Term Project Term 1 2005-06
The Design of a Graphical User Interface for the anonymous network routing system Tor

EECE412 students:
Miljan Cabrilo, Gordon Jesso, Katayon Radkhah, Ivan Tsui

***Abstract*— In today's world the Internet represents an important tool for rapid exchange of ideas and information. It is a publicly accessible worldwide system of interconnected networks that allows organizations and individuals to share information. With the increasing popularity of the Internet many security issues have arisen. One of the most important issues of which most users are still not aware is called traffic analysis.**
In this case the content of the communication is not the most important thing. What is even more important, and thus can reveal even more information than the content is the information that can be received through traffic analysis. Traffic analysis tracks where your data goes and when, as well as how much is sent.
A solution for this problem is provided by the freely available tool called Tor. Unfortunately, this anonymous network routing system lacks an accessible user interface that makes it simple to set up for the average user. Hence, as our project we designed an easy-to-use graphical user interface (GUI) for the Tor program.

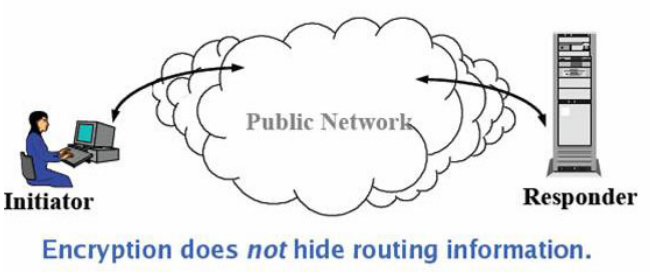
***Index Terms*— traffic analysis, anonymous network routing system, usability, graphical user interface**

I. Introduction

NOWADAYS the Internet is a broadly used tool for modern day communication and commerce, connecting people all over the world. This important communication and information medium allows people instant access to a vast and diverse amount of online information. For example, researchers obtain information about their research topics, or communicate with other researchers. Via the internet, journalists communicate with whistleblowers and dissidents. Soldiers write into online-diaries about their experiences and feelings. Non-governmental organizations allow their workers to connect to their home website while they are in a foreign country. Employers want to have access to their home website from anywhere in the world in order to send and receive relevant information. Politicians all over the world are informed about important decisions by using the world wide web. In general, this list can continue, there are lots more actions that are done by using the internet.

To conclude, the popularity of this tool is immense and increasing day by day. Thus, it is becoming more and more important to ensure the security of all the above described internet actions. In the past few years, a lot of successful research and work has been done to improve the safety and security as well as privacy on the internet. But still there is a major issue remaining which internet users are not aware of.

A relatively new threat is traffic analysis. It represents a form of network surveillance which instead of looking at the data sent, focuses on the destination and the source of the data. By using traffic analysis, it can be easily inferred who is talking to whom over a public network and where the communication participants are located. Indeed, this reveals a great deal about what you are doing, and possibly what you are saying.



Information on the internet is transmitted by using a standardized Internet protocol (IP) and many other protocols. Technical specifications or protocols like the IP describe how to exchange data over the network. "The Internet Protocol is a data-oriented protocol used by source and destination hosts for communicating data across a packet-switched internetwork", according to the wikipedia entry about IP. Data is divided into packets and send in blocks. Every packet includes a payload, e.i. the data content, and a header used for routing. Even if the payload is encrypted the header yields valuable information since it contains the message size, the source and destination.Indeed, the routing information has to be sent in clear because routers need to know packet's destinations, in order to route them in the right direction. Even if the header is hidden somehow, the packets can still be tracked as they move through the network. Thus, encrypting the payload is ineffective since the goal of traffic analysis is to identify who is talking to whom and not the content of the data.

