A Security Review of an Anonymous Peer-to-Peer File Transfer Protocol

Bryan Lipinski, Patrick MacAlpine
[lipinski,patmac]@rice.edu

Abstract

This paper examines the overall security of AP3 [2] (**A**nonymous **P**eer-to-**P**eer **P**rotocol), an anonymous peer-to-peer protocol developed at Rice University. In it we discuss the three primitives for anonymous communication provided by AP3 (anonymous message delivery, anonymous channels, and secure pseudonyms) and the vulnerabilities to attack present within them. We also describe specific types of attacks that could be used to compromise the network on which AP3 is running. Finally we look at some potential solutions and features that could be added to AP3, such as adding static graphs and a public key infrastructure, which would bolster the security of the protocol.

Introduction

Anonymity can be a very desirable trait in peer-to-peer networks. While some may argue that the only reason for wanting an anonymous P2P network is to hide illegal activities, anonymous networks have many helpful and useful functions. Anonymity can ensure DOS attacks cannot occur against a particular host. In standard P2P networks it is possible to correlate a node to an IP, and in doing so, an attacker can bring down any nodes through conventional attacks against the hosts. With an anonymous network, the attacker cannot know which node corresponds to a particular host, and as such cannot DOS a node using conventional attacks. Anonymity can protect whistle-blowers within an organization. In this way, the organization cannot know who the whistle-blower is and as such can not persecute that person for their actions. Finally, an anonymous P2P construct can allow applications in which anonymity is desired to be created easily; one such example is electronic voting. In such an application anonymity is needed to prevent the vote being tied to the voter.

There are many existing protocols built for anonymity. Most of these end up using a similar mechanism which is based upon random hops in the delivery of the message. This mechanism ensures that when a particular node A receives a message from a particular node B, node A cannot be certain that B is the originator of the message. This gives node B some degree of deniability as to whether it generated the packet or not. There are many different implementations of this ranging from Onion Routing, in which a particular set of routers know about each other and forward messages, to just random node forwarding.

AP3 is an anonymous protocol built on top of Pastry [6], a peer-to-peer network. AP3 offers 3 different services to the node: Anonymous Message Delivery, Anonymous Channels, and Secure Anonymous Pseudonyms.

Anonymous message delivery sends one particular message to a node through a random path in the network. Anonymous channels allow a persistent tie between a node and an ID while maintaining the nodes anonymity in the network. Secure anonymous pseudonyms allow a node to have a persistent id within the network which is authenticated through public key encryption.

Related Work

Previous work on anonymous protocols has primarily focused on two different approaches, Onion Routing [3] and the Crowds [4] approach. In Onion Routing messages are routed through Onion Routers which are a dedicated set of servers. A message is first encrypted into a layered data structure, known as an onion, which contains cryptographic information for every hop on the path of the message. It is then sent off to the first Onion Router in the path which is able to decrypt the first layer of encryption of the message. After decryption this router knows the address of the next router to forward the message on to. Eventually the last router in the path receives the message, decrypts the last layer of the message, and forwards the message on to its intended final destination. Each Onion Router implements a mixer based loosely on MorphMix [5] where it forwards on packets it received in a random order to thwart traffic analysis.

Tor [1] is a second generation protocol based upon the original version of Onion Routing and is an anonymous protocol for asynchronous loosely federated onion routers. Rather than using a multiply encrypted data structure, the onion, Tor uses a telescoping path-building design where session keys are negotiated between nodes at each subsequent hop in a circuit. Tor relies on directory servers which agree on the set of onion routers.

Crowds is the anonymous protocol that AP3 is based upon. Like AP3 nodes randomly decide to either forward packets on to other random nodes in the network or deliver them to their destinations. Unlike AP3, successive requests in Crowds follow the same path from one node to another as previous requests until the network signals for a periodic path reformation to occur which usually happens hourly. Crowds also uses a centralized server to provide admission control for the network.

AP3 Overview

In order to perform Anonymous Message Delivery a node first creates an anonymous request object containing a message as well as the address of the recipient of the message and forwards this object to a random node in the overlay chosen by drawing a random key. When this message is received by a node the node performs a weighted coin toss, and with a fixed probability (between 0.5 and 0.9) the message is forwarded on to another random node in the overlay. If the result of the coin toss dictates that the message is not to be forwarded on to another random node then it is sent directly to the intended recipient of the message. By forwarding the message along a random variable length path the identity of the sender is obscured from both the recipient of the message as well as other nodes in the network as nodes which receive a message are unable to tell if the node which sent them the message is the originator.

