H Tillwick and MS Olivier, "Bridging the gap between anonymous e-mail and anonymous Web browsing," *Online Information Review*, 32, 1, 22-34, 2008

Bridging the gap between anonymous e-mail and anonymous Web browsing

Heiko Tillwick and Martin S Olivier

Abstract

Purpose – Anonymity research on high latency, store-and-forward mediums, such as e-mail, have led to comparatively well-researched anonymity technologies; however, similar achievements have proven to be more difficult for low-latency communications such as Web browsing. This paper aims to propose an anonymous Web browsing protocol that harnesses some of the advantages of a store-and-forward anonymity solutions whilst retaining some of the interactive properties of Web browsing.
Design/methodology/approach – A review of existing anonymity solutions notes the advantages of mix technologies versus the advantages offered by Onion Routing. A solution is presented that features a combined approach of both solutions.
Findings – The proposed protocol differentiates between Web requests and Web responses – requests are treated as store-and-forward messages whilst the Web response is handled like a data stream.
Originality/value – The solution described can be used by existing anonymous Web browsing solutions in order to improve the level of anonymity whilst minimising the overhead of anonymously distributing Web content.
Keywords Anonymous Web browsing, anonymising proxy, privacy-enhancing technology
Paper type Conceptual

Introduction

Concerns have been raised about the difficulty of protecting personal privacy on the Internet (Chung and Paynter 2002). These concerns are particularly worrying in an increasing technology-dependent society where electronic information is readily available and easily distributable.

Some measures such as Web site privacy policies (Reagle and Cranor 1999), encryption and architectures involving trusted third parties (Mandujano and Shields 2000) offer a limited or conditional level of assurance or protection. However, privacy-related incidents are not uncommon. On the contrary, numerous examples (Sakalosky 2002, Anderson 2006) of unethical consumer data-use has lead us to believe that privacy on the Internet is still a largely unsolved problem.

Anonymity offers a useful form of privacy protection. Anonymity is viable when the use of a service does not require the exchange of personal information. Search engines are an excellent example of where anonymity could be particularly relevant. Search queries can reveal a great deal about an individual's work, hobbies, personal preferences and private life

(Aljifri and Navarro 2004). Consider having access to someone's search queries submitted over a period of a couple of months or years. Such information could be used for targeted marketing purposes or worse, could reveal incriminating or sensitive information.

By browsing anonymously one could prevent search engines or other Web sites from performing user tracking or user profiling. One might additionally want to be protected from organisational Web proxy's or Internet service providers (ISPs).

Indeed, the need to transact anonymously in society should apply similarly to online transactions. Technical solutions should allow users to transact electronically, and anonymously, as they otherwise would have done under traditional circumstances in society. Traditional anonymous activities include voting, counselling, whistle-blowing, refereeing and voicing political and other dissent.

Whilst Internet users can choose to withhold their identity whilst browsing, their Internet usage can still be traced back to a unique IP address. One might argue that proxies offer some level of protection as the user's IP address is hidden from the targeted Web server. However, proxies are likely to log requests producing a vulnerable audit trail of a user's activities.

This raises the following question: how does one effectively hide the requester's IP address from Web servers, organisational proxies, ISPs or other unauthorised observers?

A large body of research has been dedicated to building and analysing various forms of anonymising technologies. The types of anonymity provided by these technologies can be split into two categories: data anonymity and connection anonymity. Data anonymity deals with identifying information in the data itself, whereas connection anonymity protects the communication channel between the sender and receiver. Data anonymity has benefited from the relative maturity of cryptographic techniques. Connection anonymity still faces a number of challenges.

Connection anonymity research on high latency, store-and-forward mediums, such as e-mail, have led to comparatively well-established and robust technologies; however, similar achievements have proven to be more difficult for interactive, low-latency communication mediums such as Web browsing (Goldberg 2002, Tillwick and Olivier 2005). This could be partly attributed to the fact that anonymity techniques employed by high-latency, store-andforward systems translate poorly to performance-sensitive mediums. Users will not adopt anonymous Web browsing technologies that significantly degrade the browsing experience (Dingledine and Mathewson 2005). Similarly, sensitive users will not adopt an anonymity solutions that offer weak protection (Dingledine and Mathewson 2005).

