Onion Routing Access Configurations

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

Abstract

Onion Routing is an infrastructure for private communication over a public network. It provides anonymous connections that are strongly resistant to both eavesdropping and traffic analysis. Thus it hides not only the data being sent, but who is talking to whom. Onion Routing's anonymous connections are bidirectional and near real-time, and can be used anywhere a socket connection can be used. Proxy aware applications, such as web browsing and e-mail, require no modification to use Onion Routing, and do so through a series of proxies. Other applications, such as remote login, can also use the system without modification. Access to an onion routing network can be configured in a variety of ways depending on the needs, policies, and facilities of those connecting. This paper describes some of these access configurations and also provides a basic overview of Onion Routing and comparisons with related work.[1]

Keywords: Security, privacy, anonymity, traffic analysis.

1 Introduction

Preserving privacy means not only hiding the content of messages, but also hiding who is talking to whom (traffic analysis). Much like a physical envelope, the simple application of cryptography within a packet-switched network hides the messages being sent, but can reveal who is talking to whom, and how often. Onion Routing is a general purpose infrastructure for private communication over a public network. It provides anonymous connections that are strongly resistant to both eavesdropping and traffic analysis. The connections are bidirectional, near real-time, and can be used for both connection-based and connectionless traffic. Onion Routing interfaces with off the shelf software and systems through specialized proxies, making it easy to integrate into existing systems. Prototypes have been running since July 1997. As of this article's submission, the prototype network is processing more than 1 million Web connections per month. Connections have come from more than thirty thousand IP addresses in more than sixty countries and in all six main top level domains [12].

Onion Routing operates by dynamically building anonymous connections within a network of real-time Chaum mixes [3].[2] 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 a random order. A single mix makes tracking of a particular message either by specific bit-pattern, size, or ordering with respect to other messages difficult. By routing through numerous mixes in the network, determining who is talking to whom is made even more difficult. Onion Routing's network of core onion routers (mixes) is distributed, fault-tolerant, and under the control of multiple administrative domains, so no single onion router can bring down the network or compromise a user's privacy, and cooperation between compromised onion routers is thereby confounded. The prototype network is entirely under our control and connections to it are not protected. Thus, the amount of protection is limited and subject to the trust of our administrative domain. However, arrangements are currently well under way for a network consisting of NRL controlled onion routers as well as onion routers controlled by independent commercial companies. By modifying entrance configurations and exit policies, Onion Routing is completely compatible with a wide variety of policies regarding resistance to traffic analysis and other security needs. This is de-

*Address: Naval Research Laboratory, Center For High Assurance Computer Systems, Washington, D.C. 20375-5337, USA, phone: +1 202.767.2389, fax: +1 202.404.7942, e-mail: syverson@itd.nrl.navy.mil, reed@itd.nrl.navy.mil, david@goldschlag.com.

[1] Versions of portions of this paper have appeared in [7, 13, 15]. This work supported by ONR and DARPA.

[2] This paper provides only a very brief description of Onion Routing system components. For a more detailed description, cf., [13].

scribed below in section 5.

2 Application Support via Proxies

Onion Routing can be used with applications that are proxy-aware, as well as several non-proxy-aware applications, without modification to the applications. Currently supported protocols include HTTP, FTP, SMTP, rlogin, telnet, NNTP, finger, whois, and raw sockets. Proxies are under development for Socks5, DNS, NFS, IRC, HTTPS, SSH, and Virtual Private Networks (VPNs). A proxy has three logical layers: an optional application specific privacy filter that sanitizes the data streams; an application specific layer that translates the data streams into an application independent format accepted by the Onion Routing network; and lastly, an onion building layer that builds and manages the anonymous connections. Because it builds and manages the anonymous connections, the proxy is the most trusted component in the system. Also, to build onions and hence define routes the proxy must know enough about the topology and link state of the network, the public certificates of nodes in the network, and the exit policies of nodes in the network. This information is distributed securely within the network automatically as new nodes come on-line or as the information changes.