Because nodes who sent messages can not reveal their identity they must set up Anonymous Channels in order for recipients to respond to their messages. In order to set up an anonymous channel a node chooses a random id for the channel and then sends an anonymous message to the node closest to this id. As this message travels along a random path toward its destination node, each node along the path remembers the node it received the message from in a local table called the forwarding table. The nodes in the path then reconstruct the anonymous path back to the source node by forwarding any packets they receive addressed to the anonymous channel id on to the node in their forwarding tables. Entries in forwarding tables must be given an expiration time, however, because as soon as one of the nodes in the path goes down the anonymous channel is broken and the source node will no longer be able to receive messages through it. Once a channel expires a new and different anonymous channel is created by the source node. Because no node in the path knows if the node they are forwarding messages on to is the originator of the channel, the anonymity of the channel receiver is preserved.

In order to allow for encryption and authentication in message passing AP3 provides users with the ability to create Secure Anonymous Pseudonyms. Each node can create public and private key pairs and attach these to different pseudonyms. Once a node creates a pseudonym it can then establish an anonymous channel at a location given by running the pseudonym's public key through a secure hash function. Other users then encrypt their messages with the pseudonym's public key allowing them to securely send messages to the owner of the pseudonym.

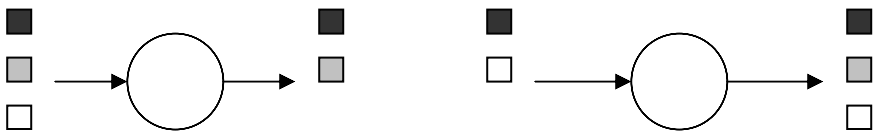
The Security of AP3

There are many problems with the overall design of AP3. These range from scalability issues to anonymity issues. Here are a few of the various problems we discovered upon examining the overall design of AP3.

Local Passive Attacks

In AP3 a local passive attack could expose the anonymity of a node on the local network. By simply studying what packets enter each node and what packets come out of each node, the attacker can deduce whether that node is the originator or recipient of the packet. Furthermore, knowing the contents of the packet, the attacker could link that node to a particular channel or anonymous pseudonym.

Since AP3 is unencrypted, all packet data segments between links are the same. Consequently, the attacker just has to compare the packet data of all packets leaving a node with those entering. If there is a discrepancy, i.e. a packet enters but doesn’t leave, than the attacker can deduce that the node was either the source or sink for that packet (See Figure 1). If the packet contains a channel ID or is a channel setup packet the attacker can then link that node to either accessing the channel or originating the channel.



Local Passive Attack to determine node is sink for white packets.

Local Passive Attack to determine node is source for gray packets.

Figure 1

It should be noted that even end-to-end encrypted systems may suffer from this attack. Even in these systems the packet data is the same for each link traversal. Furthermore, quantities such as channel ID must be readable to all nodes along the path so the encryption does not help with linking a node to a channel.

Another simple attack is to look at the contents of the packet. Since AP3 does not use encryption, if the application using AP3 is unaware of this fact, then the data transmitted may be enough to determine the node’s identity. Furthermore, if the attacker is aware of personal habits of nodes (person A always does such and such in the morning) then by looking at the packets for patterns, the attacker could correlate a packet’s content to a node.

Local Active Attacks

If an attacker is present on the same network as an Anonymous Pseudonym, the attacker may be able to use computer downtime to determine the corresponding node. Computers are shutdown regularly, or at least have some period of downtime. If a node with an Anonymous Pseudonym is down, then all packets for that pseudonym are lost. Thus an attacker can create a correlation between pseudonym queries and computer downtimes. The attacker can keep querying the pseudonym until the pseudonym does not reply, and then the attacker determines what nodes left the network in that time. Over a period of time the attacker should be able to determine the node corresponding to the anonymous pseudonym.

A local active attacker can also send packets towards nodes it suspects as being certain identities on the network. By sending a packet to their anonymous channel and watching for that packet to enter the network, the attacker can follow the packet to the destination using passive network watching.

Static Routing

AP3 makes use of static routes in order to create the anonymous channel and secure anonymous pseudonym. As shown above, this is to maintain the anonymity of the creator of the channel or pseudonym. However, by having a static route for any duration of time, the node allows the attacker to learn how packets are routed to it. The attacker can then use this knowledge to learn more about the node through simple traffic analysis.