This paper explores the relative success of store-and-forward anonymity solutions compared to their interactive, low-latency counterparts. The paper proposes an anonymous Web browsing protocol that harnesses some of the advantages of store-and-forward anonymity solutions whilst retaining some of the interactive properties of Web browsing. An attempt is made at bridging the gap between anonymous e-mail solutions and their Web browsing counterparts. Particular emphasis is placed on reducing the overhead of the anonymity protocol as decreased efficiency affects usability and final user acceptance.

The proposed protocol differentiates between Web requests and Web responses – requests are treated as messages whilst the Web response is handled like a data stream. Further anonymity-enhancing measures are introduced that increase the offered level of anonymity.

This paper is structured as follows: a review of existing anonymity technologies highlights their respective advantages and disadvantages and prepares the reader for a more technical discussion presented in the following section where the proposed protocol is introduced, discussed and analysed. Attack vectors are considered and the benefits of the proposed protocol, over existing protocols, is highlighted. Section concludes the paper.

Related Work

Two of the well-known, and often referenced, anonymising proxies include the Penet remailer (Helmers 1997) and Anonymizer (http://www.anonymizer.com). The Penet remailer is one of the first anonymous remailers. Replies to an anonymously sent e-mail are made possible through the use of pseudonyms. The remailer keeps a correspondence table mapping real e-mail addresses to pseudonymous addresses. Anonymizer similarly pursues anonymity by relaying Web requests through a single proxy.

These solutions, although simple and efficient, have numerous drawbacks. Because the service is centrally managed, it is susceptible to denial of service and compulsion attacks (which are achieved through extortion, bribing or legal subpoenas). For example, legal pressure forced the Penet remailer to reveal the identities of certain users (Helsingius 1996). The service has since shut down fearing further legal actions. A further threat is the possibility of a compromised system. This could result in all users' identities being revealed including a history of their Web or mail usage (in those cases where logs are kept). There is also the possibility that corrupt system administrators release or even sell information. Thus, a considerable amount of trust in the integrity of the system and the administrators is required.

Although Anonymizer suffers from these deficiencies it is still in use years after inception. This relative success can be attributed to a number of reasons: it is simple to use and requires no additional software, there are few suitable replacements and it caters for people who only require weak protection.

The past two decades have seen numerous improvements to both anonymous e-mail and anonymous Web browsing technologies. A common and noticeable trend is the move towards more distributed architectures. An objective is to distribute the trust an individual has to place in any one system component or system administrator.

Cypherpunk remailers (Parekh 1996) offer this advantage over the Penet remailer. Cypherpunk remailers allow for multiple proxies to be chained together. The remailer can be located in different jurisdictions making compulsion attacks more difficult and less probable. Reply blocks are used instead of correspondence tables; return addresses are thus no longer held with the remailer. Reply blocks accompany the e-mail and contain the encrypted return address. Only the respective remailer can decrypt it. A single corrupt remailer can not reveal the identity of the sender of the message.

Some peer-to-peer systems take the concept of distributed anonymity architectures even further.

Crowds (Reiter and Rubin 1998, Reiter and Rubin 1999) attempts anonymous Web browsing by collecting peers into a group called a crowd. Members of this group collaborate by forwarding Web requests amongst themselves before passing them to a specified Web server. The choice of forwarding to another proxy instead of to the end server is a random decision. Each member acts as a proxy, but is also able to issue its own request on behalf of the user. A Web server is thus only able to trace the request back to the final proxy and is not able to determine where the request originated from. In fact, every proxy along the route only knows the previous and the next proxy (or the end server). Only the original requester knows who issued the request. Sender anonymity is achieved by allowing members to get lost in a crowd.

Crowds suffers from a number of disadvantages.

- A central server for node discovery and key distribution is required. Crowds thus suffers from many of the trust issues typically related to client/server architectures.

- Each node establishes a static virtual path which is used for multiple requests. Once the path has been compromised all prior and subsequent requests are exposed.
- A simple routing algorithm with highly variable path lengths is employed. The initiating node has no means of controlling the length of the path resulting in inconsistent performance.
- Crowds requires all nodes on a virtual path to be honest nodes. Simple link-to-link encryption is employed between nodes. This requires intermediate nodes to decrypt and re-encrypt forwarded messages.