An anonymous network routing system, called Onion Routing, can help to prevent this type of network surveillance, at the very least it makes it a great deal more difficult for attackers and observers to discover any information about the source and destination of the data sent. The aim of this relatively new protection mechanism is that Internet-based connections resist traffic analysis, eavesdropping, and other attacks both by insiders (e.g. Internet routers) and insiders (Onion Routing servers themselves). The transport medium does not know who

is communicating with whom, the network knows only that communication is taking place. Furthermore, the content of the communication is hidden from eavesdroppers up to the point where the traffic leaves the Onion routing network. Our project is based on the second-generation onion router, Tor.

II. Motivation

The onion routing network system focuses on anonymous connections rather than anonymous communication. Thus the question may arise why anonymous connections are so important? Indeed, we will see by the examples given in the following, that anonymous connections are required by many daily tasks and by various different person groups.

A good example where anonymous identities are required can be found in open source intelligence gathering via the web and the pseudonym based email communications that hide the true identities of both the sender and the receiver.

Anonymous connection is also important for socially sensitive communicant: chat rooms and web forums for rape and abuse survivors, or people with illnesses.

An employee who is travelling abroad and wants to connect to his employer's computers to check or send mail does not want to reveal his national origin and professional affiliation to any observer.

Anonymous connection is required when connecting to news sites or instant messaging services.

Online elections and voting require anonymous connection. Furthermore, dissidents and whistle-blowers want to communicate more safely with journalists. Anonymous connections allow people to set up web sites where people can publish material without worrying about censorship.

Governments, companies and individuals can keep track of where people and organizations go and what they do on the internet without knowing the content of your communication. This way they can track their behavior and interests.

To conclude, network surveillance threatens personal anonymity and privacy, confidential business activities and relationships, and state security. Hence, it is even more crucial to provide a user-friendly interface.

III. What is Tor

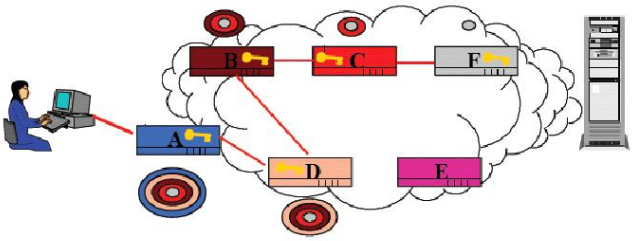
A. A brief overview of the history of onion routing

The Tor project, the second-generation onion routing system, was launched by The Free Haven Project in 2002. In the past, contracts with the Naval Research Lab (NRL) and the Electronic Frontier Foundation (EFF) funded its development. The Free Haven Project began in December 1999 as a research project initially compromised of several MIT students to design, implement, and deploy a functional data haven. The NRL is the corporate research laboratory for the Navy and Marine Corps and conducts a broad program of scientific research, technology and advanced development. It is also the birthplace of onion routing. For further information about these two cooperation partners we want to refer the reader to the references.

The Office of Naval Research (ONR) began work on Onion Routing in 1995. Many ideas were not implemented until the second generation. In 1996 a prototype is deployed on Solaris 2.5.1\2.6, consisting of a 5 node system running on a single machine at NRL with proxies for Web browsing. Two years later, a distributed network of 13 nodes is set up. Research and alysis work continues till work on generation 2 (Tor) code begins.

B. How does onion-routing work

As a traffic analysis resistant infrastructure the onion routing system dynamically builds anonymous connections within a network of Onion Routers (OR). These ORs bounce around communications in a distributed system of servers. They are roughly real-time Chaum Mixes. "A Mix is a store-and-forward device that accepts a number of fixed-length messages from numerous sources, performs cryptographic transformations on the messages and then forwards the messages to the next destination in an order not predictable from the order of the inputs"([10]). While ORs pass information in real time, a Mix can store messages for an indefinite amount of time while waiting to receive an adequate number of messages to mix together.

