Cloud Computing using Virtual Network

Kumar Kishan Chandra,
Assistant Professor
R.D.&D.J College,Munger

Dr. Anand Kumar Pandey
Assistant Professor
T.N.B. College,Bhagalpur

Supriya Raj
Research Scholar
UDCA,Bhagalpur

Abstract

Thispaper is an overview of virtual network embedding strategies within cloud infrastructure backbone networks. We will first of all summarise and define the main principles of cloud computing and its different supplied services. Then, we will describe the cloud infrastructure architecture, named the Totally Virtualized Cloud Infrastructure (TVCI), which makes use of virtualization in all the cloud's equipment (i.e., backbone and data centers). Afterwards, we will describe, analyse, and compare the main virtual network mapping algorithms for cloud infrastructure networks found in existing literature. We will evaluate the above strategies in terms of: i) virtual network request reject rate, ii) embedding cost of virtual network request, iii) embedding revenue of virtual network request and iv) average usage rate of physical resources.

Chapter 1
Introduction

A virtual network is a software-defined network where all links and components may or may not have direct interaction with physical hardware. In most cases, direct interaction with physical hardware is made by the hypervisor or the host controller. All links between virtual machines, virtual switches, virtual bridges, and virtual network interfaces are made completely virtually.

1.1 Research scope and goals

Mobile devices, ranging from smartphones to tablets, have recently become so pervasive that they are increasingly replacing personal computers in ev-eryday activities related to both entertainment and work. However, due to their limited resources, mobile devices cannot o er the same performance of personal computers and workstations. To this regard, one of the prominent approaches to overcome such limitations consists in o oading computational and storage resources to the cloud [4, 5]. With o oading, the mobile device runs only a thin layer of software which interfaces with application-speci c services in the cloud. For instance, a software for picture organization and categorization can exploit powerful and accurate face recognition algorithms running on the cloud without the need of any computation at the mobile device. However, such an approach requires that source data are available to the remote service. This may require transferring data from the mobile device to the cloud, which incurs in both communication and energy consumption overheads

A di erent option is given by remote desktop access [34]. In this case, the mobile device uses a thin client software which connects to a remote desktop server providing an operating system and its applications. The thin client shows the desktop user interface and handles the related interactions. Speci cally, input events captured by the client are transferred to the server and the display of the mobile device is updated according to the received response, in order to match the content of the

desktop screen. To a certain extent, remote desktop access can be seen as an extreme case of mobile cloud computing, wherein the mobile device only acts as remote display and input device, while all the rest is demanded to the remote system [18]. When the remote server is virtualized, this access scheme corresponds to a special case of mobile cloud networking [29].

While research targeted to mobile cloud computing has considered re-source utilization as the primary design objective, most of the commonly used solutions for remote desktop access were originally designed for per-sonal computers. As a consequence, the reference scenario was represented by systems which have enough resources, are static and access the Internet through a wired connection. Even though there are some solutions speci - cally designed for mobile devices [2, 36], they are usually not publicly avail-able, or they cannot be easily integrated in the existing infrastructure. As a consequence, the vast majority of remote desktop protocols available for mobile devices are still those designed for personal computers.
In this thesis, we aim at characterizing the energy consumption of mobile cloud computing realized through remote desktop technologies. Our goal is to analyze the performance of widely used remote desktop protocols through experiments involving di erent classes of mobile devices and realistic usage scenarios. Consequently, we have to ensure the reliability and consistency of all experiments. Finally, we also seek to relate the energy consumption to the di erent components involved and to the protocol features.

1.2 Contributions

In this thesis, we characterize the energy consumption of mobile cloud com-puting realized through remote desktop technologies. Among them, we fo-cus on Remote Desktop Protocol (RDP) and Virtual Networking Computing (VNC) as they are the most important solutions. Speci cally, we examine the impact of the considered remote desktop protocols on power consumption. We take an experimental approach by measuring the power consumption of both a smartphone and a tablet under di erent use cases corresponding to realistic usage scenarios, namely, remote desktop publishing and remote assistance.