Issues which have not been addressed by any of the solutions discussed thus far include traffic analysis attacks. Even if traffic is encrypted, messages can still be traced back to an originator using a number of attacks. These attacks include message coding attacks, timing attacks and message length attacks; each is performed by correlating a proxies incoming and outgoing messages using their coding, time and length respectively.

Mixes

The dangers of traffic analysis were already perceived as early as 1981 when Chaum (1981) published details about a mix protocol that offers anonymous e-mailing. This protocol forms the basis for many anonymous e-mail solutions. The level of anonymity offered by the system is relatively high although it suffers from severe performance implications (thus limiting its inception to e-mail only).

The mix protocol involves a series of mixes that can be chained together. Each mix hides the correspondence between incoming and outgoing messages. This is achieved by

- imposing strict message-size-invariance through slicing and padding,
- performing cryptographic transformations on messages,
- batching and reordering of messages and
- providing cover traffic which (to an outsider) is indistinguishable to real traffic.

Cryptographic transformations are performed using public-key cryptography. A sender successively encodes an e-mail in layers of encryption corresponding to each mix in a chain of mixes. Each mix can only remove his respective layer of encryption. Removing a layer reveals the address of the succeeding mix as well as the next encoded message (now with one layer less than before). Decryption thus occurs in reverse order as the encryption process. This approach requires only one honest mix to preserve anonymity.

Numerous enhancements and variations of mixes exist (Díaz and Preneel 2004b). A detailed description and analysis of mixes is beyond the scope of this paper. The reader should merely note that mixes offer an increased level of protection (compared to previously discussed solutions) and that because of the performance and resource implications, mixes are mostly used for low-latency, store-and-forward mediums such as e-mail.

Onion Routing

Onion Routing is related to Chaum's (1981) original mix protocol but is more suited to interactive or even real-time media. Onion Routing was first conceived by Goldschlag, Reed and Syverson (1996) and aims to protect the privacy of the sender and receiver of a message

whilst also preventing a compromise of the system through any number of (except all) compromised routers.

As with mixes, Onion Routing also uses layered encryption. The difference lies in the content that is encrypted. Whereas mixes wrap the actual secret, Onions are used to distribute symmetric keys between successive Onion Routers in a chain of Onion Routers. Onion Routers thus establish an anonymous, bi-directional virtual channel between two communicating parties (via a number of participating routers). The virtual channel then relies on computationally less expensive symmetric keys to encrypt a stream of data.

Onion Routing focuses on a strong insider threat model. Unlike mixes, Onion Routers fail to protect against a number of traffic analysis attacks. No attempt is made to hide or obscure related information such as message size or send or receive times. No batching or reordering of messages is performed. This invariably means that the level of anonymity offered by Onion Routing is not as high as that achieved by mixes. A popular implementation of Onion Routing is called Tor (Dingledine, Mathewson and Syverson 2004) and offers some minor improvements over the original design.

The remainder of the paper proposes a new means of harnessing some of the benefits of mix technologies whilst retaining the interactive suitability of Onion Routing.

Protocol

This paper wishes to propose an enhanced Web browsing protocol that aims to bridge the gap between anonymous e-mail solutions and their Web browsing counterparts. The protocol should specifically cater for the Hypertext Transfer Protocol (HTTP) (Berners-Lee, Fielding and Frystyk 1996, Fielding, Gettys, Mogul, Frystyk, Masinter, Leach and Berners-Lee 1999) as this is the most commonly used protocol to request and transfer Web content.

Web requests and responses

HTTP is a request/response protocol: a Web response is only issued on receipt of a Web request. HTTP/1.1 (Fielding et al. 1999) differentiates between eight different types of requests. Particular attention is drawn to the GET, POST and PUT requests. This differentiation is important because it gives an indication of the size of the request. GET requests are most common and typically small in size. Only the URL is passed in the request (besides some possible other header fields). A POST message is used to submit additional HTML form data. One can generally expect POST requests to be somewhat larger than GET requests. Nevertheless, Web requests are generally relatively small compared to the average Web response. This holds true for all Web requests except for the PUT request; the PUT request uploads a potentially large resource.

