Using a reply path and handshaking protocol to allow for incentives with mix-net servers

Timothy Atkinson and Marius-Călin Silaghi
Florida Institute of Technology

November 30, 2004

Technical Report CS-2004-16

Abstract

Sending a message to a destination anonymously so that the destination cannot figure out who the originator was, and ensuring that the message reaches the destination untampered with, can be a difficult problem. Chaum's mix-net solves this problem but offers little motivation for a person to set up a mix-net server. This paper discusses a method that not only allows monetary payments as an incentive to create a public mix-net server, but also provides some fraud detection at the time a message is being sent. Also compared with other methods, the method described here can be considered reasonable from the users perspective, because if the message is not delivered the user does not lose any money.

Keyword: mix-nets, digital cash, anonymity.

1 Introduction

Citizens of many countries voting from overseas (authors of this paper included) pay postage to have their vote delivered. What if a method existed that would provide better guarantees of anonymity of the sender, allow a receipt of message being delivered, and for a price less than that of overseas postage? The algorithm we propose fits the description above.

Let us assume that a state sets up a voting system over the Internet so that if Bob, a citizen of the state, can get the vote to the state, the validity of the vote can be proven and it could be counted correctly. However, getting the vote so that no observer could figure out that Bob sent the message (and possibly not even the state could) can be a very difficult problem. If Bob was to send the message straight to the state, then all Mallory would have to do is listen at some point, and get a packet going to the state server from Bob and Mallory will learn that Bob voted. If Bob decides to use a third party server between him and the state, he would have to trust that this server is not controlled by Mallory (see Figure 1). One solution to this problem is, given the availability of a collection of

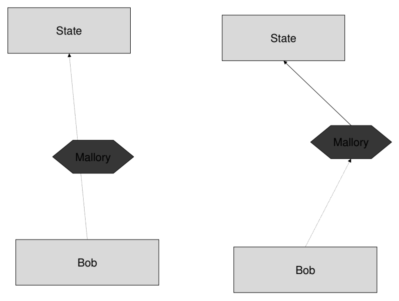

Figure 1: Attempting to send a message to State anonymously.

servers $\{S_1...S_M\}$, to set up a Chaumian mix-net with them [Cha81]. To send a payload m to A_N, Bob picks some random subset of the servers and some order on them: $P = \{A_1...A_N\}$. Let E_i be the public key of mix-net service provider A_i. Bob will then build the message $E_1(A_2, E_2(...(A_N, E_N(m))))$, forcing the message to visit each server in turn. Each server applies a decryption and sends the result to the destination it finds after decryption. This solves the problem nicely, if and only if we have a set of servers available and Bob believes that at least one does not cheat. However the problem with Chaumian mix-nets is that setting up a server has a negative immediate profit associated with it: network bandwidth costs money, processing power to handle the messages costs money (the server takes up space, the machine consumes power, etc.). If the state was to pay for the servers, than it is difficult to achieve maintenance of anonymity. If Bob pays the servers individually, then the security of the method goes back to being equivalent to Bob contacting a single server to forward his payload to the state. Finally, if a company implemented the mix-net internally (supposing you trust the company actually did implement the mix-net), they can still only be considered as a single server since they can check all the logs of the machine and correlate the incoming message to the outgoing message.

To fix this problem, a method was proposed in [XNJS04] attempting to use the market as an incentive to "enhance anonymity". If you modified the message being sent to include digital coins, then these coins can be distributed as the message passes from server to server. Because the servers are now being paid, individuals/companies will create servers to take advantage of this market. Based on competition between mix-net servers, they will be forced to offer a fair price. [XNJS04] claims that if a server has poor performance, eventually they will be discovered and will lose reputation. Lose of reputation would cut into their profits, making the system self-regulatory. To allow payments to the servers along the path, the proposed structure modification of the message is as follows: $E_1(coin_1, A_2, E_2(coin_2, A_3, ...E_N(m)...))$, where $coin_i$ is the digital coin to be paid to A_i. This method solves the problem of the negative cost associated with setting up a server, and can even make it profitable; however, this method does not provide any methods to prevent and detect fraud on a single message. Worse, in the case of a server which does not forward messages, the participant committing fraud still gets payed.

We propose a method to fix this inconvenience by slightly modifying the message being sent, obtaining:

$$\{E_1(E_2(E_1(coin_1), E_3(E_2(coin_2)), ...\\ E_{N-1}(E_N(E_{N-1}(coin_{N-1}), m))...))\}$$

(where every $coin_i$ from the previous message is replaced with a $E_{i-1}(coin_{i-1})$).

The result of this extra encryption on the coins, is that for any server A_i to receive his payment, he must first forward the message to the next server A_{i+1}, who then must send $E_i(coin_i)$ back to S_i's server. This has the net effect of increasing the chance that the message is delivered, and at minimum prevents any single server from being able to commit fraud and getting away with the money.

As mentioned above this method has problems, too. While it improves the identification of fraudulent servers, a remaining problem is that the end-user may have already lost money (cashed legally by previous servers) and still not have gotten the payload m through. The solution we propose is to hold payment until the message is fully delivered. In order to pull this off, we revert back to Chaum's original structure except instead of only supplying the message in the last encryption, we also concatenate m with the money in the following pattern:

$$\{E_N(E_{N-1}(E_N(m), E_{N-2}(E_{N-1}(coin_{N-1}), ...\\ E_2(coin_3, E_1(E_2(coin_2), dummy))...)))\}$$

similar to the structure of the previous payment method, but the order is reversed, and the destination has to confirm the message reception.