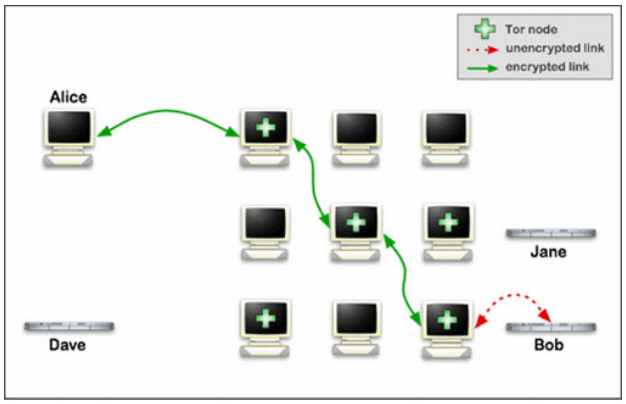


A session is launched by creating an onion which traverses a sequence of proxy servers. The task of these ORs contains rerouting messages in an unpredictable path. As we can see from the figure above the router at the head of a transmission selects a number of ORs at random and generates a message for each one of them. Each one of these messages is provided with the symmetric keys for decrypting the encapsulated message for the next route. So every router gets instructions about which router is next in the path. These instructions can be decrypted with the symmetric key which is encrypted with the corresponding OR's public key. Here we have a good example of hybrid encryption, using symmetric and asymmetric encryption. As a result, this method of creating messages provides a layered structure in which it is necessary to decrypt all outer layers of the onion in order to reach the most inner layer which has the main information about the destination. This leads to the onion metaphor: Every proxy server peels off a layer of the onion by decrypting it with its private key, revealing the routing instructions meant for that router which includs the encrypted instructions for all the routers located farther down the path. Thus, the full content of an onion can only be revealed if it is transmitted to every router in the path in the order specified by the layering.

The final OR connects to a responder proxy. This proxy will forward data to the remote application. After the connection

is established, data can be sent.

C. Tor-the second-generation onion routing system



Since the first-generation onion routing system had limited functionalities, an improved new circuit-based low-latency anonymous communication service, called Tor, was designed. In other words, the developers focussed on forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies, and location-hidden-services via rendezvous points. In the following some of these improvements will be explained:

- Perfect Forward Secrecy: Single hostile nodes are not able to record traffic and compromise successive nodes in the circuit and force them to decrypt it. This is achieved by an incremental path-building design, i.e. the initiator negotiates session keys with each successive hop. After these keys have been deleted, subsequently nodes in the path cannot decrypt old traffic. Thus, the process of building circuits is more reliable.
- end-to-end integrity checking: The earlier Onion Routing Design did not check the integrity of data. Thus, any node in the circuit was able to change the contents of data as they passed by. Tor prevents these attacks by verifying data integrity before it leaves the network.
- Rendezvous points and hidden services: The original Onion Routing System required “reply-onions” to connect with hidden services. In Tor, rendezvous points are negotiated by clients.
- Directory servers: Instead of flooding state information through the network certain nodes act as directory servers and describe known routers and their current state.

The Tor program is a freely available software that can run on the following operating systems: Linux, BSD, OS X, Windows, Solaris. It can help to reduce the risks of both simple and sophisticated traffic analysis. However, it is still not widely used because it lacks a user-friendly interface. According to estimations made by the Tor designers, about 50,000 people are using Tor, routing their traffic through about 250 volunteer Tor servers on six continents. The security of Tor is actually due to this variety of people who use it. The more people use Tor the more they will be hidden among other users on the network. Thus, the more users will be protected. By providing a user-friendly interface for Tor we intend to increase the number of Tor users.

IV. Analysis of Tor’s Security

The designers of Tor adhered to a number of security system design principles:

- The **Principle of least Privilege** is demonstrated in the Onion router’s level of access to incoming messages. The precomputed path a message will take in order to reach its destination is encrypted in such a way that it provides as little information as is necessary to transmit the message. Each Onion Router knows what a router a message came from and is able to decrypt where to send the message next. Thus, the router can extract only the information it needs in order to perform the next step in the chain.