3 Anonymous Connections

Onion Routing's anonymous connections are protocol independent and exist in three phases: connection setup, data movement, and connection tear-down. Setup begins when the initiator creates an onion, which defines the path of the connection through the network. An onion is a (recursively) layered data structure that specifies properties of the connection at each point along the route, e.g., cryptographic control information such as the different symmetric cryptographic algorithms and keys used during the data movement phase. The onion is treated as a destination address by onion routers; thus, it is used to establish an anonymous connection. Onions themselves appear differently to each onion router (OR) as well as to network observers. (The same goes for data carried over the connections they establish.) Connection routes can traverse an arbitrary number of ORs. A single onion can set up a route of eleven hops. Longer routes require tunnelling of onions through a connection. Each OR along the route uses its public key to decrypt the entire onion that it receives. This operation exposes the cryptographic control information, the identity of the next OR, and the embedded onion. The OR pads the embedded onion to maintain a fixed size, and sends it to the next OR. After the connection is established, data can be sent in both directions. Data from the initiator is repeatedly pre-encrypted using the algorithms and keys that were specified in the onion. As data moves through the anonymous connection, each OR removes one layer of encryption as defined by the cryptographic control information in the onion defining the route, so the data arrives as plaintext at the recipient. This layering occurs in the reverse order (using different algorithms and keys) for data moving backward. Connection tear-down can be initiated by either end, or in the middle if needed.

All information (onions, data, and network control) are sent through the Onion Routing network in uniform-sized cells. All cells arriving at an OR within a fixed time interval are mixed together to reduce correlation by network insiders. Likewise, the longstanding connections between ORs can be padded and bandwidth-limited to foil external observers. An onion looks different to each OR along a connection because of the layered public-key cryptography. Similarly, the layering of symmetric cryptography over the data phase cells makes them appear different to each OR. This design resists traffic analysis more effectively than any other deployed mechanisms for Internet communication.

4 Connection Overhead

Onion Routing's overhead is relatively small. Connection setup overhead is typically much less than one second and appears to be no more noticeable than other delays associated with normal Web connection setup on the Internet. Computationally expensive public-key cryptography is used only during this connection setup phase. Also, because public-key decryption is much more expensive than encryption, the public-key burden rests mainly upon the onion routers themselves, where the option of dedicated hardware acceleration can be justified. Our modular design is completely compatible with doing the public-key operations in either hardware or software, and we are using both in our test networks. (Aside: Because there is no cryptography in the system code itself, Onion Routing system code has already been approved for unlimited distribution.)

The data movement phase uses only secret-key (symmetric) cryptography, which is much faster. Furthermore, since the symmetric encryption can be pipelined, data throughput can be made as fast as ordinary link or end-to-end encryption. Data latency is affected by the number of ORs along the connection and can vary with route length and the duration of the mix cycles.

5 Access Configurations and Exit Policies

Proxies, onion routers, and other components can be run in a variety of distributed configurations. This allows Onion Routing to mesh well with a wide variety of operational and policy environments. We now consider some of these possibilities for access configurations.

- **Remote-Proxy Access**

  At one extreme, proxies can run remotely. If a user makes an encrypted connection to a trusted remote proxy, Onion Routing's protection can be utilized without installing any software or inducing local computational overhead. If the initiator can trust the remote proxy to build onions, his association with the anonymous connection from the first OR to the responder is hidden from observers and the network. In a similar way, an encrypted connection from an exit funnel (demultiplexor) to a responder hides the association of the responder with the anonymous connection.

  Therefore, if an initiator makes an anonymous connection to some responder, and layers end-to-end encryption over that anonymous connection, the initiator and responder can identify themselves to one another, yet hide their communication from the rest of the world. So, we can build virtual private networks without protected sites.

  Notice, however, that the initiator trusts the remote proxy to conceal that the initiator wants to communicate with the responder, and to build an anonymous connection through the OR network. The next paragraph describes how to shift some of this trust from a remote site to the initiator.

- **Customer-ISP Access**

  Suppose, for example, an Internet Services Provider (ISP) runs a funnel (multiplexor) that accepts connections from onion proxies running on subscribers' machines. In this configuration, users generate onions specifying a path through the ISP to the destination. Although the ISP would know who initiates the connection, the ISP would not know with whom the customer is communicating, nor would it be able to see data content. So the customer need not trust the ISP to maintain her privacy. Furthermore, the ISP becomes a *common carrier*, who carries data for its customers. This may relieve the ISP of responsibility both for whom users are communicating with and the content of those conversations. The ISP may or may not be running an OR as well. If he is running an onion router, then it is more difficult to identify connections that terminate with his customers; however, he is serving as a routing point for other traffic. On the other hand, if he simply runs a funnel to an onion router elsewhere, it will be possible to identify connections terminating with him, but his overall traffic load will be less. Which of these would be the case for a given ISP would probably depend on a variety of service, cost, and pricing considerations. Note that in this configuration the entry funnel must have an established longstanding connection to an OR just like any neighboring OR. In most other cases, where the funnel resides on the same machine as the onion router, establishing an encrypted longstanding connection should not be necessary since the funnel can be directly incorporated into the Onion Router.