Although HTTP does not enforce any rigid size categories, the intended use of each type implies a certain size expectation. Web responses are expected to be considerably larger than Web requests (with the exception of some PUT requests). This should be obvious when considering that Web servers generally host requested content.

We wish to illustrate how the size distinction between Web requests and Web responses allows our proposed anonymising protocol to harness advantages offered by mix and Onion Routing technologies. We believe that store-and-forward communications are not suitable for large Web responses. This is mainly due to the high resource constraints – simply forwarding data is cheaper than storing and then forwarding it. However, this disadvantage is not as

severe if the message content is small. Smaller Web requests could benefit from the existing anonymity techniques which are available to store-and-forwards mediums.

We wish to show how a Web request and not necessarily a Web response could expose the identity of a Web surfer. An anonymity-improving technique is proposed that prevents the Web response from exposing the identity of the original requester. Once this has been achieved, one can proceed to secure Web requests using mixing technologies. This approach increases the offered level of anonymity whilst minimising the overhead of transferring Web responses.

This paper thus offers an anonymous Web browsing protocol that offers an alternative approach to the stream-based communication of Onion Routing. Similarly to mixes and Onion Routing, multiple proxies are used to route both the request and the response.

Assumptions

For our protocol to function efficiently a fully connected network of participating nodes is assumed. This network is henceforth referred to as the anonymity network. All nodes and their respective Internet gateways or firewalls are required to allow both inbound and outbound traffic.

Node discovery is not addressed in this paper. Using a centralised server for node discovery is one on option, a fully distributed architecture is presented elsewhere (Tillwick 2006).

Another assumption is that it is difficult, if not impossible, for an adversary to monitor all (or even the majority of nodes) in the anonymity network. Thus, the likelihood of a global observer is considered negligible. Such an observer would cripple the anonymity efforts of the protocol as traffic analysis on all nodes could expose the identity of the original requester. We believe this assumption is not unrealistic on condition that participants are distributed across multiple networks, administrative domains, ISPs and countries. Location diversity is thus required. This assumption is not unique as many anonymising technologies, including Crowds (Reiter and Rubin 1998) and (Freedman and Morris 2002), also make this assumption.

Communications tail

Our previously stated requirement that the Web response should not expose the identity of the Web surfer can be solved by using broadcasts. An adversary can not trace a response back to the original requester if all participants receive the same response.

However, broadcasts are resource intensive and thus scale poorly. Instead it is proposed that a (single) response is routed via a number of proxies, including the original requester, through to a dummy start node. An adversary can at most determine which nodes participate in routing the response but can not assume that the last node is the original requester. Note that the response could be still be encrypted, meaning that only the node with the appropriate key can decrypt it.

We shall refer to the path that the response maps as the communication channel. The original requester is referred to as the initiating node. The node which issues the Web request to the Web server is the final node. The node which is the last to receive the Web response is referred to as the dummy start node (as this is where the request would have originated from if conventional anonymity technologies had been used). The path from the initiating node to the dummy start node is the communications tail as it merely exists to prevent identity exposure. A graphical representation is depicted in figure 1.

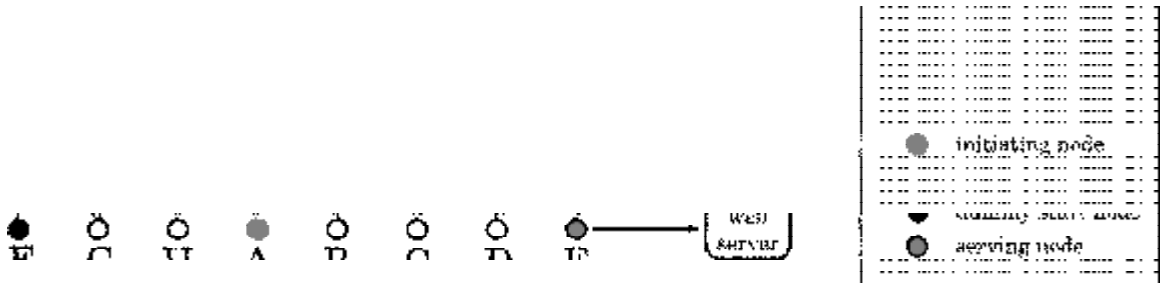

Figure 1: Communications tail

The channel discussed here bears some similarities to the channel constructed by Onion Routing. The proposed protocol could similarly benefit from having an Onion distribute symmetric keys between succeeding nodes. Onion Routing constructs a bi-directional virtual channel from the initiating node the the final node. The proposed protocol differs in that a uni-directional channel is required starting at the final node passing through to the dummy start node via the initiating node.

