Testing performance of tools for online anonymity

Heidi Howard

June 25, 2012

Contents

1 Aims and Objectives

1.1 Types of Anonymity

The two types of anonymity that I will be considering are:

- hide identify for web hosts including your locations, IP address, browsing habits, MAC address, facebook username etc.. - hide browsing from man-in-the-middle dispute use of unsecure Wi-Fi e.g. in cafes

1.2 Difference between Anonymity and Security

1.3 Evaluation Criteria for Anonymity Tools

I will evaluate a range of tools for online anonymity aganist the following criteria: 1. Ease of install 2. Impact on network performance 3. Degree to which aims are achieved 4. Useability 5. Side Effects (positive + negative)

2 Scope

- Consider a range of tools from simple tools such as broswer plugins to more complex solution such as VPN's
- Only consider open source tools that are avaliable on linux or andriod (don't need to consider mac or windows)
- Only use hardware set out below
- Only use the networks set out below

3 Hardware

3.1 Laptop (typically as a client) running Ubuntu 12.04

System Specs

- OS: Ubuntu 12.04 LTS
- Memory: 3.7. GiB
- Intel Core i3
- 64 bit
- Disk: 25.9 GB

3.2 Desktop (typically as a server) running Ubuntu 12.04

3.3 Raspberry pi

3.4 Andriod Phone

The andriod phone that I am choosing to work with, is as follows:

- Model: HTC Dream
- Android Version: 2.2.1
- Mod Version CyanogenMod-6.1.1-DS
- Biuld Number FRG83

In order to work with this phone I've install the SDK starter package and the ADT Plugin for Eclipse. Then from Eclipse, I've used the Android SDK Manager to get the SDK platform, samples and API for Android 2.2 (the version on the phone)

Next, I created an Android Virtual Device (AVD) called my_HTC_Dream and follow the basic helloWorld tutorial. To get my apps onto the phone, i put the phone into USB Debugging mode and add the udev rule as follows sudo gedit /etc/udev/rules.d/51-android.rules Then into that file add SUBSYSTEM==”usb”, ATTRidVendor==”0bb4”, MODE=”0666”, GROUP=”plugdev” and execute chmod a+r /etc/udev/rules.d/51-android.rules

Before contiuning to look at loading my own applications on to andriod, I am going to take a look at applications avaliable on the andriod platform for anonymity online

4 Avaliable Networks

4.1 College ethenet (possible issues with the firewall/NAT) (typically for the server)

4.2 wgb Wi-Fi (typically for the client)

4.3 Lapwing (typically for the client)

4.4 Eduroam (typically git push -u origin masfor the cleint)

4.5 3G (possiblity for the android phone)

5 Software/Tools Used Investigations

5.1 Firefox browsers

5.2 Traceroute ??

5.3 Wireshark ??

5.4 iperf

The suggest tool for my investigations is iperf. Iperf is a netwrok testing tool that measures stats on the netwrok performace of TCP and UDP data

streams. When testing UDP capcity Iperf allows the user to specify the datagram size so I will need to decide a suitable size.

Jperf is a GUI for Iperf which is written in Java but I have choosen not to use this as I will be trying to automate my tests using a unix script and I hope to improve my knowledge of unix commend line.

Iperf has a client and server functionality to measure network performance between two ends either unidiectionally or bi-directionally. This means I need two computers to run my tests. One will be my desktop in my college room and the other will be my laptop in the computer labs. I intend to make the following measurements

5.5 dig / host

5.6 Octave (to analysis quantative results)

5.7 Freedom Box

5.8 CyanogenMod

5.9 Iodine ?

6 Project Tor (Qualative Anaylsis)

TOR, the ancroym for The Onion Rounter is a system designed to provide individuals with anonmity online. Tor works by routing internet traffic through a network of server to conceal the user's location and usage information. I am going to focus my investigations on the Tor Browser Bundle. The Browser bundle lets you run Tor directly of a USB flash drive without installation and includes a pre-configured firefox browser which further aims to protect the users identify by:

- ensuring that the browser to properly configured to send their traffic through Tor
- blocking browser plugins as they can be manipulated to reveal your IP address
- uses HTTPS Everywhere to force HTTPS encryption on major websites that support it
- warning the user before allowing them to open documents that are handled by external applications as this can reveal there non-Tor IP address

The TOR browser bundle includes Vidalia, a GUI for Tor that allows users start and stop tor, view a graphical representation of the current TOR network, switch to a new IP address, set up a relay and edit Tor's settings.

The homepage for the browser check.torproject.org confurms to the user that Tor is successfully working and returns to the user the current IP address that there network traffic appears to be coming from.

Looking at TOR from an alternative pospective, that of an adversary. An adversary may aim to do the following:

- Bypassing Tor - get the user to connect directly to an IP of an adversary's choosing
- Correlation of Tor and Non-Tor Activity - detecting what a tor user did then not using tor
- Identifying Tor Users - identifying the indivduals that use Tor
- History Disclosure - the ability to quiery to see if an individual has visited particular sites previously
- Identifying location - using information such as timezone to determine a individuals geographical location
- Miscellaneous anonymity set reduction - Link activity to an particular individual using obscure information
- History records / On-disk information - with physical access to a machine using caching and historical records to identify previous activity

An adversary can locate themselves where they have the best chance to attack, this could include as an exit node, upstream router, ad servers, hosting malicious websites, ISP or gain physical access. The Tor browser aims to protect against this forms of attack

Once of the issues what I quickly found with Tor was the speed of my internet connection compared to when I was not using Tor. I therefore coolected some data on the speed of Tor vs Non Tor connections and looked into whether the speeds achieved were sufficient to use Tor for everyday browsing.

