Journal Of Harmonized Research (JOHR)

Journal Of Harmonized Research in Engineering
2(1), 2014, 75-79

ISSN 2347 – 7393

Original Research Article

# TRIGGERING FASTER AUTOMATED DOWNLOADS ON A WLAN USING MULTITHREADING AND THE JXTA PROTOCOL SUITE

**B.A.Pimpare, H. F. Surve, S.H. Lonkar, S.R. Kharage**

BE Comp, Modern Education Society's College of Engineering, Pune-411001

**Abstract-** Most organizations, educational institutes and homes have computers that require having the same downloadable software, files or even applications. If it is mandatory to have separate copies of the same on each of these computers, it is naturally time consuming to download these manually. All these devices, in an office or a lab, are usually on the same network. We can leverage these existing networks to share download links amongst devices. This can be done with minimal manual interaction. Only one user from any one of these devices would be able to trigger such a parallel download. In this paper we suggest a faster alternative to the conventional direct download and the general practice of exposing shares on a network. The proposed tool, having JXTA at its core is independent of the platform and can be installed on various desktop and laptop devices running different operating systems.

***Key terms*- Wireless Networking, WLAN, P2P, JXTA, multithreading**

## Introduction

A WLAN or wireless LAN is a group of networked devices situated in relatively close physical proximity to each other.

These devices communicate using radio or infrared signals and require no network cabling. WLANs can be found in many homes, schools, and businesses. The devices in a WLAN today, range from desktop and laptop computers to mobile phones to multimedia devices such as TVs and CD/DVD players.

Wireless networking has brought about a fundamental change to the way in which people communicate share and access data on the network and the Internet. University campus networks, computer laboratories, office intranets and home networks; all conventionally, have some tools that are expected to be installed and services, which are to be made available on all the devices of the network. Developers can capitalize on these existing networks to provide the solutions needed.

**For Correspondence:**
1bhagyashri2pimpareATgmail.com
Received on: February 2014
Accepted after revision: February 2014

**Peer-To-Peer Networks**

Peer-to-Peer (P2P) networks are a special case of WLANs. A P2P network is created when two or more PCs are connected and share resources without going through a

Separate server computer. A P2P network varies widely in scale. It can be an ad hoc connection—a couple of computers connected via a Universal Serial Bus or a conventional Windows ‗Home group'to share and transfer files. A P2P network also can be a permanent setup that links somewhere under 10 computers in a small office over wired lines. It can also be much bigger so as to encapsulate special protocols and applications to set up direct relationships among users over the Internet.

P2P systems are most often associated with P2P file sharing clients that have been in use extensively over the last decade. Their popularity has led to their use in information and file sharing systems [4]. Napster, the now defunct internet service for file sharing was once the most popular P2P application which allowed users to share MP3 files over the Internet. Gnutella also garnered millions of Internet peers and a massive 40% of the Internet traffic [6]. These applications refer to file sharing over the Internet amongst Internet peers. This idea can and has been extended to smaller home and office networks. Most home computer networks today are peer to peer networks. Residential users configure their computers in peer groups to allow sharing of files, printers and other resources equally among all of the devices. Although one computer may act as a file server at any given time or be the one to which the printer is permanently connected, other home computers are also capable of providing the same services and fulfilling the same responsibilities. Thus in effect, every connected PC can at once be a server and a client. In a P2P environment, access rights are governed by setting sharing permissions on individual machines.

**Existing Downloading Mechanisms**

The ubiquitous task of downloading files and software from Internet servers has been subject to many improvements since its inception. The method of downloading, the tools for downloading, the download rates and the types of files being downloaded have all undergone a sea change. High download rates ensure multiple users on the network can download simultaneously. Large files can however still take an excruciatingly long amount of time to get downloaded entirely. We now have desktop applications that manage downloads and also some that are inbuilt into browsers that have the same or a slightly stripped-down feature set. Some download managers accelerate downloads by downloading form multiple servers at once. Download acceleration – referred to as multipart download, on the other hand deals with downloading a single file by splitting it into segments. Each segment is associated with a different connection to the same server so as to reduce the bandwidth required per connection and subsequently allow the server to serve the client better. Internet Download Manager is best known for using accelerated downloads as it is for its error recovery prowess and its capability to resume downloads. Another download mechanism not extensively used is remote triggering of downloads. One service to accomplish this allows users to add files to a Dropbox folder from work/home which are later synched to the same folder on a MAC device at home/work. All these tools are employed, for users and network administrators to be able to keep track of as well as achieve higher download speeds.

