IsRemote

< iskandar@arc.itb.ac.id | iskandarsetiadi@students.itb.ac.id >

@author : Iskandar Setiadi (freedomofkeima)

An easy to use Remote Desktop Applications

Recommended for ones who want to learn the basic of remote desktop

Distributed System Laboratory Assistant Selection Test

Sister - 妹 lab

Institute Technology of Bandung

Jl. Ganesha No. 10, Bandung 40132

Year 2013

Document Change Records

Version	Date	Author	Description
0.1	30-08-13	IS	Initial

Contents

I. Introduction

Remote desktop is an application which is used to access a computer without physical access through some kind of network. Remote desktop can be divided into two main parts: client side and server side (host). There are a lot of functionalities such as sending mouse and keyboard inputs, uploads, downloads, and file transfers from the local computer (client) and send them to the remote computer (server). Beforehand, both computers (client and server) need to be connected to the same network. TeamViewer, LogMeIn, RemotePC are several notable tools which work based on remote desktop principle.

Remote desktop has a lot of functionalities such as technical troubleshooting of users' problems and educational presentations. Remote desktop also has several drawbacks regarding security issue and malicious use. Within remote desktop protocol, one could access other people data which may violate privacy issues. Malicious use such as backdoor installation will give the attacker full control of victim's system.

If you don't have any special purpose in using remote desktop, at least, you could use it for one simple reason called prank. The idea is simple, you only need to install a remote desktop application in your friend computer without his/her consent, wait for midnight, take control of his/her system and start typing randomly. Rest of the idea depends on how wicked you are. Anyway, this idea is just one of the basic functionality of remote desktop application.

II. Main Application (IsRemote)

Without UI control support, IsRemote is divided into two main parts, IsRemoteServer and IsRemoteClient. The simplified version supports connection from client to server system. IsRemote is built using Java as its backbone.

II.1 Server Side (IsRemoteServer)

To execute IsRemoteServer, you may use *"java –jar remoteServer.jar"* to start this application.

IsRemoteServer consists of **5 classes**, which can be divided into this categorization:

Main	Main.java	Main class, activating protocol
Protocol & Service	RemoteController.java	Handling server system robot access (using *java.awt.robot*)
	RemotePreferences.java	Default address to run, default port, default screen size, default sleep time, and several application constants
	TCPServer.java	Used for maintaining connection with client
Helper	Viewer.java	Helper for screen capture

This application starts with its TCPServer activation, which is waiting for clients' connections. After a client is connected, TCPServer will send screen capture of its system (**Viewer.java**) and wait for each mouse & keyboard events. **RemoteController** will take access of each received events and execute it by using Java Robot. Basically, these transactions will occur in certain interval (defined in **RemotePreferences** as *sleepTime*).

II.2 Client Side (IsRemoteClient)

To execute IsRemoteClient, you may use *"java –jar remoteClient.jar"* to start this application. Beforehand, you need to ensure that IsRemoteServer is already running properly on target system.

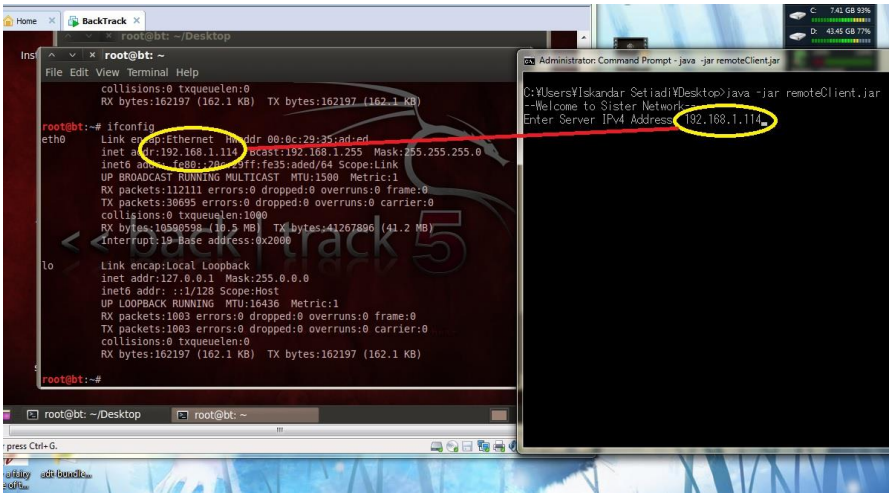
IsRemoteClient consists of **4 classes**, which can be divided into this categorization:

Main	Main.java	Main class, binding to server with certain IP address and resolution size
Protocol & Service	RemotePreferences.java	Default address to run, default port, default screen size, default sleep time, and several application constants
	TCPClient.java	Used for maintaining connection with server
Interface	UserInterface.java	Handling UI for server remotely system, listener for mouse and keyboard event

After running IsRemoteClient, user will be prompted with questions regarding server IP address and server's resolution from **Main** class. After this information is correctly set, TCPClient will try to bind TCPServer from IsRemoteServer. TCPClient will receive server's screen capture, transmit it to **UserInterface**, and send all information regarding mouse & keyboard events to server. Basically, these transactions will also occur in certain interval (defined in **RemotePreferences** as *sleepTime*).