- As a consequence of the above Tor design also exhibits the **Principle of fail-safe defaults**. At no node is an Onion Router capable of seeing the complete path a message will take, this essentially places it in a constant state of fail-safe operation. In addition to this, each Tor circuit is valid for only a couple of minutes, limiting the amount of time a potential attacker has to compromise the communication system.

- The **Principle of Economy of Mechanism** requires that a security mechanism be as simple as possible. Tor demonstrates the application of this principle through frequent testing of Onion routers and consequent updates of the directory listings.

- The **Principle of complete mediation** states that caching of access related information should be avoided. By continuously reconfiguring message paths and checking the availability of Onion Routers Tor follows this principle.

- Another design principle Tor adheres to is the **Principle of Open Design** which states that the security a mechanism provides should not depend on secrecy of the design. The developers provide documents that among other things render a security analysis of the Tor design and an insight into its operation.

- The **Principle of Separation of Privileges** requires that multiple conditions should be used to grant access to an object. This is demonstrated in Tor through the use of multiple encryptions in order to hide a path a message will take.

- The **Principle of Least Common Mechanism** ensures that a hostile cannot overwhelm a system, thus creating a denial of service attack. The designers of Tor followed this principle by providing circuit-level and stream-level throttling that ensures a single user cannot hijack the

complete system.

- The **Principle of Psychological Acceptability** states that a security mechanism should be easy to set up, operate and should provide a negligible burden to the overall system performance. Presently Tor does not adhere to this principle as there is no user friendly way of controlling the system.

- The **Principle of Design in Depth** states that a security system should be designed in a layered fashion. This is demonstrated by the usage of Onion Routers. An attacker would have to compromise multiple router.

- Tor developers have considered a number of threats to different aspects of their system. The history of its development reveals that they continue to adjust Tor to newly discovered threats. Hence, it also follows the **Principle of Questioning Assumptions**.

V. Achievements

Onion Routing achieves two goals: First of all the content of a message is hidden and no information about the participating principals is openly available. It provides strongly private communications in real time over a public network at reasonable cost and efficiency.

Secondly, it is possible to communicate with each other without being able to identify the other partner, either sender or recipient. Identification can be separated from routing.

This flexible communication infrastructure is resistant to both eavesdropping and traffic analysis. It can be employed to ensure anonymous web browsing and publishing, instant messaging and internet relay chat.

VI. Problem: Usability and Security Tradeoff

The current interface of Tor is uninviting and difficult to use. The user cannot directly interact with the Tor process. Furthermore, it is difficult to find the configuration files.
Currently, Tor runs as a daemon in the background and in order to change any settings the user must manually edit the configuration files and restart the service. The above discussed problem clearly shows that usability is regarded as a crucial requirement for security. The Tor developers define usability as follows:

> "A system is usable if it allows the user to see and\or manipulate (as appropriate) all and only the relevant information, using the fewest number of least error-prone gestures possible"([12]).

VII. The Graphical User Interface-Our Solution

The GUI provides a simple method of getting the Tor process and the Internet browser Firefox to work with minimum hassle to the user. As a prototype the GUI offers the basic features, such as launching Firefox and enabling and disabling Tor usage in Firefox.

The user needs to install the Tor programm and Privoxy which is a filtering web proxy that integrates well with Tor and Firefox 1.5. As a result the mere action of starting our GUI will immediately setup Firefox to start using the Tor Onion Network.

Four main classes have been written for the GUI:

1) The class "GUI.java" represents the main programm that creates the interface and the tasktray icon.
2) The class "configuration.java" allows the GUI to save settings between sessions.
3) The class "torControl.java" contains the necessary functions to send commands, i.e. to restart, to get version,etc., to the Tor daemon as well as a method to change configuration settings. The class also provides the function to kill the Tor process. The final component of torControl checks and configures the proxy server Privoxy to link up with Tor.
4) The class "firefoxControl.java" contains functions that will start\exit firefox and enable\disable Tor's proxy configuration. There is an additional code that will make a backup of the original Firefox configuration.