- **Island-Unto-Yourself Access**

  If one wants to gain the maximum protection afforded by Onion Routing, it is necessary to have local control of an onion router. Assuming that this OR also serves as an intermediate node for routing of other traffic, not only data and route are hidden but also the time and volume information about connections originating or terminating locally. Of course this additional protection comes at the price of having adequate Internet bandwidth to function in this way.

- **Proxy-and-OR-at-Firewall Access**

  When a proxy and onion router sit on the firewall of a sensitive site, they can serve as an interface between machines behind the firewall and the external network. Connections from machines behind the firewall to the onion router are protected by other means (e.g., physical security). To complicate tracking of traffic originating or terminating within the sensitive enclave, this OR should also route data between other ORs. This configuration might represent the system interface from a typical corporate or government site.

  Connections between machines behind firewall ORs are protected against both eavesdropping and traffic analysis. Since the data stream never appears in the clear on the public network, this data may carry identifying information, but communication is still private. (This feature is used in constructing VPNs via Onion Routing.)

The onion router (more precisely the proxy) at the originating protected site knows both the source and destination of a connection. This protects the anonymity of connections from observers outside the firewall but also simplifies enforcement of and monitoring for compliance with corporate or governmental usage policy.

The use of anonymous connections between two sensitive sites that both control ORs effectively hides their communication from outsiders. Also, by employing a layering of funnels and ORs within firewalls, enclaves can incorporate traffic analysis resistance to their defense-in-depth.

- **Local-Proxy-with-OR-at-Firewall Access**

  It is possible to hide the route and origination of connections originating at an enclave while also protecting the route, application, and data being transmitted from enclave administrators. In this arrangement Onion Routing connected users are visible within the firewall, but not to where they are connected or what applications they are running.

The above discussion describes the various ways that a connection can enter an onion routing network. But, exiting is also important. Unless the responder of a connection is behind a firewall on which the terminal OR resides (e.g., if the responder is some arbitrary Web server) the data stream from the sensitive initiator must also be anonymized to avoid exposing the initiator. For example, an external attacker could simply listen in on the connections to a Web server and identify initiators of any connection to it. This point about exiting an OR network applies equally regardless of the configuration of entrance access.

There are other issues concerning how a connection exits an OR network. Exit points can also set policies for exiting based on where the traffic is going, and what application protocol is being run. Thus, for example, an onion router at a corporate firewall might allow anyone to attempt to remote login, but only to machines behind the firewall. It might allow email traffic to exit to any company site and Web traffic to exit to anywhere. Of course, these exit limitations might be a problem for the proxies attempting to create connections if they did not have the policies available when attempting to build a route.

There is a *database engine* attached to each of the onion routers. These ensure that any changes to this information propagates throughout the entire network to the proxies that construct routes. They also inform neighboring ORs about changes to network topology and link state, as well as ensuring that such information from others propagates throughout the network. In this way, proxies are given the most up-to-date information possible about potential routes. This greatly reduces the chance of bad connections that would then have to be attempted again via another route. The system that generates and distributes this information can be configured to be just as flat as the network itself, and authentication is of the relevant exit point or onion router. Thus, there is much less danger of central failure (or hostile manipulation) of this information. So, it is also more difficult to, e.g., manipulate this information to cause routes to pass through only compromised cooperating onion routers.

6 Background and Comparisons

As mentioned above, Chaum [3] defines a layered object that routes data through intermediate nodes (mixes). These intermediate nodes may reorder, delay, and pad traffic to complicate traffic analysis. In mixes, the assumption is that a single perfect mix adequately complicates traffic analysis, but a sequence of multiple mixes is typically used because real mixes are not ideal. Because of this, mix applications can use mixes in fixed order, and often do. Onion routers differ from mixes in using an indeterminate number of mixes in an indeterminate order, and in at least two other ways: onion routers are more limited in the extent to which they delay traffic at each node because of the real-time expectations that the applications demand of socket connections. Also, in a some Onion Routing access configurations, onion routers are also entry points to the onion routing network, and traffic entering or exiting at those nodes may or may not be visible to outsiders. This can make it hard to track packets, because they may drop out of the network at any node, and new packets may be introduced at each node. While Onion Routing cannot delay traffic to the extent that mixes can, traffic between ORs is multiplexed over a single channel and is link encrypted with a stream cipher. This makes it hard to parse the stream.