To add context to the speeds which I am achieving I could investigate what speed are sufficient for different internet activities, but just because a speed is sufficient it don't mean that it means the users expectations. Instead I will consider the figures in the context of the UK's average broadband speeds. I will be using the data provided by "Wi-Fi in the Home - A Study

Tor		Non-Tor	
Download	Upload	Download	Upload
1282	697	9824	9170
1373	1316	9849	8200
1517	1807	8508	8273
1782	483	2454	311
1455	497	2320	453
1349	458	1781	621

Table 1: Comparing connection speeds acheived using Tor verse those achieved when not using Tor, all measurements given in Kbps

into the Effect of the 'air Mile' on Consumer Boardband Performance" produced by Epitiro in 2011. Epitiro found that the average UK Wi-Fi download speed was 6.1 Mbps

7 Tor (Quantative Anaylsis)

8 Planning

To begin with I'm collect data on the performace of Tor and the normal network performace for comparision. Later I intend to extend this to other solutions to analoy online. For now I am going to run all test on ubuntu on my laptop but later this can be extended to other platforms like android.

THe plan for my initial tests is as follows:

- Each test will be run over UDP and then TCP
- Each test will run for 24 hrs
- Each test will be run over Tor and without-Tor as a control
- The reley on Tor needs to be regularly changed
- The first network that I will run the test on is Edurom at WGB

9 Privoxy with Hamachi (Qualative Anaylsis)

Privoxy is a heavy costomizable non-caching web proxy. Privoxy can be used to conjection with Tor. Privoxy aims to allow user more fine-grained control

over thier internet experience. Privoxy acts as an middleman between the browser and webservers. Privoxy requests objects on behalf of the browser

9.1 Introduction

To enable me to encrypt my web traffic when using public Wi-Fi regardless of wheather a website supports HTTPS, to prevent attacks like Firesheep. I am going to set up an encrypted web proxy between my desktop in my college room and my laptop in a coffee shop.

The desktop in my room (the proxy) acts as a middleman between my laptop (the client) and the world wide web. This means that internet requests go from the client to the proxy and the proxy then executes that on the clients behalf and returns the result to the client.

The encryption is set up between the client and the server, this means that the traffic from the cleint can't be intercepted by other users on the public Wi-Fi in the coffee shop. The encryption only takes place been the client and the server, not between the server and the world wide web. But this is not important since the connection from the server to the world wide web is more secure than that directly from the client using public Wi-Fi in a coffee shop

9.2 The Plan

For this experiment I will be making use the follow software:

- Hamachi - a cross-platform VPN server the gives encrypted access to my server from my client
- Privoxy - A web proxy (as tested last week)

In over to do this experiment, I will need to IP address of my server. The server is a desktop in my college room and my college uses a NAT. So how can get the servers IP address if its behind a NAT ?

The university runs its own authorative DNS server, this maps domain names to each individuals computers. The domain name for my server is therefore:

hh360.pem.private.cam.ac.uk

I can use this domain name to refer to my server so that even if the IP address changes, I will still be able to access the server.

Now that I have the information that I need, I will:

- Install Hamachi on my server

- Set a new private network
- Install Hamachi on my client
- Get IP address of client (for use in testing)
- Join the newly created network
- Install Privoxy on my server
- Edit config.txt by changing listen-address to IP address created by Hamachi
- edit firefox preference to using the Hamachi-powered server as a HTTP proxy on port 8118
- Go to http://config.privoxy.org/ to check that privoxy
- Using dig or host to perform a DNS query for the server IP address
- Go to WhatIsMyIP.com and the IP address should be that of the server
- Run a couple of quick speed tests whilst using the proxy
- Turn off the proxy etain firefox, and go to WhatIsMyIP.com to get the IP address of the client again (compare to that found earlier)
- Now that the proxy is off, run a couple of quick speed tests
- From the client, using nmap to check port scan the servereta

9.3 Implementation

9.4 Conclusions and Results

10 Using Dreamplug as a FreedomBox (Qualative Anaylsis)

Globalscale's Dreamplug is an classic example of a Plug Computer, it sports a 1.2GHz ARM processor, 512MB of RAM, 1GB Storage as well as 2 ethernet port, SD card slot and Wi-Fi all running on 5V DC power. A plug computer is a small form factor computer often used as a server. Dreamplug has typical features of a Plug computer such as:

- low power consumption

- small form factor
- no video card
- good connectivity

Plug computer such as Dreamplug can be used as "FreedomBox". FreedomBoxes are personal home servers that aims to free individuals from centralised systems such as popular social networking sites and email servers and move forward to more distributed systems putting individuals in control of there own data and how its transmitted through the packaging of opensource software

In the future, new hardware such as the raspberry pi could be used as a freedom box, with a notably lower unit cost than hardware such as the Dreamplug.

11 Firefox Addons (Qualative Anaylsis)

12 OpenVPN (Qualative Anaylsis)

Firstly I'm going to look at using openVPN to set up a VPN between my android phone and laptop. To do this I will install openVPN on my laptop (as the server) and connect my andriod phone to the VPN (as the client).

OpenVPN can be downloaded straight from ubuntu repositories. Next I need set up a public-key infrastructure (PKI) which will allow my cleint and server to identify and authontiate each other. The PKI conists of a public and private key pair for the server and another pair for the cleint and a CA cerificate and key wich is used to sign the server and cleints public and private key pair.

To set up a CA and generate the keys and certificates

13 References