In order to ensure reliability and consistency for all the experiments, we have developed a methodology to automate experiments with an Android device in order to produce repeatable experiments with comparable results. Through the application that we have developed, we managed to achieve reliability and consistency in our experimental results. We have also veri ed that our software

The Virtual Cloud Network builds upon the fundamentals you're already familiar with from NSX—these include (but are not limited to) integrated security, consistent connectivity, and inherit automation, but really focuses on tying together an end-to-end architecture that allows our customers to deliver applications and services everywhere. Our customers have asked and we have listened… the demand for any infrastructure, any cloud, any transport, any device, and any application has drastically changed the landscape and technologies associated with building/architecting and having a modern enterprise network. Cin their functionality to a point where it is now possible to have cellular phones execute Java programs. As a result, cellular users throughout the world are now able to read and write e-mail, browse Web pages, and play Java games using their cellular phones. This trend has prompted us to propose the use of a cellular phone as a device for remotely controlling computers. For example, if a cellular user is able to remotely access computers (such as workstations in offices.) Mobile cloud computing can give mobile device users a number of advantages. Company users are able to share resources and applications without a high level of capital expenditure on hardware and software resources. Mobile cloud computing provides a solution to meet the increasing functionality demands of end-users, as all application logic is executed on distant servers and only user interface functionalities reside on the mobile device. The mobile device acts as a remote display, capturing user input and rendering the display updates received from the distant server. The most sophisticated form of remote access enables users on one computer to see and interact with the actual desktop user

interface of another computer. Setting up remote desktop support involves configuring software on both the host (local computer controlling the connection) and target (remote computer being accessed). When connected, this software opens a window on the host system containing a view of the target's desktop.Current versions of Microsoft Windows include Remote Desktop Connection software. However, this software package only supports target computers running Professional, Enterprise or Ultimate versions of the O/S, making it unsuitable for use with many home networks. For Mac OS X computers, the Apple Remote Desktop software package is also designed for business networks and sold separately. For Linux, various remote desktop software exists. Many remote desktop solutions are based on Virtual Network Computing (VNC) technology. Software packages based on VNC work across multiple operating systems. The speed of VNC and any other remote desktop software can vary, sometimes performing effectively the same as the local computer but other times exhibiting sluggish responsiveness due to network latency.

Chapter 2

Background

In this chapter, we will introduce the concepts of remote desktop and virtual networking and then give an overview of mobile cloud computing. Finally, we will discuss possible approaches for energy pro ling.

2.1 Remote Desktop Protocols

In this section, we introduce the concepts of remote desktop access and vir-tual networking. Remote desktop is a thin client architecture where a remote desktop client sends keyboard and mouse input events to a server and receives screen updates as responses. This concept is illustrated by Figure 2.1. First, the user starts the remote desktop client software and connects to the ter-minal server. The interactions between the client and the server are closely related to the I/O operations. Initially, the client receives the content of the whole screen from the server. Then the client starts sending input events to the server. For instance, moving of the mouse pointer causes the client send the corresponding coordinates to the server. If the input events generate a change in the content of the screen, the region of the screen is calculated by the server and sent back to the client. The client then updates its own screen with the received update to match the content of the server. This process goes on until the client disconnects. Generally, the concept of re-mote desktop access can be realized through di erent protocols. The most important ones are Remote Desktop Protocol (RDP) and Virtual Network-ing Computing (VNC). The main di erence between them mainly lies on the features speci c to the protocols. Such features will be detailed in the next subsections.

2.1.1 Virtual Network Computing

VNC [32] is a well established multi-platform remote desktop solution. It is built on top of the Remote Framebu er Protocol (RFB), which evolved over time to include extensions and optimizations introduced by software derived from the original VNC server [31]. RFB, as the name suggests, operates at the framebu er level and is based on primitives which can draw rectangular regions on the screen. Each region can be encoded in di erent ways and a sequence of drawing primitives produces a framebu er update. Updates are triggered by the client which can autonomously decide when to request them.