One extremely simple attack is as follows. If a node is on an anonymous channel, i.e. it has an entry in its forwarding table, then it simply pings the node it forwards to, and also sends a message on the channel. It can then compare the two times and determine if that node is the originator of the channel. Even if they do not match, the node now has more information as to the potential number of hops away the originator may be. This added information could be useful in exploiting other attacks to determine the identity of a node.

Another attack is to simply burst a large number of packets towards the head of the channel. Since the route is static, all the packets will propagate through the same network path at the same time. Thus the attacker simply watches where the burst propagates to. Eventually the burst will hit the originator of the channel, and not continue on. In this way the identity of the originator of the channel is found.

Denial of Service

AP3 is susceptible to denial of service attacks, especially when there are large numbers of malicious nodes on the network. For a denial of service attack to occur a malicious node must be somewhere along the route of the packet. In a normal network the route length is proportional to the number of nodes in the network. In pastry the average route length is $\log_{2^{\wedge}b} n$, where b is a parameter of pastry, typically 4 [6]. In AP3 the average number of hops for anonymous message delivery is dependent on the forwarding probability, but is usually around 2 or 3. This multiplicatively increases the number of nodes which see the packet along its full anonymous route. Thus a fraction of the nodes needed to perform a DOS attack on a particular node in the full network are needed to perform a DOS on AP3, due to the larger number of nodes traversed in each transmission.

If there is a high probability a malicious node will be along the routing path for a packet then a malicious node can intercept the packet. The node can then do many things in order to DOS the system. It could simply discard the packet, and if every malicious node does the same, then the node will not be able to send any information. However, malicious nodes may want to be able to send packets, and so they might include a secret key in the data section of the packet in order to distinguish malicious node packets from others. Then when receiving a packet a malicious node checks for the key and if it is present it acts normally. In this way the malicious nodes can identify non-malicious node packets, in order to act differently.

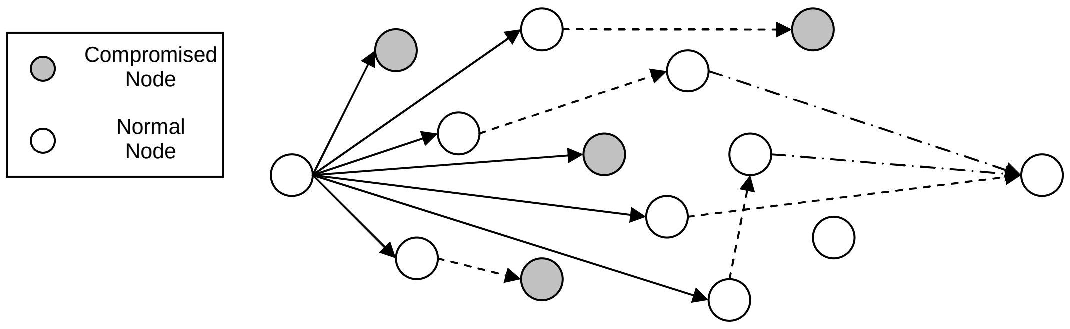
AP3 allows channels to have expiration timers. These timers create temporary static routes for anonymous channels. As shown above these can be very dangerous, but AP3 could be configured to have an expiration based on packet counts, such that after 1 or 2 packets the channel is invalidated. While this solves the static routing it allows another DOS attack. A malicious node could read the reply channel ID of a packet, and send enough packets to that channel to invalidate it while sending the original packet to the intended recipient. The recipient would then attempt to reply to the channel to find it invalidated. If the sending node reestablishes a channel with the same ID, it could still be invalidated by the attacker. Even if the sending node establishes a new channel and channel ID, it must send that information to the receiving node, in which case it could once again be overheard and invalidated. In this way an overall thrashing of the network on channels could occur, as packets bounce between invalidated channels.

Since AP3 is unencrypted the reply channel ID is sent in plain text. An intercepting malicious node could easily change this ID to an invalid one in order to create excess network traffic and prevent communication back to the sender. The malicious node could even set up its own channel and replace the reply channel ID in the original packet with its own. In this way the node can setup a man-in-the-middle attack.

AP3 does nothing to create indistinguishability between packets. Consequently, the malicious nodes could be selective in their DOS to either certain packet types, or even packets which fit a particular description. In this way if a particular group or application adds some signature to a packet before transmission over AP3, a malicious node could identify it, and perform one of the attacks presented above.