A disadvantage of Onion Routing is the fact that once a channel has been compromised, all subsequent communication is also compromised. In Onion Routing, the channel leads straight back to the initiating node thus revealing the identity of the original requester. This is not the case when a communications tail is used.

Whilst such a uni-directional channel allows a requester to anonymously receive Web content, it does not address how the request could be issued anonymously. This is discussed next.

Hiding the Web request

We propose embedding the Web request inside the Onion instead of transferring it via the communication channel, as is the case with the original Onion Routing protocol. This approach offers numerous benefits.

An Onion, with a comparatively small payload, lends itself well to a store-and-forward means of communication. This allows for mixing techniques to effectively hide and anonymise the request. Onions can be sliced and padded, batched and reordered and the use of dummy Onions can provide cover traffic.

One could ask why similar obfuscation techniques could not be applied to the anonymous communication channel. Why not segregate the Web response into separate chunks and treat them as messages instead of a continuous data-stream? Would this not enable one to hide the response data packets amongst other data packets containing the Web response for other requests? It should be noted that although the network layer segregates data in (TCP or UDP) packets, we are referring to the application layer because HTTP is an application layer protocol.

We argue that message-based communication is less feasible for large Web responses than for small Web requests because of the following reasons:

- It has been argued that mixing techniques are only useful if all mixing techniques have been successfully implemented (Tillwick 2006, p. 46). Message slicing and padding is only useful in combination with message batching and reordering. And inversely: message batching and reordering is only useful if fixed-size messages are used. Implementing only the one or the other would allow an adversary to perform

either timing or message length attacks. The same applies to cover traffic: omitting cover traffic allows an adversary to analyse the transferred data volume thus possibly leading to identity exposure. Partially implementing a mix technology thus offers limited benefits. The Freedom architecture (Boucher, Shostack and Goldberg 2000) is one example where a partially implemented mix strategy was dropped because the additional overhead and minimal benefit outweighed the advantages.

- In order to effectively hide a large Web response, a sufficient amount of cover traffic is required (Díaz and Preneel 2004a). Systems such as JAP (Berthold et al. 2000) originally planned the use of cover traffic. However, the current implementation does not send cover traffic because of the exorbitant additional bandwidth.

Mixing costs could be reduced significantly if only the smaller Web request is mixed. This approach could prove invaluable to Web browsing solutions, such as Freedom (Boucher et al. 2000), JAP (Berthold et al. 2000) and Tor (Dingledine et al. 2004), where the cost of mixing prompted for more efficient protocols with a reduced level of anonymity.

Channel construction

Our proposed solution uses an Onion to transfer the secret (the Web request) and establish a channel for the response. The channel construction is detailed next.

It is assumed each node has a suitable routing table from which it can select the network address and the respective public key of participating nodes. The routing table thus maps a network address A_n and a public key K_n to the respective node n.

We propose that upon receipt of a client Web request, the initiating node selects a random sequence of nodes $\{n_1, n_2, \mathbf{K}, n_j\}$. The sequence contains one dummy start node, one initiating node, one serving node and zero or more intermediate nodes. The sequence should consists of three or more nodes – a greater number increases latency but also lessens the chance of a collusion attack. The length of the channel is implementation-specific or alternatively can be user-specified according to the desired strength of anonymity versus the expected latency.

The first node of the sequence is the dummy start node whilst the last is the serving node. The initiating node n_I is inserted at any position except first or last. A channel is therefore given by:

$$S_{channel} = \{n_1, n_2, \mathbf{K}, n_I, \mathbf{K}, n_j\} \quad (1)$$

$$= \{n_1, \mathbf{K}, n_{I-1}\} \cup \{n_I\} \cup \{n_{I+1}, \mathbf{K}, n_j\} \quad (2)$$

$$= S_{tail} \cup \{n_I\} \cup S_{forward} \quad (3)$$

The initiating node prepares two Onions: one for the tail path and one for the forward path. The routing table is queried for the respective address A_{ni} and public key K_{ni} for all $n_i \in S_{tail} \cup S_{forward}$. The Onion is constructed by successively encoding the Web request (R) with layers of encryption corresponding to each node in the sequence of nodes. A layer is encoded using K_{ni} such that only n_i can decrypt it.