The message exchange between a client and a server using the RFB proto-col is shown in Figure 2.2. It consists of three main phases: Handshake, Ini-tialization and Normal Protocol Interaction. The Handshake phase involves the exchange of a sequence of messages that determines the RFB protocol version and the type of security to be used. In the context of VNC, VNC authentication will be used. The server sends a random 16-byte challenge and the client will encrypt the challenge with DES, using

the user supplied password as a key. This type of authentication, however, is known to be cryptographically weak and is not recommended for use on untrusted net-works [31]. The Initialization phase involves the exchange of initialization message between the client and the server. The ClientInit message sent from the client speci es if the server should share the desktop with other clients or grant exclusive access to this client. Upon receiving the ClientInit message, the server replies the client with a ServerInit message that consists of the width and height of the server's framebu er, its pixel format and the name associated with the desktop. The Final phase involves normal protocol in-teraction between the client and the server. The client can send any request messages to the server, and may receive response messages or framebu er updates from the server.The RFB protocol also supports a number of encoding types (Raw, Copy-Rect, RRE, TRLE, Hextile and ZRLE) but, in practice, the current imple-mentations only use ZRLE, TRLE and CopyRect encodings as they o er the best compression for typical desktop In addition to the screen, the RFB protocol supports any input device which can be mapped to a keyboard and a pointing device (e.g., a mouse) with multiple buttons.

2.1.2 Remote Desktop Protocol

RDP is a proprietary protocol designed by Microsoft for remote input and display of hosts running the Windows operating system, and it is based on the Multipoint Application Sharing (T.128) recommendation by the In-ternational Telecommunication Union [35]. Despite being OS-speci c, RDP clients and servers also exist for operating systems other than Microsoft Win-dows. RDP supports more primitives than VNC for screen updates, as well as data transmission over multiple channels.

RDP message interactions between a client and a server are shown in Figure 2.3 [19] and consist of ten distinct phases. The Connection Initia-tion phase involves the agreement on a Class 0 X.224 Connection Protocol Data Unit (PDU) between the client and the server. After this phase, all subsequent data sent between the client and the server are wrapped in an X.224 Data PDUs. The Basic Settings Exchange phase involves message exchange between the client and server using Multipoint Communication Services (MCS) Connect Initial and Response PDU. These packets contain concatenated blocks of settings data (e.g., core display data, security data and network data) which are eventually read by the client and the server. In the Channel Connection phase, the client attaches primary user identity to MCS and the server replies with the User Channel ID. The Client then joins the user channel, the input/output (I/O) channel, and all of the static virtual channels. Likewise, the server con rms the client's joining.

Chapter 3

Remote Desktop Access for Mobile Cloud Networking

In the following, we will discuss the most relevant approaches aimed at min-imizing the energy consumption of mobile nodes by exploiting the cloud and by designing optimized remote access protocols. For clarity, we group the related work in the two sections below.

3.1 Cloud-based Solutions

O oading resources to the cloud has been one of the major aspects of re-search e orts targeted to improving the performance and reducing the energy consumption of mobile devices [16]. Some approaches have focused on tech-niques for automated and even dynamic o oading of mobile applications to the cloud. Among them, MAUI [5] is a platform which can evaluate the energy-cost tradeo of applications running on top of the Microsoft .NET framework. Based on such evaluation, MAUI automatically decides to exe-cute part of the application in the cloud rather than on the mobile device. ThinkAir uses a similar approach but targets Android devices and pro-vides a ner grained granularity for o oading decisions. Di erent from those solutions, CloneCloud enables o oading Android applications to the cloud without the need of special support by the application developer,