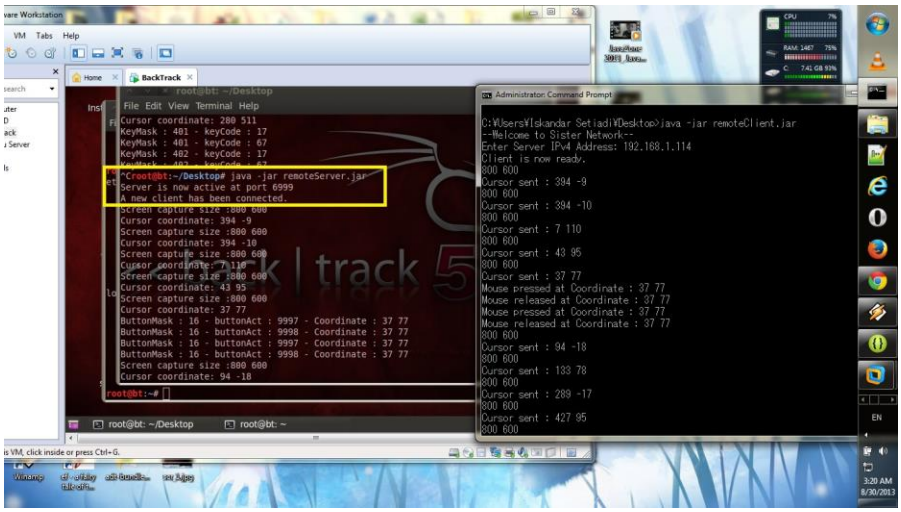
II.3 Server-Client Connection

The following example shows remote connection from Windows 7 to Backtrack 5 operating system, by using VMWare Workstation 9.0.1. Using *ifconfig* in UNIX-based or *ipconfig* in Windows-based, one should be able to obtain IPv4 address of its system. See Picture 2.3-1 below for easier understanding.



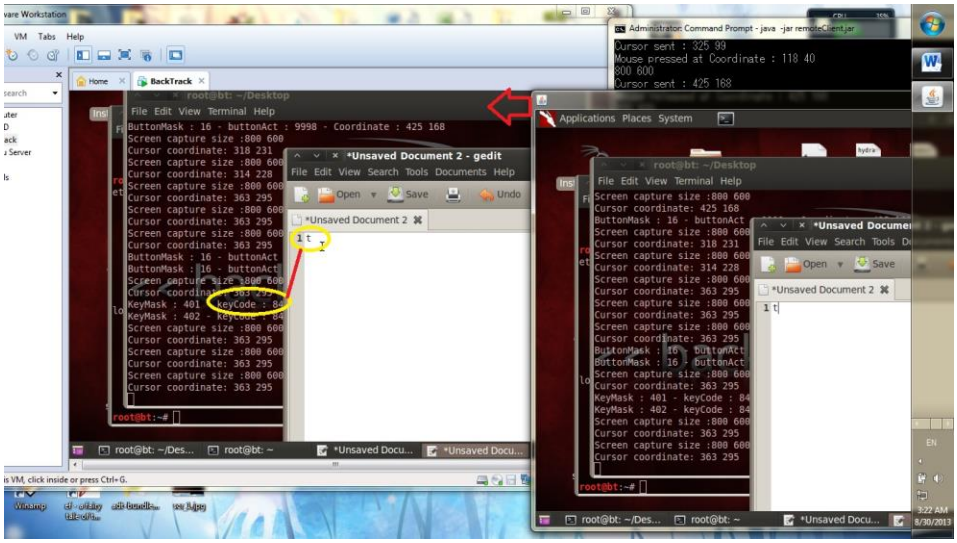
Picture 2.3-1 Obtaining Ipv4 Address

Currently, all connections will occur within port 6999 as default port. After successfully establishing connection with server, IsRemoteServer will give brief message "A new client has been connected".