A symmetric encryption key K^t is generated by the initiating node which is used by the serving node to encrypt the response. A symmetric cipher should be used because these are generally computationally less expensive than asymmetric ciphers and are well suited for stream-based encryption. Note that the response is treated as a data stream.

The Onion for the forward path is given by:

$$(\mathrm{K}_{\mathrm{n\,I+1}}(\mathrm{K}_{\mathrm{n\,I+2}}(\mathbf{L}\ (\mathrm{K}_{\mathrm{n}\,j}(\mathrm{R},\ \mathrm{K}^{t}),\ \mathrm{A}_{\mathrm{n}\,j}),\ \mathbf{L}\ ),\ \mathrm{A}_{\mathrm{n\,I+2}}),\ \mathrm{A}_{\mathrm{n\,I+1}}) \qquad (4)$$

The iterative process of constructing an Onion is detailed in steps (5), (6) and (7). Step (5) shows an Onion with a single layer. The public key of the last (and serving) node is used to encrypt the Web request (Req) and the symmetric encryption key K^t .

$$K_{n_j}(R, K^t) \qquad (5)$$

$$K_{n_{j-1}}(K_{n_j}(R, K^t), A_{n_j}) \qquad (6)$$

$$\mathbb{M}$$

$$(K_{n_{I+1}}(K_{n_{I+2}}(\mathbf{L}\ (K_{n_j}(R, K^t), A_{n_j}),\ \mathbf{L}\ ,),\ A_{n_{I+2}}),\ A_{n_{I+1}}) \qquad (7)$$

The second step (6) shows an Onion with a second layer. The public key of the second-last node $A_{n_{j-1}}$ is used to encrypt the encoded message as well as the address of the succeeding node (node A_{n_j} in this case). When node $A_{n_{j-1}}$ unwraps its layer of the Onion it will retrieve a peeled Onion as well as the address of the next hop.

Note that node $A_{n_{j-1}}$ does not know that it is the second-last node. It should not be able to calculate its position from the size of the embedded Onion. It does not know how many layers still need to be unwrapped to get to the Web request. It does also not know the size of the Web request. Any inference based on the size of the Onion is therefore infeasible except for very small Onions.

Inference is only possible if the Onion only has one layer and if the Web request is close to the minimum Web request size. This allows an adversary to guess that it is unlikely for the Onion to have more than one layer. This can easily be rectified by having the initiating node expand the size of the Web request by inserting unneeded HTTP header values (by including fake user-agent or referrer (Fielding et al. 1999) details for example). Also note that even if a node can guess how many hops are still to come, it can not infer how many hops have already been completed.

The process of recursively wrapping the Onion and the address of the next node is repeated for all nodes on the forward path. A similar Onion is constructed for all nodes on the tail path (i.e for all $n \in S_{tail}$. The Onion for the tail path is given by:

$$(\mathrm{K}_{\mathrm{n\,I\text{-}1}}(\mathrm{K}_{\mathrm{n\,I-2}}(\mathbf{L}\ (\mathrm{K}_{\mathrm{n}\,2}(\mathrm{A}_{\mathrm{n}\,1}))\ \mathbf{L}\ ),\ \mathrm{A}_{\mathrm{n\,I-2}}),\ \mathrm{A}_{\mathrm{n\,I-1}}) \qquad (8)$$

Note that the Onion for the tail path does not have an embedded Web request. This is not necessary. The tail merely allows for a Web response to be returned to (and past) the initiating node without revealing its identity.

Once both Onions have been constructed, they are passed to the first node on the forward and tail path respectively. Each intermediate node can only remove its respective layer from the Onion. Doing so reveals the network address of, as well as the Onion for, the next hop.

The serving and the dummy start node remove the last layer. The serving node additionally retrieves the embedded Web request. It determines which Web server this HTTP GET request is destined for, issues the request, encrypts the response using the

received symmetric key and sends the response back through the recently established communication channel. The response then traverses the channel all the way back to the dummy start node.

