Proxies for Anonymous Routing

Michael G. Reed, Paul F. Syverson, and David M. Goldschlag
Naval Research Laboratory

Center for High Assurance Computer Systems
Washington, DC 20375-5337
Phone: +1 202.767.2389 (voice)
Fax: +1 202.404.7942 (fax)
e-mail: {reed, syverson, goldschlag}@itd.nrl.navy.mil

Abstract

Using traffic analysis, it is possible to infer who is talking to whom over a public network. This paper describes a flexible communications infrastructure, onion routing, which is resistant to traffic analysis. Onion routing lives just beneath the application layer, and is designed to interface with a wide variety of unmodified Internet services by means of proxies. Onion routing has been implemented on Sun Solaris 2.4; in addition, proxies for World Wide Web browsing (HTTP), remote logins (RLOGIN), e-mail (SMTP), and file transfers (FTP) have been implemented.

Onion routing provides application independent, real-time, and bi-directional anonymous connections that are resistant to both eavesdropping and traffic analysis. Applications making use of onion routing's anonymous connections may (and usually should) identify their users over the anonymous connection. User anonymity may be layered on top of the anonymous connections by removing identifying information from the data stream. Our goal here is anonymous connections, not anonymous communication. The use of a packet switched public network should not automatically reveal who is talking to whom. This is the traffic analysis that onion routing complicates.

1. Introduction

1.1 The Problem

Using traffic analysis, it is possible to infer who is talking to whom over a public network (Figure 1). For example, in a packet switched network [11], packets have a header used for routing, and a payload that carries the data. The header, which must be visible to the network (and to observers of the network), reveals the source and destination of the packet. Even if the header were obscured in some way, the packet could still be tracked as it moves through the network. Encrypting the payload is similarly ineffective, because the goal of traffic analysis is to identify who is talking to whom and not (to identify directly) the content of that conversation.

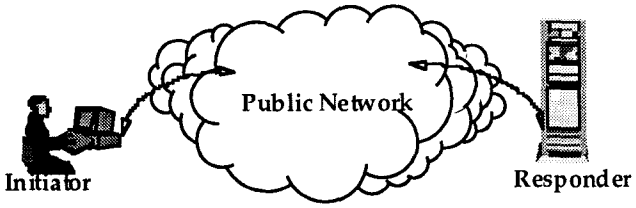

Figure 1. Communication over a Public Network

The efficiencies of the public Internet are strong motivation for companies to use it instead of private intranets. However, these companies may want to protect their interests. For example, a researcher using the World Wide Web (Web) may expect his particular focus to remain private, and inter-company collaborations should be confidential. Individuals may wish to protect their

privacy as well. For example, the sending of e-mail should keep the identities of the sender and recipient hidden from observers. Also, a person shopping online may not want his visits tracked. Certainly someone spending anonymous e-cash would expect that the source of the e-cash be untraceable.

The use of a packet switched public network should not require revealing who is talking to whom. This paper presents a flexible communications infrastructure, *onion routing*, which is resistant to traffic analysis.

1.2 Objective

Onion routing is an infrastructure that

- complicates traffic analysis,
- separates identification from routing,
- supports many different applications.

Without dedicated links between every node and full utilization of each link, traffic analysis can, in principle, always be effective. But traffic analysis can be made more costly. Onion routing accomplishes this goal by separating identification from routing. Onion routing provides *anonymous connections* that are resistant to both eavesdropping and traffic analysis. Instead of containing source and destination information, packets moving along an anonymous connection contain only next hop and previous hop information. These anonymous connections can replace socket connections. Since socket connections are commonly used to support applications running over the Internet (like Web browsers, remote login, and e-mail) onion routing's anonymous connections can support a wide variety of unmodified applications using *proxies* that interface between applications and the onion routing network.

1.3 Overview of the Solution

Onion routing works in the following way: An application, instead of making a (socket) connection directly to a destination machine, makes a connection to an *onion routing proxy* on some remote machine. That onion routing proxy builds an anonymous connection through several other *onion routers* to the destination. Each onion router can only identify adjacent onion routers along the route. When the connection is broken, even this limited information about the connection is cleared at each onion router. Data passed along the anonymous connection appears different *at* and *to* each onion router, so data cannot be tracked en route and compromised onion routers cannot cooperate. An onion routing network can exist in several configurations that permit efficient usage by both large institutions and individuals.