As part of our project, we aim to trigger direct downloads to all devices on the network from one device and in cases where the server supports byte range operations, we plan to employ a multipart download but to different devices. The proposed sharing technique is discussed in the next section. The goal is to limit manual interaction to one person and to better utilize network bandwidth.

**File Sharing Tools And Mechanisms**

For files and freeware that can be shared on a network, general file sharing techniques

involve either exposing a share on one device on the network or downloading to the network attached storage for users on the network to access at their will.

Existing tools include WinSCP (Windows Secure CoPy)- a Windows-specific open source FTP, SFTP and SCP client. Its basic function is to ensure transferring files securely between two computers. For secure transfers, it uses Secure Shell (SSH) and supports the SCP protocol in addition to SFTP.

Another popular tool - Samba is the software implementation of the SMB/CIFS protocol. It allows Windows and Linux machines on a network to talk to one another [7]. It allows directories on a Linux system on the network to be shared with multiple other Windows and Linux systems much like NFS (Network File System). Samba uses Microsoft‘s SMB protocol (Server Message Block) that enables sharing of files, printers, serial ports as well as communications abstractions such as named pipes and mailboxes between computers. SMB essentially is a client-server request-response protocol. Samba thus, is a SMB server for UNIX machines enabling it to participate in a Windows Workgroup or Domain. Microsoft on the other hand has a different set of SMB server implementations for Windows machines. There are numerous cross platform FTP clients available that connect one device or multiple peers to a server to securely download and share data.

Since we are downloading the file in parts on different devices we need to share the parts across the network. This mechanism is inspired by the way Bit-Torrent clients work. Here the peers are on the same network and hence we use local peer discovery to locate the peers to which we send the downloaded section. Every peer performs this multicast to other peers on the network.

**Jxta As A Means For Developing P2p Applications**

Project JXTA was introduced for peers running on different physical machines to be able to communicate [4]. These machines could be running different operating systems, have different security settings and run in different tools environments. It has made P2P independent of programming language, system platform as well as networking platform.

JXTA allows peers to form and manage peer groups. A peer can communicate to the rest of the peer group, the services it has to offer as well as the pipes it shall use to send and receive data using advertisements. An advertisement is an XML structured document being broadcasted to all the network peers or it could be pipe information being sent to a particular peer in the peer group. XML further warrants language and platform independency for the P2P application.

We learnt from [1] and [2] of the JXTA protocol suite that had six major protocols when it first came out: Peer Discovery Protocol deals with letting a peer discover other peers, peer groups of interest and advertisements on other peers. Peer status and peer information can be obtained using the Peer Information Protocol. The Pipe Binding Protocol addresses how pipe advertisements bind to pipe endpoints. This can be achieved at runtime. JXTA allows both multi-peer and peer-to-peer pipes for one way point to point communication. The Peer Resolver Protocol enables a peer to send and receive queries to search for other peers, peer groups and pipes. To be a member of a group a peer needs to apply for membership using the Peer Membership Protocol. It also includes methods to receive, update and cancel membership credentials. Two peers that wish to communicate may not be able to reach each other directly or may be separated by a firewall or NAT. In such a situation any peer on the network can route a message between these two peers by being configured as a peer router. This is a feature provided by the Endpoint Routing protocol.

In the proposed tool we plan to use Peer Information Protocol to first acquire system and platform information from every network peer on the peer that triggers the download. The user on this peer can then appropriately

send platform specific download links to the peers before downloads can be triggered. For locating peers on the local network the master device can employ local LAN based Discovery which involves sending a broadcast message over LAN or use discovery via Rendezvous points which serves better in more complicated network configurations where some peers might not be immediately discoverable [1]. Peers can be categorized into groups based on their properties and gain membership to them temporarily using the Peer Membership Protocol.

Once the parallel downloads are triggered, sharing on the peer group can commence simultaneously where the parts can be shared over the local network and subsequently put together by every peer.

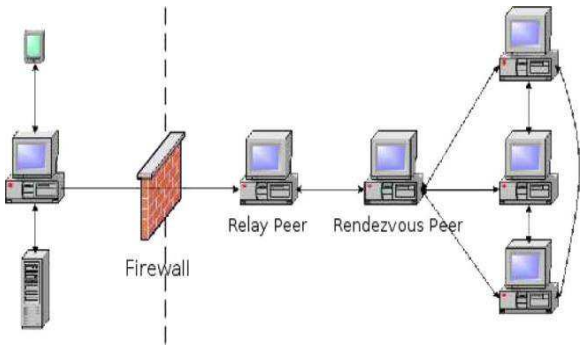


Fig 1. Communication on a JXTA network amongst differently configured peers

**Network Configuration**

Our network peers in the WLAN should be part of a JXTA peer group. Each peer should be configured in the infrastructure mode. Every peer will be made capable of functioning as a master or a slave. The master is the device over which the user will acquire the platform specific file URLs. A slave is one which responds to the master with system information and over which the download thread is triggered. Every peer (master/slave) will know about all other peers in the peer group at each time, for the master to be able to monitor download progress and for the slaves to function as masters at any point in time. Once the download is completed the file can be shared on the network with the peers configured as ad hoc peers. This ensures better utilization of the local network bandwidth.