The initiating node can be located at any position along the channel. The initiating node accepts the response, decrypts it and simultaneously forwards the encrypted response to the tail path.

Mixing

One question remains: why employ mixing techniques for the Web request (and hence all Onions) when the response already implicates those nodes participating in any particular communication channel?

An adversary should not be able to observe the sending of an Onion for the forward path followed or preceded by an Onion for the tail path. If this were possible, one would be able to infer that the observed node is an initiating node.

Mixing techniques allow all nodes to effectively hide their Onions amongst other real and dummy Onions. An adversary could thus only observe how the encrypted Web response is relayed via a number of nodes through a virtual communication channel, but would not be able to determine how the channel is constructed. Mixing prevents an observer from tracing an Onion as it is routed through the anonymity network.

In addition, it has been shown that identity exposure is no longer possible after the channel has been constructed. An adversary can at most determine which nodes participate in the channel but can not determine which of these nodes issued the original request (assuming that no adequate logging was performed).

The only time the initiating node is at risk is when the first node on the tail and the forward path collude. Both nodes relay the same Web response and would be able to infer, by comparing channel creation times, that the channel was initiated by the node situated between them. Assuming the first node on the forward path is not the serving node, the request and the response is still in encrypted form meaning that the requester's privacy is still preserved.

The proposed anonymity protocol fails when three nodes collude: the first node on the tail path, the first node on the forward path and the serving node. Compare this to Onion Routing which only requires a corrupt first and serving node. In addition, to compromise Onion Routing one would merely have to monitor traffic at the first and serving nodes. The proposed protocol requires the adversary to have internal knowledge of the channel creation process. Simply observing traffic is insufficient because mixing techniques prevent the correlation of received and sent Onions.

A common advantage of both Onion Routing and the proposed protocol is the fact that the initiating node constructs the Onions and can thus choose which nodes participate in the communications channel. Adequate node selection, using integrity or reliability statistics, can reduce the likelihood of selecting corrupt nodes.

Analysis

The proposed protocol offers further advantages over the current Onion Routing protocol.

Onion Routing (Syverson, Goldschlag and Reed 1998) uses a long-lived static channel for multiple Web requests. Static channels pose a threat to forward secrecy: once the original requester has been exposed all prior communication could be compromised (assuming that

adequate logging was done). The proposed protocol requires a new Onion and a new channel for each Web request. Only a single Web request would be compromised.

Embedding the Web request inside the Polar Onion means the serving node can immediately issue the Web request as soon as channel setup has been completed. The serving node does not have to wait for the request to traverse the recently established channel. This is the case with the original Onion Routing protocol.

Once a communication channel has been established, identity exposure is no longer possible by tracing the returned Web response. The response is sent past the initiating node to the dummy start node. An adversary can at most determine which nodes participate in a channel but can not infer which node originally issued the request.

Conclusion

It was noted how anonymity research on high latency, store-and-forward mediums, such as e-mail, have led to comparatively well-established and robust technologies; however, similar achievements have proven to be more difficult for interactive, low-latency communication mediums such as Web browsing.

The paper proposed an anonymous Web browsing protocol that harnesses some of the advantages of store-and-forward anonymity solutions whilst retaining some of the interactive properties of Web browsing. Moreover, it was proposed that Web requests and Web response be treated differently. Web requests are embedded inside Onions and are treated as store-and-forward messages. Mixing techniques can be employed increasing the offered level of anonymity.

The Web response is handled like a data stream. A communications tail allows the Web response to be passed through the initiating node to a dummy start node. An observer can thus not trace the response back to the original requester.

It has been shown that the protocol offers some advantages over current anonymity solutions. The author believes that certain existing anonymity solutions can benefit from the proposed approach. We additionally hope that the ideas presented in this paper contribute towards improving research on anonymity and privacy-enhancing solutions.

References

Aljifri, H. and Navarro, D. S. (2004), “Search engines and privacy”, *Computer and Security*, Vol. 23 No. 5, pp. 379–388.

Anderson, N. (2006), “The ethics of using AOL search data”, http://arstechnica.com/news.ars/post/20060823-7578.htm. Last accessed 15 September 2006.

Berners-Lee, T., Fielding, R. and Frystyk, H. (1996), “Hypertext transfer protocol – http/1.0”, RFC 1945.