Anonymous remailers like Penet [9] strip headers from received mail and forward it to the intended recipient. They may also replace the sender's address with some alias, permitting replies. These sorts of remailers store sensitive state: the mapping between the alias and the true return address. Also, mail forwarded through a chain of remailers may be tracked because it appears the same to each remailer.

Mix based remailers like [4, 8] use mixes to provide anonymous e-mail services. Essentially, the mail message is carried in the innermost layer of an onion-like data structure. Another onion-like structure, used for a return address, can be contained in the message. This makes the return path self contained, and the remailer essentially stateless. Onion Routing shares many structures with Babel [8] but it uses them to build application independent end-to-end connections. This makes anonymous connections accessible to a wide variety of applications.

In [10], mixes are used to provide untraceable communication in an ISDN network. Here is a summary of that paper. In a 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 between the caller and the long distance network. These connections are made anonymous by routing calls through a predefined series of mixes within each switch. 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. The data is different, with real-time constraints more severe than mail, but somewhat looser than voice. Both HTTP and ISDN connections are bidirectional, but, unlike ISDN, HTTP connections are likely to be small requests followed by short bursts of returned data. As described in [10], 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, mixing is dispersed throughout the Internet, which improves hiding.

Onion Routing's flexibility with respect to access configurations also make it a natural complement to other services like the Anonymizer [1] and Proxymate [11].

The Anonymizer is a proxy Web site that filters the HTTP data stream to remove a user's identifying information. This makes Web browsing private in the absence of any eavesdropping or traffic analysis. The Anonymizer is vulnerable in three ways: First, it must be trusted. Second, traffic between a browser and the Anonymizer is sent in the clear, so that traffic identifies the true destination of a query, and includes the identifying information that the Anonymizer would filter. Third, even if the traffic between the browser and the Anonymizer were encrypted, traffic analysis could be used to match incoming (encrypted) data with outgoing data. Onion Routing's privacy filters provide a similar function to the Anonymizer. However, the Anonymizer's filters are perhaps the most up-to-date of any readily available filters for the ever changing means by which anonymity can be compromised in the data stream. Also, the high volume that this longstanding service attracts provides some degree of natural cover traffic. An Anonymizer could be used together with Onion Routing as the HTTP proxy front end to provide a nice interface and good filtering for anonymity, with strong resistance to both eavesdropping and traffic analysis. Security is improved because the filtering executes on a machine the user trusts, and communication leaving that machine will resist traffic analysis. Such security in depth removes the central point of failure for network traffic anonymity.

Proxymate (formerly known as LPWA) is a "proxy server that generates consistent untraceable aliases for you that enable you to browse the Web, register at web sites and open accounts, and be 'recognized' upon returning to your accounts, all while still *preserving your privacy.*" Proxymate thus provides various pseudonymy-based services. Like Onion Routing it is designed to handle email in addition to HTTP. And, like Onion Routing, it can be configured so that trusted functions are performed at various locations [2]. However, communication between and from these points is not itself anonymous or resistant to traffic analysis. This makes Proxymate and Onion Routing especially natural complements.

Pipe-net [5] is a proposal somewhat similar to Onion Routing. It has not been implemented, however. Pipe-net's threat model is more paranoid than Onion Routing's: it attempts to resist active attacks by global observers. For example, Pipe-net's connections are permanent and carry constant traffic (to resist timing signature attacks). And, disruptions to any connection are propagated throughout the network. This makes the design impractical for any short lived or large bandwidth connections and implies that the entire network shuts down if even one connection does so. Thus, it is also highly vulnerable to denial-of-service attacks. Pipe-net's design provides the strongest traffic analysis resistance guarantees of any given for connection-based communication infrastructures running over mix-like nodes. But, it accomplishes this at a very high price, so it is not likely to be implemented for large scale Internet use.