Global Node Compromising

In the initial paper on AP3, the authors conclude that even on a network with 20% malicious nodes a node's anonymity is not compromised. However, this is for a single packet, and most network interactions involve numerous packets. When numerous packets which can be correlated to a conversation are sent the anonymity of the sending node is easily determined by the 20% malicious node content. This is due to the fact that the probability the set of malicious nodes receive packets directly from the sender [.2] is much greater then the probability of a malicious node receiving a packet routed through another particular node after the first hop [(1 / nodes) * pf *.2]. Thus with a modest number of correlated packets sent, and communication between the malicious nodes, the malicious nodes can determine the identity of the sending node.



The compromised node set receives 2 packets from the source node, but at most one from any other single node. Consequently, the compromised nodes can say, with a high probability, that the source was indeed the sender. As more packets are sent the probability increases.

Figure 2

Node ID Proximity

AP3 runs on top of Pastry, which uses localization to assign node IDs and optimize routing. In normal p2p exchanges packets are routed towards node IDs that are "close" to the destination ID. It therefore becomes more likely to overhear packets on the local network that are addressed to nearby nodes. In such a case it would then be possible to use this information, combined with locality based knowledge, to determine a node's identification as outlined in the Local Attacks sections. Furthermore, local nodes would have a higher probability of being along a p2p routing path for a local node, in which case a high proportion of malicious local nodes can have the same effect on a local node as a high proportion of malicious global nodes. Thus the routing optimization in assigning to nearby nodes gives the local attacker more information, and more power then they would have normally.

AP3 counteracts this issue by only sending routing packets through the p2p network. When sending a packet on a single hop within AP3, the protocol sends a routing packet through the network to determine the final identity of the node. Upon finding the identity, the node then sends the anonymous packet directly to that node. In this way AP3 does not allow local nodes to see the transmitted packet directly.

However, AP3 still allows local nodes to see the routing requests. A malicious node with knowledge of a routing request is almost as dangerous as one that receives the packet directly. The malicious node could easily use the information from the request to eavesdrop on the local network and hear the packet anyways. Since the request must also include the originator for reply purposes the malicious node now knows the hop originator. If instead the malicious node just received a packet to forward, that packet could have anonymity properties similar to the crowds approach. Finally, since there is no encryption/authentication on the routing packets, a hostile local node could simply fake the reply, and act as a man-in-the-middle.

Suggested Solutions

Given the inherent structural problems in any anonymous peer-to-peer protocol, solutions must be found which address not only the problems, but also scalability. While there have been many more complicated attempts to construct more secure anonymous p2p networks, such as TOR, these protocols end up having scaling problems. AP3's simplicity allows it to scale very well, however this simplicity also causes many problems, as shown above. Consequently any solution to one of these problems should not affect AP3's scalability

Indistinguishability

One important improvement to AP3 would be to introduce the concept of packet indistinguishability. In this context indistinguishability is the notion that a node cannot distinguish between two uncorrelated or correlated packets. If AP3 only routed packets with this characteristic then local passive attacks would become harder, and packet correlation attacks would be impossible. If the attacker is unable to distinguish between any two packets, how can he correlate packets? Furthermore, if a node cannot distinguish between two packets, when a node receives 1 packet and outputs 2, how can a local passive attacker determine which output originated at the node? If AP3 packets could achieve this trait then a lot of the traffic analysis attacks would be invalidated.

The problem then becomes how to create indistinguishability in the packets. Packets can be identified through size and data analysis. Since AP3 runs at the application layer, any lower level packet fields can be ignored from this analysis. Therefore, in order to create this characteristic, the size of a packet as well as the data within it must be of a similar consistency between individual packets. This notion of indistinguishability must also be elaborated on in that it can occur in two different ways. For data indistinguishability a node can not tell the difference between two packets handed to it. For network indistinguishability two packets on the network cannot be correlated. Packet indistinguishability will be used in reference to an object which achieves both data and network indistinguishability.

In order to achieve size consistency between packets simply create an acceptable size range. If a packet is smaller then the range allows zero pad it until it fits somewhere within that limit. If a packet is too large then perform segmentation on it, and zero pad any smaller segments. In this way all packets have the same size. However, an attacker could easily see the zero-padding, and simply ignore it to determine the overall packet size. Thus, there must be some way in order to hide the zero padding. But, this is really the same problem as hiding the data of the packet.