who would otherwise need to annotate the application code to exploit the o oading platform. Due to the focus on computational and storage resources, the so-lutions for o oading mentioned above address computing aspects more than networking ones. However, it has been shown that real applications do not bene t from o oading very much, since they are not so computationally-bound to be worth running (even partly) in the cloud .Here, we take a more practical approach targeted to mobile cloud networking, where users access feature-rich remote desktop applications. Di erent from ,which presents an experimental evaluation of energy consumption in mobile cloud computing, we consider realistic use cases and speci cally address remote desktop scenarios. Some research on o oading proposed running native mobile applications in the cloud and accessing them through a virtual network connection. This concept of virtual smartphone has been introduced in ,which also pre-sented an architecture suitable to support the remote installation and ex-ecution of mobile applications (Figure 3.1). The approach in relies on the Virtual Network Connection (VNC) protocol for remote access, but it does not provide any characterization of the related resource utilization. In this thesis, instead, we explicitly address the energy .The concept of virtualized screen was introduced in [18], as well as an architecture for thin-client mobile devices to perform screen rendering in the cloud. The proposed model targets rich graphic environments in which rendering is very computationally expensive, and takes a cross-layer approach that is both network and content-aware. Even though the architecture is described in detail, the system is not evaluated in real-world scenarios. In contrast, in this thesis we experimentally evaluate the energy consumption of di erent remote desktop solutions available o -the-shelf.

3.2 Mobile-friendly Remote Access Solutions

Solutions for remote desktop access have been traditionally targeted to per-sonal computers [32]. Approaches suitable for mobile devices have been de-signed only recently. For instance, MobiDesk was proposed in as an infrastructure for mobile virtual desktop computing. Mobi Desk uses different techniques to provide display and network virtualization suitable to both local and wide area network environments, including session migration. However, this solution mainly targets mobile laptops and not devices such as smartphones or tablets which have signi cantly lower energy resources. In contrast, we speci cally consider energy-constrained mobile devices in this thesis.

SmartVNC is a solution built on top of VNC in order to improve the user experience in remote computing scenarios. The proposed solution operates directly at the user interface level and records the sequence of opera-tions performed by the user as application-independent smart-macros. These macros can then trigger the playback of procedural operations with a single tap or gesture, thus eliminating cumbersome or redundant operations at the user interface of the mobile device. The goal of SmartVNC is to increase the productivity and e ciency of an existing remote desktop protocol, with focus on the user experience. Di erent from that solution, we speci cally address the energy consumption of remote desktop access at the mobile device by considering several existing protocols, including VNC

Remote display solutions for mobile cloud computing were surveyed in The authors recognized battery lifetime as one of the major challenges for cloud access from mobile devices and highlighted the importance of o oad-ing and cross-layer design as key aspects of energy-e cient cloud network-ing.

Remote access is the ability to access a computer or a network remotely through a network connection. Remote access enables users to access the systems they need when they are not physically able to connect directly; in other words, users access systems remotely by using a telecommunications or internet connection. People at branch offices, telecommuters and people who are traveling may need access to their companies' networks.

Remote access enables remote users to access files and other system resources on any devices or servers that are connected to the network at any time, increasing employee productivity and enabling them to better collaborate with colleagues around the world.

A remote access strategy also gives organizations the flexibility to hire the best talent regardless of location, remove silos and promote collaboration between teams, offices and locations.

Technical support professionals also use remote access to connect to users' computers from remote locations to help them resolve issues with their systems or software.

One common method of providing remote access is via a remote access VPN connection. A VPN creates a safe and encrypted connection over a less secure network, such as the internet. VPN technology was developed as a way to enable remote users and branch offices to securely log into corporate applications and other resources.

The principle of mobile cloud computing physically separates the user interface from the application logic. Virtual network computing (VNC) is a desktop sharing system which uses the RFB (Remote Frame Buffer) protocol to remotely control another computer. It transmits the user events from one computer to another relaying the screen updates back in the other direction, over a network using Buffered IO Stream.