1.4 Traffic Analysis

Traffic analysis makes inferences from three sources of information:

- Routing information
- Coincidences
- Load

Routing information is available in many forms: packet headers, phone touch-tones, and envelope addresses. This is the most obvious source that needs protecting. Coincidences, like similar traffic entering or leaving a node, or connections opening or closing at roughly the same time, are more difficult to hide. Finally, the very presence of communication over some link may reveal sensitive information. But load is very difficult to obscure if one is unwilling to use a constant amount of capacity all the time.

1.5 Organization of Paper

This paper is organized in the following way: Section 2 presents background information. Section 3 presents our goals and threat model. Section 4 presents our solution, and sections 5 and 6 provide more details. Section 7 describes the implemented prototype. Section 8 discusses vulnerabilities, costs, and variants of onion routing. Section 9 presents some concluding remarks.

2. Background

Chaum [1,2] defines a mechanism for routing data through intermediate nodes, called *mixes*. These intermediate nodes may reorder, delay, and pad traffic to complicate traffic analysis. Our onion routers are based upon mixes.

Anonymous Remailers [4,6] use mixes to provide anonymous e-mail services and also invent an address through which mail can be forwarded back to the original sender. Remailers work in a store-and-forward manner at the mail application layer by stripping off headers at each mix and forwarding the mail message to the next mix. Some remailers provide confirmation of delivery.

In [8,9], mixes are used to provide untraceable communication in an ISDN network. In the described phone system, each telephone line is assigned to a particular local switch (i.e., local exchange), and switches

are interconnected by a (long distance) network. Anonymous calls in ISDN rely upon an anonymous connection within each switch between the caller and the long distance network, which is obtained by routing calls through a predefined series of mixes. The long distance endpoints of the connection are then mated to complete the call. (Notice that observers can tell which local switches are connected.) This approach relies upon two unique features of ISDN switches. Since each phone line has a subset of the switch's total capacity pre-allocated to it, there is no (real) cost associated with keeping a phone line active all the time, either by making calls to itself, to other phone lines on the same switch, or to the long distance network. Keeping phone lines active complicates traffic analysis because an observer cannot track coincidences.

Also, since each phone line has a control circuit connection to the switch, the switch can broadcast messages to each line using these control circuits. So, within a switch a truly anonymous connection can be established: a phone line makes an anonymous connection to some mix. That mix broadcasts a token identifying itself and the connection. A recipient of that token can make another anonymous connection to the specified mix, which mates the two connections to complete the call.

Our goal of anonymous connections over the Internet differs from anonymous remailers and anonymous ISDN. Unlike anonymous remailers, anonymous connections are application independent and are meant to be used by a wide variety of Internet applications. The data carried by anonymous connections is varied, with real-time constraints often more severe than mail, but usually somewhat looser than voice. Both Web and ISDN connections are bi-directional, but, unlike ISDN, Web connections are likely to be small requests followed by short bursts of returned data. In a local switch, capacity is pre-allocated to each phone line, and broadcasting is efficient. But broadcasting over the Internet is not free, and defining broadcast domains is not trivial. Most importantly, the network topology of the Internet is more akin to the network topology of the long distance network between switches, where capacity is a shared resource. In anonymous ISDN, the mixes hide communication within the local switch, but connections between switches are not hidden. This implies that all calls between two businesses, each large enough to use an entire switch, reveal which businesses are communicating. In onion routing, because of the topology of the Internet, mixing has to be dispersed throughout the Internet, so hiding is greatly improved.

3. Objectives

3.1 Applications

Onion routing's anonymous connections are designed to replace TCP/IP socket connections [3] and to be able to work with unmodified applications. A socket connection is a reliable bi-directional connection carrying a stream of data between two machines. Socket connections provide the abstraction that shields an application from the unreliable and unordered communication that is provided by lower levels of the IP stack.