We expect the proposed tool to support a WLAN the size of a small Windows ‗Homegroup' of 3-6 desktop or laptop computers or a medium sized office network with a dozen computers to a computer lab with up to 20-25 computers running Windows or any Linux distribution.

**Multithreading to Download Parts of a File**

Java's built-in support for multithreading allows a developer to run multiple threads simultaneously. Multithreading enables writing of software that increases CPU utilization. This feature along with JXTA's Java implementation – JXSE, helps us develop interactive network applications.

As part of the project we plan to run different threads on different peers/slaves. These threads would be triggered from the master device on to the slaves. Each thread downloads a part of the file to be downloaded to each of the devices of the network or peer-group. These threads can run in parallel and will put no load on the master/triggering device. The network bandwidth would be divided equally amongst the threads downloading the parts. This method of file download is similar to multipart downloading in principle but here, different parts go to different devices rather than a single one. This not only reduces the load on the server on which the file is being split (as in multipart downloads) but also has to handle sockets with different peers which makes it easier to manage the individual connections.

**Security Considerations in Jxta Networks**

Ref. [3] introduces us to the security aspects of JXTA networks. JXTA uses TLS version 1 to support private and reliable connections between peers. This is with reference to Internet peers in a peer group or across peer groups. The main reason why TLS was adopted was cryptographic security – the protocol should be able to establish a trusted connection between two parties. TLS allows applications running on JXTA based networks to be able to communicate and exchange cryptographic parameters without

knowing the implementation details. JXTA also permits peers to form peer groups and designate one member as the group‘s certificate authority by making its root certificate part of the peer group‘s binary version of JXTA. In an internet peer group, a group member can transfer such a certificate to another reliable prospective member over a TLS connection. We can secure our system by ensuring only an authorized network administrator gets access to the system. Also when receiving parts from a peer every peer can ensure whether that peer belongs to the same peer group as it does.

**Conclusion and Future Works**

Multipart downloading is being slightly tweaked so as to download parts to multiple peers of the network. Communication and sharing amongst related peers on the network is achieved by configuring them as JXTA peers and using protocols that help in local discovery and information sharing. The goal is to attain better download speeds and make managing software and data that is required across the network an easy task. Future enhancements could include using this mechanism in installers for software which is to be made available to the entire network. Also system could be improved on the security front by making every peer check the credibility of the download link it receives from the master device as well as the parts it receives from other peers.

**Acknowledgments**

The authors would like to thank their sponsors namely Persistent Systems and the Computer Department at MES College of Engineering for their support.

**References**

1. Li Gong, ―JXTA: A network Programming Environment‖, IEEE Internet Computing, May- June 2001, pp.88-95
2. Leonard Barolli and Fatos Xhafa, ―JXTA-Overlay: A P2P Platform for Distributed, Collaborative, and Ubiquitous Computing‖, IEEE Transactions on Industrial Electronics, vol. 58, no.6, June 2011,pp.2163-2172
3. William Yeager and Joseph Williams, ―Secure Peer-to-Peer Networking: The JXTA Example‖, IEEE IT Pro, March-April 2002,pp.53-57
4. Hao Shi, Yanchun Zhang, Jingyuan Zhang, Beal, E, ―Collaborative Peer-to-Peer Service for Information Sharing Using JXTA‖, Computer and Computational Sciences, 2006. IMSCCS‘ 06, June 2006,pp.552 – 559
5. Ou Kaiqian, Xu Yinlong, Ma Guanjun, Zhu Yulin, ―Dasher: A peer-to-peer content distribution system based on combined network coding‖, Broadband Network & Multimedia Technology, 2009. IC-BNMT '09, 18-20th Oct 2009, pp.687-692
6. Eric Bangeman, Ars Technica Study: ―Bit Torrent sees big growth, LimeWire still #1 P2P app‖, April 21, 2008
7. A Samba Overview - http:// www.linuxvoodoo.com/resources/howtos/ amba.