In order to hide the packet data some form of encryption of that data should occur. The ideal encryption would be end-to-end as well as hop-to-hop. The encryption hop-to-hop would ensure that an attacker could not distinguish whether a packet leaving a node was the same packet that entered it. This encryption would be end-to-end on a transmission in order to hide all data content of a transmission to potentially malicious nodes. This end-to-end encryption would then have to use asymmetric cryptography, since a shared key between nodes must either be known by all nodes, thus rendering it useless, or only by the two communicating nodes which would undermine the anonymity of the protocol. Thus encrypting a packet with the destination’s public key, and computing a valid MAC with nonces, would create a secure anonymous transmission with data indistinguishability to the routing nodes. Furthermore, the use of public encryption on a hop-to-hop basis creates overall packet anonymity on the network which results in network anonymity.

The real problem with public key cryptography is designing the key distribution protocol. Furthermore, since anonymity is the purpose of the protocol, all requests for public keys should be secure from outside listeners, since they could compromise a node by giving away its future communication intentions. In using the above infrastructure, numerous key lookups would be required per sent message, which would lead to a performance decrease.

Currently AP3 uses end-to-end encryption with the Anonymous Pseudonym primitive. The authors assume an already setup asymmetric key infrastructure. If this is a valid assumption in this case, one could easily create weaker packet indistinguishability by only doing end-to-end encryption. Then you could limit use to the Anonymous Pseudonyms primitive, and not bother with the original Anonymous Message Delivery. In this case we have data indistinguishability, but not network indistinguishability. If the public key infrastructure assumption is valid only on a small scale, a few anonymous pseudonyms, then AP3 could greatly benefit from a public key infrastructure. This addition could provide at least some notion of indistinguishability.

If all packets routed through the network are zero-padded or truncated to a set length and then encrypted, the packets are then indistinguishable to any potential malicious nodes. By having this property the overall protocol can make itself more robust to traffic analysis techniques. Furthermore, since the encryption is end-to-end only a slight performance hit at the end nodes of a path will take place. However, the setup and maintenance of the public key infrastructure, especially at the scale of 10,000+ nodes, would definitely be a logistical nightmare, and could affect overall scalability.

Filler Traffic

Even the packet indistinguishability characteristic is not enough to prevent all types of traffic analysis techniques. A local passive attacker could still monitor packets coming into the network, and determine if they dead end at a node, or if they originate from a node. While the attacker could not read the data, it could look at lower level packet information and possibly correlate a channel to the node. Thus a node could attempt to hide the reception of packets and the transmission of packets with filler traffic. This would be “junk” protocol traffic which would be used to throw off this type of attack. When a node receives a packet, it sends a random packet to another node in the network. A node could also send out random packets periodically and thus hide a transmission with this filler traffic. However, the filler traffic does not scale well, and would congest the network. It is the opinion of the authors of this paper that while this may help with local passive attacks, the packet indistinguishability characteristic is enough to provide adequate anonymity while not hurting scalability.

Static Graph Routing

In order to deter traffic shape analysis techniques on the algorithm, some sort of packet diffusion mechanism is needed instead of the static paths. As shown above even with packet indistinguishability overall traffic shape analysis attacks using burst traffic can still compromise the anonymity of a node. If the packets of a burst were able to be diffused throughout the network, instead of flowing along a single link, the attacker would be unable to follow the burst, and the attack could be discouraged or nullified. Furthermore, packet diffusal can minimize the overall affect an attacker within the static route to the source node may have upon the overall anonymous channel. In a static path, a single attacker along the path would have access to all packets flowing through that channel. With a diffused network, the probability of there existing at least one "clean" path from attackers becomes very high.

The overall anonymous channel path must converge at two points: the channel creator, and the channel head. Consequently, packets or the path must have some notion of their destination, thereby eliminating a random approach to packet diffusion. Thus we are limited to using limited randomness in the path, or in coming up with a more diffuse static path.

One simple approach based on limited randomness would be to simply use anonymous message delivery in the routing through the anonymous channel. This would diffuse the traffic throughout the network. However, each packet sent with anonymous message delivery must contain the information for its destination. The attacker could retrieve this information through a variety of means (examining routing requests, controlling a node the packet is sent through, etc.), and consequently find the next node in the path. Finally, this approach could also cause an unreasonable time delay, as the path length has been increased by the randomness.

We instead suggest using a static directed graph for packet diffusion. The graph is formed with a mechanism similar to the static path forming. There are two main differences though: a single node is able to branch to multiple nodes instead of just one, and within the packet is information on another node in the graph to which a node can bridge. In this way we can form a directed graph from an end-point to a source-point. Packets are routed through the graph by each node randomly choosing which edge of the graph to send on. Our algorithm (See Figure 3) outlines our basic approach to building this graph. We define a branch to be when a node randomly expands to another node in the network. We define a bridge to be when a node expands to another node in the network.