Many applications use socket connections:

- Web requests (HTTP)
- Remote logins (RLOGIN)
- e-mail (SMTP)
- File transfer (FTP)
- Internet Relay Chat (IRC)
- Encrypted IP Tunnel

These applications can connect to onion routing's anonymous connections using *proxies*. A proxy [11] is usually a relay between an initiating and responding application. In onion routing, anonymous connections are terminated by application specific proxies that relay information between the connection and the unmodified applications. Many applications are already *proxy aware* because proxies are commonly used to communicate through firewalls. For example, a Web browser on a network with a firewall will reach sites outside the firewall through an HTTP proxy on the firewall machine. In that way, direct connections are never made between internal and external machines.

3.2 Threat Model: Active and Passive Attacks

Onion routing's design is very conservative since it assumes that the public network is very vulnerable. In particular, we assume that:

- All traffic is visible.
- All traffic can be modified.
- Onion routers may be compromised.
- Compromised onion routers may cooperate.

In addition, a sophisticated adversary may be able to detect timing coincidences such as the near simultaneous opening of connections. Timing coincidences are very

difficult to overcome, especially when real-time communication is important. But, if connections are routed over an unpredictable path in a busy network, this sort of attack is also very expensive.

The first four vulnerabilities, however, directly motivate certain design decisions in onion routing. Because traffic is visible, the headers and payloads of all traffic are essentially link encrypted between onion routers so the same data looks different when traveling between routers. Because traffic can be modified, stream ciphers [10] are used for encryption. Inserting, deleting, or modifying traffic en route will disrupt the stream and produce random bits downstream. Because onion routers may be compromised, anonymous connections span several onion routers, even though a single "perfect" mix is adequate to provide privacy. Because compromised onion routers may cooperate, data is encrypted in a layered fashion so it appears different to each onion router, not only between onion routers.

4. The Solution: Onion Routing

Onion routing has two parts: A network infrastructure that carries anonymous connections, and a proxy interfaces that mate these connections to unmodified applications.

4.1 Onion Routing: Network Infrastructure

The public network contains a set of onion routers. Each onion router has a single (socket) connection to each of a small set of neighboring onion routers. Onion routers only talk to their neighbors. Neighboring onion routers are neighbors for onion routing only. That is, communication between two neighboring onion routers is carried over a socket connection, and packets are routed (perhaps dynamically) through many hops by the IP protocol.

An anonymous *connection* is routed through a sequence of neighboring onion routers. Common segments of these routes are multiplexed over the single connection between neighbors. An onion router's obligation is to pass data from one connection to another after applying the appropriate cryptographic operations.

An anonymous connection from an initiator to a responder through four onion routers is illustrated in Figure 2.

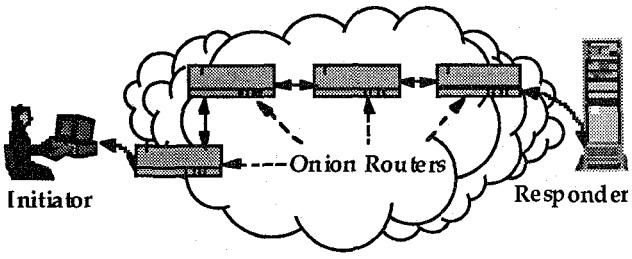

Figure 2. Onion Routing Network Infrastructure

4.2 Onion Routing: Proxy Interface

How are anonymous connections used? Proxies interface between applications and the network infrastructure. When a proxy is used on a firewall, it relays traffic between the protected site and the rest of the world. In onion routing, a proxy's functions are split into two: one part links the initiator to the anonymous connection and the other part links the anonymous connection to the responder. In this way, the initiating and responding applications need not be modified (although they do have to be able to use proxies).

Imagine an initiator sitting at her workstation using a Web browser. When she "clicks" on a URL link, the browser sends an HTTP request for that URL to some *onion routing proxy* instead of directly to the responder. In Figure 3, this is the onion routing proxy named W. W looks at the request and chooses a route through several other onion routers (e.g., W-X-Y-Z). W then sends an *onion* (see section 5.1) along that route; the onion is an instruction to those onion routers to construct an anonymous connection.

The last onion router in the route (Z) also functions as an onion routing proxy for the responder. Z passes data from the anonymous connection to the responder, and passes data from the responder back to the connection.