Berthold, O., Federrath, H. and Köpsell, S. (2000), “Web MIXes: A system for anonymous and unobservable Internet access”, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, Springer-Verlag, LNCS2009, pp. 115–129.

Boucher, P., Shostack, A. and Goldberg, I. (2000), "Freedom systems 2.0 architecture", Whitepaper, Zero Knowledge Systems, Inc.

Chaum, D. L. (1981), "Untraceable electronic mail, return addresses, and digital pseudonyms", *Communications of the ACM*, Vol. 24 No. 2, pp. pp. 84–90.

Chung, W. and Paynter, J. (2002), "Privacy issues on the Internet", *Proceedings of the 35th Hawaii International Conference on System Sciences*, pp. 7–10.

Díaz, C. and Preneel, B. (2004a), "Reasoning about the anonymity provided by pool mixes that generate dummy traffic", Proceedings *of 6th Information Hiding Workshop (IH 2004)*, LNCS, Toronto.

Díaz, C. and Preneel, B. (2004b), "Taxonomy of mixes and dummy traffic", *Proceedings of I-NetSec04: 3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems*, Toulouse, France.

Dingledine, R. and Mathewson, N. (2005), "Anonymity loves company: Usability and the network effect", *Security and Usability*, O'Reilly Media.

Dingledine, R., Mathewson, N. and Syverson, P. (2004), "Tor: The second-generation onion router", *Proceedings of the 13th USENIX Security Symposium*.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and Berners-Lee, T. (1999), "Hypertext transfer protocol – http/1.1", RFC 2616.

Freedman, M. J. and Morris, R. (2002), "Tarzan: a peer-to-peer anonymizing network layer", CCS '02: *Proceedings of the 9th ACM conference on computer and communications security*, ACM Press, Washington, DC, USA, pp. 193–206.

Goldberg, I. (2002), "Privacy-enhancing technologies for the Internet, II: Five years later". Workshop on Privacy Enhancing Technologies 2002.

Goldschlag, D. M., Reed, M. G. and Syverson, P. F. (1996), "Hiding routing information", *Proceedings of the First International Workshop on Information Hiding, Springer-Verlag*, London, UK, pp. 137–150.

Helmers, S. (1997), "A brief history of anon.penet.fi - the legendary anonymous remailer", http://www.december.com/cmc/mag/1997/sep/helmers.html. Last accessed 29 January 2006.

Helsingius, J. (1996), "Johan Helsingius closes his Internet remailer", http://www.fitug.de/news/1997/penet.html. Last accessed 8 September 2005.

Mandujano, S. and Shields, C. (2000), "Confidentiality and anonymity analysis of on-line payment protocols", *Congreso del Día Internacinal de la Seguridad en Cómputo*, Mexico City, Mexico, pp. 45–52.

Parekh, S. (1996), "Prospects for remailers", *First Monday*, Vol. 1 No. 2.

Reagle, J. and Cranor, L. F. (1999), “The platform for privacy preferences”, *Communications of the ACM*, Vol. 42 No. 2, pp. 48–55.

Reiter, M. K. and Rubin, A. D. (1998), “Crowds: anonymity for Web transactions”, *ACM Transactions Information Systems Security*, Vol. 1 No. 1, pp. 66–92.

Reiter, M. K. and Rubin, A. D. (1999), “Anonymous Web transactions with crowds”, *Communications of the ACM*, Vol. 42 No. 2, pp. 32–48.

Sakalosky, M. (2002), “Doubleclick’s double edge”, http://www.clickz.com/experts/crm/analyze/data/article.php/1455141. Last accessed 3 April 2006.

Syverson, P. F., Goldschlag, D. M. and Reed, M. G. (1998), “Anonymous connections and onion routing”, *IEEE journal on Selected Areas in Communications*, Vol. 16 No. 4, pp. 482–494.

Tillwick, H. (2006), “Polar: Proxies collaborating to achieve anonymous communication”, Master’s thesis, University of Pretoria.

Tillwick, H. and Olivier, M. (2005), “Towards a framework for connection anonymity”, *Research for a changing world Proceedings of SAICSIT 2005*, J Bishop and DG Kourie (eds), White River, South Africa, pp. 113–122.