Crowds [14] is roughly a distributed and chained Anonymizer, with encrypted links between crowd members. Upon receiving traffic for the first time on a *path* a crowd member flips a weighted coin, and depending on the outcome, continues the path to another randomly chosen crowd member or terminates the path and forwards this (and any future traffic on the path) to its ultimate destination. Crowds is less general than Onion Routing, both in its applications (it is designed only for Web traffic) and its anonymity goals (there is no attempt to hide the ultimate destination of traffic from any node on the path).

Zero-Knowledge Systems [17] has designed a system with many similarities to Onion Routing. Beta versions are available of Freedom, their software for network access akin to the remote-proxy or customer-ISP access described above. Freedom also incorporates local pseudonym management and other features. However, the currently described Zero-Knowledge system appears to limit routes to a fixed length of three hops, which makes connections much more vulnerable to some forms of traffic analysis [16]. Also, the system design does not seen to be as naturally compatible as Onion Routing to enclave level traffic protection, on either the initiator or the responder end.

A natural extension to Onion Routing is the introduction of reply onions. Reply onions allow connections to be made back to an anonymous sender through an onion routing network long after the original connection existed. Reply onions could be used to send anonymous replies in response to a previously received anonymous email. They could also enable novel applications such as anonymous publishing (anonymous URLs) similar to the Rewebber project [6].

7 Conclusion

In summary, Onion Routing is a traffic analysis resistant infrastructure that is easily accessible, has low overhead, can protect a wide variety of applications, and is flexible enough to adapt to various network environments and security needs. The system is highly extensible, allowing for additional symmetric cryptographic algorithms, proxies, or routing algorithms with only minor modifications to the existing code base. Instructions for accessing the prototype network can be found on our Web page along with additional background, pointers to publications, and contact information [12].

References

[1] The Anonymizer. `http://www.anonymizer.com/`

[2] D. Bleichenbacher, E. Gabber, P. Gibbons, Y. Matias, and A. Mayer. "On Secure and Pseudonymous Client-Relationships with Multiple Servers", *Proc. Third USENIX Electronic Commerce Workshop* , Boston, Mass. Sept. 1998, pp. 99–108.

[3] D. Chaum. "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms", *Communications of the ACM*, v. 24, n. 2, Feb. 1981, pp. 84-88.

[4] L. Cottrell. "Mixmaster and Remailer Attacks", `http://obscura.obscura.com/~loki/remailer/remailer-essay.html`

[5] W. Dai. Pipe-net, February 1995, post to the `cypherpunks` mailing list.

[6] I. Goldberg and D. Wagner. "TAZ Servers and the Rewebber Network: Enabling Anonymous Publishing on the World Wide Web", *First Monday*, vol. 3 no. 4, April 1998.

[7] D. Goldschlag, M. Reed, P. Syverson. "Onion Routing for Anonymous and Private Internet Connections", *Communications of the ACM*, vol. 42, no. 2, February 1999, pp. 39–41.

[8] C. Gülcü and G. Tsudik.

"Mixing Email with *Babel*", in *Proceedings of the 1996 Symposium on Network and Distributed System Security*, San Diego, February 1996.

[9] J. Helsingius. `http://www.penet.fi/`

[10] A. Pfitzmann, B. Pfitzmann, and M. Waidner. "ISDN-Mixes: Untraceable Communication with Very Small Bandwidth Overhead", in *GI/ITG Conference: Communication in Distributed Systems*, Mannheim Feb, 1991, Informatik-Fachberichte 267, Springer-Verlag, Heildelberg 1991, pp. 451-463.

[11] Proxymate. `http://www.proxymate.com/`

[12] The Onion Routing Home Page. `http://www.onion-router.net/`

[13] M. Reed, P. Syverson, and D. Goldschlag. "Anonymous Connections and Onion Routing", *IEEE Journal on Selected Areas in Communications*, vol. 16 no. 4, May 1998, pp. 482–494.

[14] M. K. Reiter and A. D. Rubin. "Crowds: Anonymity for Web Transactions", *ACM Transactions on Information and System Security*, vol. 1 no. 1, November 1998, pp. 66–92 .

[15] P. Syverson, M. Reed, and D. Goldschlag. "Private Web Browsing", *Journal of Computer Security*, vol. 5 no. 3, 1997, pp. 237–248.

[16] P. Syverson, G. Tsudik, M. Reed, C. Landwehr. "On The Security of Onion Routing", Preprint.

[17] Zero-Knowledge Systems. `http://www.zerokowledge.com/`