respondents from Purdue University than for respondents from Indiana University. We hypothesize that this could stem from two differences in the systems. First, the IU system is primarily advertised as a separate, stand-alone system rather than as a front-end to a cluster like several of Purdue's systems. Due to this difference in presentation and architecture, users may be more inclined to execute their entire work flows (including data management and manipulation) in the IU system, as opposed to using only the desktop system for the components of the work flow for which a GUI is essential.

Table 2: Summary statistics by university for features showing difference.

Item	Indiana University			Purdue University		
	Mean	St. Dev.	N	Mean	St. Dev.	N
Point & Click	4.20	1.10	172	3.95	1.10	135
File Browser	4.03	1.16	171	3.76	1.19	135
Thindrives	3.87	1.13	170	3.50	1.24	137

Secondly, variation in the composition of the respondent population may play a role in the difference cited above. When looking at the primary research disciplines of the respondents, the Purdue respondents were largely concentrated into a single discipline with 43% of respondents coming from the Engineering disciplines and the second most populous discipline, Biological Sciences, having only 8% of the respondents. On the other hand, the IU respondents were more evenly distributed, with the top three disciplines Biological Sciences, Computer and Information Science, and Psychological and Brain Sciences having 16%, 12%, and 9% of the respondents, respectively. With a large concentration of respondents in the Engineering discipline, it could be that respondents from Purdue generally use engineering work flows that do not use these interactive data management features. Work flows from the Biological Sciences, Computer and Information Science, and Psychological and Brain Sciences likely rely more heavily on interactive data management features.

In addition to the Likert response items asking about the importance of these features, there were three items designed to assess the importance of the desktop service to users' research, their ability to use HPC resources, and their overall satisfaction with the desktop services. Table 3 reports the mean values, standard deviations, and total number of respondents for these items.

Table 3: Summary statistics for overall importance and satisfaction.

Item	Mean	St. Dev.	N
Importance to research	4.08	1.05	315
Importance to HPC	4.06	1.07	314
Satisfaction	4.10	0.78	313

Overall, satisfaction is quite high and respondents rate the desktop service as being moderately or extremely important to their research and their ability to effectively use HPC resources. Of these three items, the only one that shows a significant difference between the IU and Purdue respondents is the overall importance to research with the IU mean value being 4.21 and the Purdue mean value being 3.93.

7 FURTHER STEPS AND CONCLUSION

The variety of research communities that benefit from advanced computing is growing. For many of those who are new to research computing, the command line seems disruptive to the workflows to which they are accustomed. In order to welcome users from varying backgrounds, institutes have turned to offering remote or interactive desktops to users as a means to streamline users' moves to HPC.

As data sizes continue to grow, the HPC community will need to address the needs and various computing backgrounds of the booming population of researchers who can increase and improve their scientific output through advanced computing. Remote and interactive desktops have a role to play in supporting this growth.

The goal of detailing our steps and our combined experiences in this paper is to inform other institutions, and to serve as a call-to-action to advanced computing centers that may thinking of offering an interactive desktop. Many institutions support similar services without actually thinking of this as a new service area. Knowledge-sharing of how to deploy and support these services can advance and enhance how research computing facilitators improve their offerings. With this paper and with further community building efforts, we hope to establish a community of practice around the desktop computing front ends for HPC.

ACKNOWLEDGMENTS

The authors would like to thank Robert Henschel, David Hancock, Joseph Rinkovsky and Research Technologies at Indiana University for supporting these services and this work. We would also like to thank Dr. Mark Daniel Ward and the Purdue College of Science for providing funding to expand the Purdue Scholar service and Preston Smith, Betsy Hillery and the Research Engineering team for their tireless efforts on behalf of the Purdue Remote Desktop services.

REFERENCES

[1] [n. d.]. Cendio ThinLinc. https://www.cendio.com/thinlinc/what-is-thinlinc. Accessed: 2019-02-20.

[2] [n. d.]. Citrix Virtual Apps and Desktops - Virtual Apps and Desktops - Citrix. https://www.citrix.com/products/citrix-virtual-apps-and-desktops/ Accessed: 2019-04-08.

[3] [n. d.]. DCV - Remote OpenGL - 3D in the Cloud - NICE. https://www.nice-software.com/products/dcv Accessed: 2019-03-27.