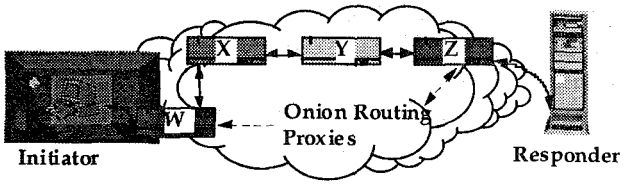

Figure 3: Onion Routing Proxy Interface

Instead of a single socket connection between an initiator and a responder, onion routing requires a socket connection between the initiator and his proxy, an anonymous connection between the initiator's proxy and the responder's proxy, and a socket connection between the responder's proxy and the responder. However, the three connections function as if they were a single (bi-

directional and real-time) socket connection between the initiator and responder.

There are many configurations of an onion routing network. In one basic configuration, a site that is concerned about traffic analysis should control an onion routing proxy in order to protect communication between that proxy and its users. That onion routing proxy must also function as an intermediate onion router in other anonymous connections. If it is not used in this way, observers can monitor the load coming from onion routing proxy and trace it back to the sensitive site. However, if the onion routing proxy is also a busy intermediate onion router, observers cannot tell whether the sensitive site is consuming, producing, or relaying traffic.

Individuals may access an onion router through their Internet Services Provider (ISP), if the ISP controls an onion routing proxy. An individual could also make an encrypted connection to some public domain onion routing proxy. Finally, a user could run an onion routing proxy on his workstation, and route anonymous connections through other onion routers.

5. Using Onion Routing

After the initiator contacts his proxy, onion routing follows four stages:

1. Define the route.
2. Construct the anonymous connection.
3. Move data through the anonymous connection.
4. Destroy the anonymous connection.

The next four sections describe these stages in more detail. (The extra details in each *Details* subsection are independent of the rest of the paper.)

5.1 Defining the Route

Consider Figure 3. The initiator's proxy, W, chooses to make an anonymous connection through (W-X-Y-Z). Therefore, W constructs a layered data structure called an *onion* (Figure 4):

Each layer of the onion is intended for a particular onion router and contains the identity of the next onion router in the anonymous connection, and the key that should be used when communicating with the previous onion router in the connection. The final layer of the onion is intended for Z. Since Z is the last onion router in the connection, its layer only contains a key.

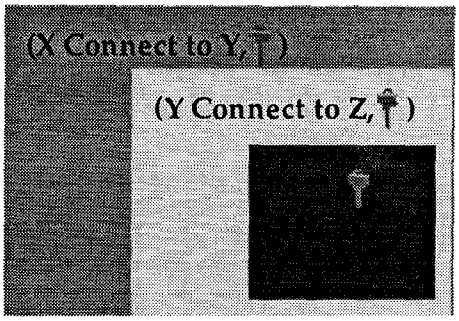

Figure 4: An Onion

Using public key cryptography [10], the onion is constructed so only the intended recipient can peel off the outermost layer, thereby revealing both his layer and the onion embedded inside. No recipient knows who created the onion. So, onion routers can identify only whom they received an onion from and to whom they are obliged to send the embedded onion. And, no recipient can determine what the other onions embedded in an onion look like.

The onion routing proxy that creates an onion keeps a copy of the keys in the onion until the anonymous connection is destroyed. We will see how these keys are used in sections 5.3 and 5.4.

5.1.1 Onion Details

The onion routing proxy routes the anonymous connection through neighboring onion routers. Therefore, it must know the topology of the onion routing network.

The size of an onion limits the length of a route. To prevent observers from inferring the length of a route, onions are padded to some fixed size. This padding becomes part of and is indistinguishable from the already embedded onion.

The key at each layer of the onion is used for bi-directional communication between an onion router and the previous onion router. Therefore, the key really specifies two stream ciphers, one for forward communication (in the direction the onion travels) and the other for backward communication (in the opposite direction).

Each layer of an onion also contains an expiration time. An onion router is to ignore an expired onion and is to ignore replayed onions. Therefore, onion routers must keep track of onions during their lifetimes.

For efficiency, the entire onion is not encrypted using a public key cryptosystem. Instead some prefix (corresponding to the block size of the public key

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.