The Tasktray-functionality was provided by Snoozesoft's system tray (for further information see http://systray.sourceforge.net/doc/snoozesoft).

The Tor connection classes are provided by the Tor developers on their website (see References). The Process killer is a freeware application.

As for difficulties, we had to find out how Tor could be controlled as well as discover where Firefox stored its settings. At first we could not disable Tor usage for Firefox. However, upgrading to Firefox 1.5 seemed to solve this problem. Although Tor had an internal command to halt itself, using that command crashed the main program. As such we are using a freeware program and executing it from inside the program to kill the Tor process.

Firefox needs to be restarted for new configuration settings to take hold, therefore, the Firefox process is also killed before its configuration is modified.

VIII. Conclusions

As the result of our term project we can present a highly usable interface for using the onion routing. It represents an alternative to the current way of using Tor.

Our interface allows the average user to quickly and easily set up Tor rather than manually searching for and opening text files. Tor users are able to learn about the current state of their Tor connection, in other words they can find out how well the current connection is working and whether any applications are using it. Emphasis is placed on ease of configuration and abstraction of the current Tor interface.

Furthermore, the user is able to monitor the availability of Tor servers. This is because the status of the local Tor process is integrated into the GUI. The implemented GUI helps users to easily and quickly set up multiple profiles and configure Tor.

Additional desirable functionalities we intend to implement are the following:

- provide detailed information about which applications, ports, or packets are (or are not!) passing through Tor
- provide additional statistics about the Tor connection
- allow users to control more over how Tor behaves at certain times of day.

Our prototype will hopefully substantially increase the usage of the anonymous network routing system in the future. But it should be emphasized that Tor does not assert its claims for totally strong anonymity as it is still in the development stage. Thus, if really strong anonymity is required one should not rely solely on the current Tor network.

References

[1] Bishop, Matt: *Computer Security*, Addison-Wesley, 2003.
[2] Dingledine, Roger; Mathewson, Nick; Syverson, Paul: *Tor: The Second-Generation Onion Router*, in Proceedings of the 13th USENIX Security Symposium, August 2004.
[3] Goldschlag, David; Reed, Michael; Syverson, Paul: *Onion Routing Access Configurations*, DISCEX 2000: Proceedings of the DARPA Information Survivability Conference and Exposition, Volume I Hilton Head, SC, IEEE CS Press, January 2000, pp. 34–40.
[4] Goldschlag, David; Reed, Michael; Syverson, Paul: *Onion Routing for Anonymous and Private Internet Connections*, Communications of the ACM, vol. 42, num. 2, February 1999.
[5] Goldschlag, David; Reed, Michael; Syverson, Paul: *Anonymous Connections and Onion Routing*, IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection, 1998.
[6] Goldschlag, David; Reed, Michael; Syverson, Paul: *Privacy on the Internet*, INET '97, Kuala Lumpur, Malaysia, June 1997.
[7] Goldschlag, David; Reed, Michael; Syverson, Paul: *Anonymous Connections and Onion Routing*, Proceedings of the 18th Annual Symposium on Security and Privacy, IEEE CS Press, Oakland, CA, May 1997, pp. 44-54.
[8] Goldschlag, David; Reed, Michael; Syverson, Paul: *Proxies for Anonymous Routing*, Proceedings of the 12th Annual Computer Security Applications Conference, IEEE CS Press, San Diego, CA, December 1996, pp. 95-104.
[9] Landwehr, Carl; Reed, Michael; Syverson, Paul; Tsudik, Gene: *Towards an Analysis of Onion Routing Security*, Workshop on Design Issues in Anonymity and Unobservability Berkeley, CA, July 2000.
[10] Reed, Michael; Syverson, Paul: *Onion Routing*, Proceeding of AIPA '99, March 1999.
[11] Syverson, Paul: *Making Anonymous Connections*, National Science Foundation, 8th June 2004.
[12] http://tor.eff.org/index.html.en