[4] [n. d.]. MATE Desktop Environment | MATE. https://mate-desktop.org/ Accessed: 2019-04-04.

[5] [n. d.]. NoMachine - Free Remote Desktop For Everybody. https://www.nomachine.com/ Accessed: 2019-03-27.

[6] [n. d.]. PC X Remote Desktop Software - FastX, Leading PC X Server Solution - StarNet. https://www.starnet.com/fastx/ Accessed: 2019-03-27.

[7] [n. d.]. Performance Co-Pilot. https://pcp.io/ Accessed: 2019-04-04.

[8] [n. d.]. start [X2Go - everywhere@home]. https://www.x2go.org/ Accessed: 2019-03-27.

[9] [n. d.]. Workflow automation for monitoring | Sensu. https://sensu.io/ Accessed: 2019-04-04.

[10] [n. d.]. xrdp by neutrinolabs. http://www.xrdp.org/ Accessed: 2019-03-27.

[11] [n. d.]. Xvnc - the X-based VNC server. https://www.hep.phy.cam.ac.uk/vnc_docs/xvnc.html Accessed: 2019-03-27.

[12] E. I. Alwi, T. B. Adji, and S. Sumaryono. 2016. Implementation and performance analysis of MPI cluster system in distributed rendering. In *2016 International Conference on Computational Intelligence and Cybernetics*. 103–108. https://doi.org/10.1109/CyberneticsCom.2016.7892575

[13] F Ambrosino, A Colavincenzo, G Guarnieri, A Funel, G Ponti, A Mariano, F Palombi, G Bracco, and S Migliori. 2015. NEPTUNIUS: the ENEA HPC portal for multiphysics simulations. ISC.

[14] M. E. Baldwin, X. Zhu, P. M. Smith, S. L. Harrell, R. Skeel, and A. Maji. 2016. Scholar: A Campus HPC Resource to Enable Computational Literacy. In *2016 Workshop on Education for High-Performance Computing (EduHPC)*. 25–31. https://doi.org/10.1109/EduHPC.2016.009

[15] Patrice Calegari, Marc Levrier, and PawełBalczyński. 2019. Web Portals for High-performance Computing: A Survey. *ACM Trans. Web* 13, 1, Article 5 (Feb. 2019), 36 pages. https://doi.org/10.1145/3197385

[16] Paul Calleja, Stuart Rankin, Filippo Spiga, and Cambridge HPCS. 2014. An overview of Remote 3D visualisation VDI technologies.

[17] A G Demenev. 2018. Perm State University HPC-hardware and software services: capabilities for aircraft engine aeroacoustics problems solving. *IOP Conference Series: Materials Science and Engineering* 307 (feb 2018), 012053. https://doi.org/10.1088/1757-899x/307/1/012053

[18] Roi Costas Fiel. [n. d.]. *Graphics processing on HPC virtual applications*. Ph.D. Dissertation. Chalmers University of Technology. http://publications.lib.chalmers.se/records/fulltext/245767/245767.pdf

[19] Richard H. Grant, Stuart D. Smith, Stephen Lien Harrell, Alex Younts, and Preston Smith. 2018. Windows-based Workflows on Linux-based Beowulf Clusters. In *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*. ACM, New York, NY, USA, Article 32, 5 pages. https://doi.org/10.1145/3219104.3219118

[20] Victor Hazlewood, Dan Lapine, J. Ray Scott, Derek Simmel, David Walling, Peter Enstrom, Tabitha Samuel, Jason Charcalla, Maytal Dahan, and John Towns. 2017. XSEDE Technology Investigation Service (TIS). In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact (PEARC17)*. ACM, New York, NY, USA, Article 17, 10 pages. https://doi.org/10.1145/3093338.3093357

[21] David E. Hudak, Douglas Johnson, Jeremy Nicklas, Eric Franz, Brian McMichael, and Basil Gohar. 2016. Open OnDemand: Transforming Computational Science Through Omnidisciplinary Software Cyberinfrastructure. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale (XSEDE16)*. ACM, New York, NY, USA, Article 43, 7 pages. https://doi.org/10.1145/2949550.2949644

[22] Scott Lathrop, Celso Mendes, Jeremy Enos, Brett Bode, Gregory Bauer, Roberto Sisneros, and William Kramer. [n. d.]. Best practices for management and operation of large HPC installations. *Concurrency and Computation: Practice and Experience* 0, 0 ([n. d.]), e2653. https://doi.org/10.1002/cpe.5069 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.5069 e2653 CPE-18-0839.R2.