Algorithm for a Node Creating a Static Graph

1. Node A, wishing to form a channel, determines an endpoint for the channel.
2. Node A then picks N random nodes to start the graph with. ($N \geq 2$)
3. Node A sends a packet to each of these nodes containing the ID of a known node, either itself, or one of the other randomly chosen start nodes.

Algorithm for a Node Receiving a Static Graph Packet

1. Node B picks a random number and comparesthis to a bridging probability. If the number is less then the probability the node creates a bridge path to that node.
2. Node B determines whether to forward the packet to more random nodes or to send it directly to the endpoint using p_f.
3. If Node B decides to forward the graph packet:
    a. The node then determines the number of nodes, N_f to forward the packet to using the branching probability.
    b. The node chooses N_f random nodes and sends a graph packet to them containing the graph end-point, and a known node (itself, it's parent, the bridge node, or other branch nodes)
4. The node then reverses all branch directions to form the graph.

Figure 3

This algorithm produces a DAG-like graph from the end-node to the source-node. However, it is not a true DAG in that there can by cycles of nodes of the same rank, which is a side-effect of the bridging and the inclusion of a node's future children in the known node list. This can be overcome by simply modifying the forwarding algorithm to exclude bridge edges after a packet comes in on an edge formed by a bridge. In this way we can avoid potential cycles in the forwarding, while maintaining a DAG-like structure to the graph.

The algorithm also allows for a node to control the overall anonymity of its static graph, in that it can control the original number of nodes the graph branches to. By branching to more nodes originally, the overall graph will become much wider without necessarily becoming deeper. This is because the overall graph path length will still be the same for each of the original branches. Thus, a user can determine their own level of anonymity through setting the time-out on the static graph, as well as the overall diffusion factor through setting the number of starting branch nodes.

Conclusions

AP3 was designed to be light-weight and scalable, however the simplicity involved in achieving these goals renders it susceptible to more sophisticated attacks. As this paper has shown AP3 is vulnerable to a wide variety of attacks whose results vary from denial of service to breaking anonymity guarantees of AP3. Consequently additional security features should be implemented in order to maintain the anonymity of the user as well as making the overall network robust against malicious attacks. We've proposed adding a public key infrastructure and a packet diffusion method to achieve a more desirable level of security.

There are many potential problems with implementing a public key infrastructure for the indistinguishability attribute. Implementing and maintaining a public key infrastructure at a large scale is an extremely difficult problem and as such drastically affects the scalability of the protocol. Furthermore, the overhead of encryption and decryption may cause undesirable time lag dependent on the overall message length and delivery path.

While our research may indicate that the Tor/Onion Routing approach has more merit in its overall approach and encryption, we believe AP3/Crowds approach still has much to offer as an anonymous peer-to-peer protocol. Tor, in its reliance on a central set of trusted Onion Routers, leaves it extremely vulnerable if that set of nodes is compromised. Furthermore the static routing of packets through a single path per transmission leaves it dependent on mixing to protect against traffic shape analysis attacks. Additionally, the overall encryption scheme and mixing require more overhead than desired.

AP3's anonymous channels as well as non-centralized routing provide a less node dependent network which would seem to make it more robust against a single attacker. While it may not be in the spirit of the original design of AP3, the inclusion of our proposed additions would increase the security while still maintaining the non-centralized focus which is the heart of the Crowds approach.

For more information on our research visit our website: http:www.owlnet.rice.edu/~lipinski/comp527/Project/.

References

[1] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second generation onion router. *Proceedings of the Thirteenth USENIX Security Symposium* (San Diego, CA), August. 2004.

[2] A. Mislove, G. Oberoi, A. Post, C. Reis, P. Druschel, and D. Wallach. AP3: Cooperative, decentralized anonymous communication. *11th ACM SIGOPS European Workshop* (Leuven, Belgium), September 2004.

[3] Onion Routing. http://www.onion-router.net/

[4] M. K. Reiter and A.D. Rubin. Anonymous Web transactions with Crowds. *Communications of the ACM*, 42(2):32-48, February 1999.

[5] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-peer based anonymous internet usage with collusion detection. *Proceedings of the Workshop on Privacy in the Electronic Society* (Washington, DC), November. 2002.

[6] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. *IFIP/ACM Middleware 2001* (Heidelberg, Germany), November 2001.