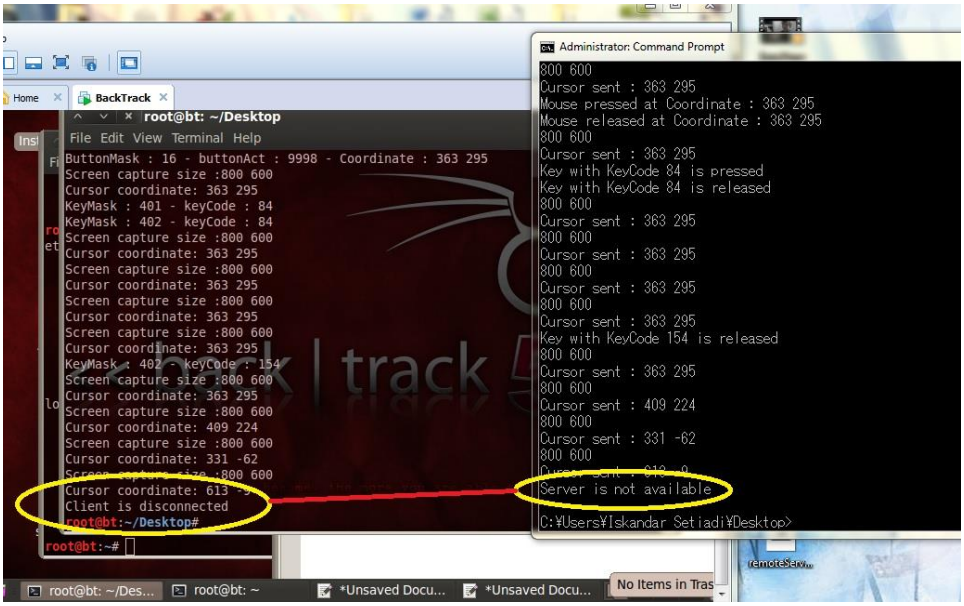
Picture 2.3-2 Starting IsRemoteServer and IsRemoteClient

Server will send its screen capture in PNG format and receive mouse (cursor, wheel, click) & keyboard informations from client. For example, picture 2.3-3 below shows keyboard event after a character "*t*" is typed from client application.



Picture 2.3-3 Interraction between Server and Client

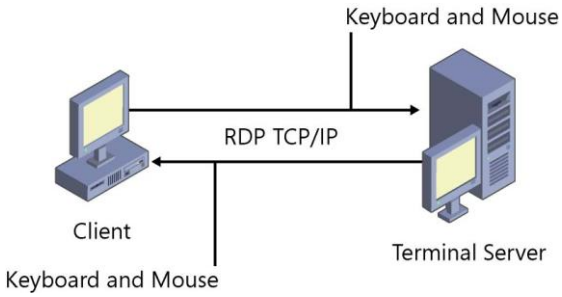
After client closes its UI window or server stops sending its screen capture, both IsRemoteServer and IsRemoteClient will be halted. Picture 2.3-4 shows message for disconnecting application.



Picture 2.3-4 Disconnecting IsRemoteServer and IsRemoteClient

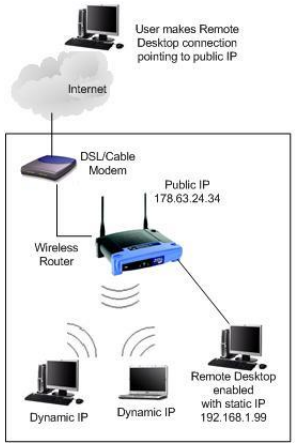
III. How Remote Desktop Works

Nowadays, there are a lot of protocols which are practically used in remote desktop applications. Several notable protocols are RDP (Remote Desktop Protocol), ICA (Independent Computing Architecture), ARD (Apple Remote Desktop), and VNC (Virtual Network Computing). As each protocol varies, there's also a lot of custom-built protocol which is called as proprietary protocol. Each protocol may support different built-in encryption such as SSH, SSL, TLS, RSA, and AES encryption. These encryptions functionality is somewhat optional, but a secured connection is a compulsory priority for users nowadays.



Picture 3-1 Client and Server in RDP

Picture 3-1 represents the simple example of remote desktop protocol. Basically, you only need two computers with input devices such as keyboard and mouse. Several remote desktop applications also support reverse remote connections. For connection over internet, see illustration from picture 3-2 below.



Picture 3-2 Remote Desktop over Internet

IV. FAQ & Troubleshooting

This section is intended to be used for frequently asked question (FAQ) and troubleshooting.

Q : Is this application free?

A : Of course. This is an open source project. You can even modify it as long as my courtesy is still written there ☺

Q : Could I contribute in this project?

A: Yeah! You can access this project from git repository
(http://github.com/freedomofkeima [still private project atm])

Q : How can I report error / bug / troubleshoot?

A : You may contact me via e-mail which are stated above (in cover page).

Q : Am I cute?

A : Yes, you are!

"I CAN SEE THE ENDING!" – KATSURAGI KEIMA