In a mobile cloud computing environment, the remote display protocol deliver complex multimedia graphics over wireless links and render these graphics on a resource constrained mobile device. Offloading applications to the cloud is a straight forward way to save on energy consumption because the amount of local processing is reduced. Efficient compression techniques to reduce the amount of exchanged data are done using compression techniques and versatile graphics encoding, downstream data peak reduction and Optimization of upstream packetization overhead

IV. PROBLEM FORMULATION

Conventional desktop applications need to be redesigned to operate on mobile hardware platforms, thereby often losing functionality. More demanding applications typically require specific hardware resources that are very unlikely to be available on mobile devices.

The web hosts increasingly powerful computing resources and has evolved to a ever-present computer, offering applications ranging from simple word processors, over all-encompassing enterprise resource planning suites to 3D games, to make availability of such application is not easy.

Chapter 4
Experimental Plan

In this chapter, we discuss the methodology used to conduct the experimental evaluation. In order to automate the experiments in such a way that they are not much a ected by the timing of the user inputs, we developed an application that can record the touch events and replay them for a certain number of times. In addition to the implementation of application, we also discuss the experimental setup.

4.1 Automation of Experiments

In order to conduct the experiments in such a way that they are not much af-fected by the timing of the user inputs, an application-independent approach to perform a certain sequence of operations is required.Currently, there are two existing approaches: recording and replaying of live trace of input events on mobile devices; and the monkeyrunner tool from the Android SDK [10] to control the mobile device from the workstation b sending speci c commands or input events from an API.

The proposed solution in records the sequence of user touch events and translates the recorded events before replaying them on the mobile de-vice. This approach requires the captured trace to be further processed and translated o ine on a workstation instead of running directly on the mo-bile device. Another solution mentioned in [14], used the Android SDK's monkeyrunner tool to automate user interaction with applications in the experiments that investigated the impact of storage on application perfor-mance. Monkeyrunner requires a number of small programs customized for di erent benchmarking use cases. Both of these approaches also require ex-ternal connection to the host workstation through an Android Debug Bridge (adb) connection [9]. adb is a tool used to communicate with the Android mobile device via either USB (in the USB debugging mode) or a TCP/IP connection with the target device. This is undesired for energy measurement as the USB cable connection to the workstation serves as a form of charging mechanism that will cause inaccuracies in energy measurements. Likewise, TCP/IP connection to the workstation also introduces additional tra c and unwanted bias in the power measurements.Android follows the standard Linux input system, which involves events from the keyboard, touchpad and so on. These events are available in the /dev/input/eventX where X is a number starting from 0. Di erent mobile de-vice models have di erent event number representation. Each of these events is a 128-byte structure input event in Android that contains a timestamp, an event type and a payload. To read the events supported by the mobile devices, we exploited the getevent tool from the Android SD

The Virtual Cloud Network Summary

Traditional enterprise networks were built by connecting physical devices (PCs, servers, switches, routers, etc.) to one another. Once the network was established, typically speaking security solutions were added into the mix at the end. The emergence of software-driven networks has challenged these "traditional" norms. Today, with the advancement of software-defined networking, businesses that want to modernize their network with software open up a new set of possibilities to drive business innovation. Virtual networks run more efficiently and lower operational costs by automating network infrastructure functions such as maintenance, patching, and updating. They also increase uptime, improve service, and reduce costs—enabling organizations to maximize the value of their existing network hardware while creating new innovation and business opportunities.

The majority of enterprises we speak with are already using, or plan to use, multiple clouds to run their business. Today, everything of value is connecting to the network and new customer experiences are delivered through the cloud. Users connect not only through their desktops and laptops, but there has been a tremendous uptick in mobile endpoints. Applications run everywhere across a company's infrastructure; however, many companies don't have a clear understanding about the security and connectivity of their applications and devices across the data center, branch, cloud, and edge. These emerging technology trends increase the complexity of enterprise security and challenge the limitations of hardware-driven networks. Networking in software creates a business fabric that securely and consistently connects a company's data centers, carrier networks, branches, endpoints, and clouds… and it's all done independently of the underlying network hardware.