[23] Matthew L Massie, Brent N Chun, and David E Culler. 2004. The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Comput.* 30, 7 (2004), 817 – 840. https://doi.org/10.1016/j.parco.2004.04.001

[24] Curtis Maves and Jason St. John. 2018. cgroups py : Using Linux Control Groups and Systemd to Manage CPU Time and Memory. In *In HPCSYSPROS18: HPC System Professionals Workshop*. Dallas, TX. https://github.com/HPCSYSPROS/Workshop18/tree/master/cgroups_py_Using_Linux_Control_Groups_and_Systemd_to_Manage_CPU_Time_and_Memory

[25] I. MUNRO, E. GARCÃŊA, M. YAN, S. GULDBRAND, S. KUMAR, K. KWAKWA, C. DUNSBY, M.A.A. NEIL, and P.M.W. FRENCH. 2019. Accelerating single molecule localization microscopy through parallel processing on a high-performance computing cluster. *Journal of Microscopy* 273, 2 (2019), 148–160. https://doi.org/10.1111/jmi.12772 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/jmi.12772

[26] Jeremy W. Nicklas, Doug Johnson, Shameema Oottikkal, Eric Franz, Brian McMichael, Alan Chalker, and David E. Hudak. 2018. Supporting Distributed, Interactive Jupyter and RStudio in a Scheduled HPC Environment with Spark Using Open OnDemand. In *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*. ACM, New York, NY, USA, Article 26, 8 pages. https://doi.org/10.1145/3219104.3219149

[27] J. T. Palmer, S. M. Gallo, T. R. Furlani, M. D. Jones, R. L. DeLeon, J. P. White, N. Simakov, A. K. Patra, J. Sperhac, T. Yearke, R. Rathsam, M. Innus, C. D. Cornelius, J. C. Browne, W. L. Barth, and R. T. Evans. 2015. Open XDMoD: A Tool for the Comprehensive Management of High-Performance Computing Resources. *Computing in Science Engineering* 17, 4 (July 2015), 52–62. https://doi.org/10.1109/MCSE.2015.68

[28] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. 1998. Virtual network computing. *IEEE Internet Computing* 2, 1 (Jan 1998), 33–38. https://doi.org/10.1109/4236.656066

[29] Zebula Sampedro, Thomas Hauser, and Saurabh Sood. 2017. Sandstone HPC: A Domain-General Gateway for New HPC Users. In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact (PEARC17)*. ACM, New York, NY, USA, Article 33, 7 pages. https://doi.org/10.1145/3093338.3093360

[30] R Scheifler. 1987. *RFC 1013 X Window System Protocol*. Technical Report.

[31] Jon Stearley, Sophia Corwell, and Ken Lord. 2010. Bridging the Gaps: Joining Information Sources with Splunk. In *Proceedings of the 2010 Workshop on Managing Systems via Log Analysis and Machine Learning Techniques (SLAML'10)*. USENIX Association, Berkeley, CA, USA, 8–8. http://dl.acm.org/citation.cfm?id=1928991.1929003

[32] Vanish Talwar, Klara Nahrstedt, and Dejan Milojicic. 2009. Modeling remote desktop systems in utility environments with application to QoS management. In *Integrated Network Management, 2009. IM'09. IFIP/IEEE International Symposium on*. IEEE, 746–760.

[33] Abhinav Thota. 2016. Introducing graphical access to IU's supercomputers with Karst Desktop Beta. Indiana University Workshop in Methods. https://scholarworks.iu.edu/dspace/handle/2022/20627

[34] Tatu Ylonen and Chris Lonvick. 2005. *RFC 4251 The secure shell (SSH) protocol architecture*. Technical Report.

[35] Ann Zimmerman and Thomas A Finholt. 2007. Growing an infrastructure: The role of gateway organizations in cultivating new communities of users. In *Proceedings of the 2007 international ACM conference on Supporting group work*. ACM, 239–248.