Because payments and confirmation need to return along the same path on which the message is sent, each message appends the address of the server from which it received the message it and encrypts it with a secret key that only it can decrypt. This forces the message to return along the same path the message was sent, does not compromise security, and can be done quicker and with smaller message size than by generating the return path at the message source. I also enables the usage of a symmetric encryption algorithm, which is faster.

The achievement of the system is the guarantee that end-users spend money only if the payload was delivered at the final destination (even if they do not go to court), making the application more reasonable. Also since the message is passed from server A_i to A_{i+1}, then A_i knows where it is getting its money, and expects to get paid. The end result is a system which can be safely audited, fraudulent nodes can be found or at least narrowed down given the cooperation of the chain. This increases the accuracy with which fraudulent nodes are found and removed. Moreover, because it is possible to construct the message so that there is no difference between Bob and some A_i along the path P, Bob can get a confirmation that his message was successfully delivered while his anonymity is increased.

Outline. In Section 2, we present a more detailed examination of the background. In Section 3, we go more into detail of the algorithms detailed in this paper. Finally in Section 4, we conclude the paper.

2 Background

There have been many attempts to solve this problem or similar problems before. Chaum's mix-net method [Cha81] is the first one. It solves the problem, given a set of available mix-net servers, but offers no incentive to create any such servers. Later, [GRS96] presented a better known implementation which was based on Chaum's mix-net but using a proxy to hide this from the application layer, and they showed how to hide the destination and source by making them look like servers. Also [Sim96] proposes a mix-net for digital cash. An application of mix-nets to voting was discussed in [Mer83]. Recently, independently from us, [XNJS04] proposed using digital currency for paying servers in a mix-net. In [JJR02], it is shown how to improve the reliability of the servers by probabilistic checking. Besides these papers there are other important works on improving mix-nets. For example, [RR98, FM02] study how to avoid timing analysis by generating false traffic.

2.1 Chaum's mix-net method

Chaum's method of sending a message anonymously proposed in [CE86] is, given a set of available servers $\{S_1...S_N\}$, to let a sender choose a subset of them for a path $P = \{A_1...A_N\}$. Then the source will encrypt the message with the public keys from E_N to E_1 forcing the message to go along the specified path. Also, since any sever along the path only knows who sent the message to it and its destination, the only way the destination could discover the source is for every server along P to cooperate and reveal the piece of information they have.

As mentioned above, Chaum's mix-net falls short because it does not offer any incentive to set up a public server. If you take into consideration that setting up a server on the Internet costs money, setting up a server for a Chaumian mix-net has negative immediate incentive.

2.2 Digital cash in mix-nets

Recently in a group of researchers [XNJS04] independently from us proposed the use of digital cash as an incentive for service providers. They argue that by providing money to the servers along the path, then reputation will force them to play fairly since bad systems will be filtered out of the system. However, in the case of a fraudulent server, the source loses his money up to and including the fraudulent server with no guarantee of service. Further more, localizing fraudulent servers is difficult at best when a collection of messages are sent, and impossible on a single message.

3 Evolution between problems arising and solutions to those problems

Our initial problem is how to enable Bob to send a message to Alice, such that nobody knows that Bob sent the message.

3.1 The initial problem

Let us start with an initial system based on Chaum's work [Cha81]. In such a system, an end user A_1 (message sender to a destination A_N) decides to ask help from a subset $P = \{A_2, ..., A_{N-1}\}$ of the available service providers. Each server A_i makes available a public key, E_i, for some asymmetric cryptosystem. As end user of the system, A_1, may pick the order it wishes on P. To send payload m to A_N, A_1 prepares and sends the message $E_2(A_3, E_3(...(A_{N-1}, E_{N-1}(A_N, m))...))$ to the first server in the chosen list, A_2. A_2 decrypts the message, and forwards it to the next server in the list, A_3, who then decrypts the message and forwards it to the next server in the list, A_4, who again decrypts the message and again forwards the message, etc. When the packet reaches A_N, A_N retrieves the payload m. This algorithm solves the problem of anonymity of the source at the destination so long as A_1 believes that at least one of the servers in P is reliable. If A_N would like to learn the identity of the source, it needs to trace backward the path of the message. A_N must go through every server to figure out who gave them the message. If any one of the servers refuses to cooperate, then it is impossible for the destination to find the source.

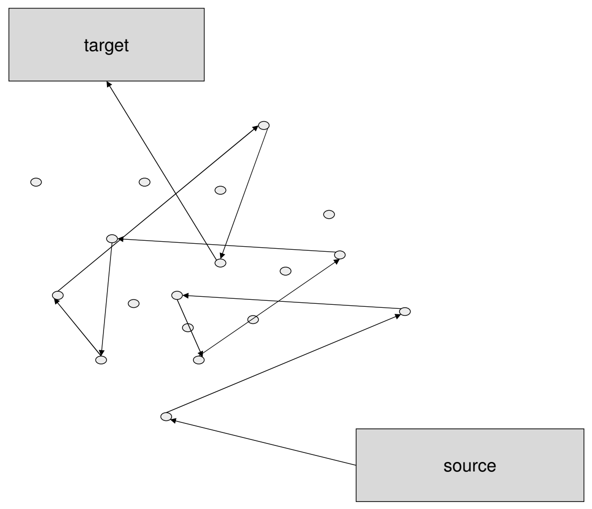

Figure 2: A message being sent among a bunch of mix-net service providers.

3.2 The introduction of monetary incentives to create new servers (the market mix-net)

Let us assume that we wanted to set up a system on the Internet that anybody could use, and that would allow a person to send an anonymous message to some destination. The problem with Chaum's system as a generic Internet application, is that it has a negative incentive for a person/organization to set up a service providing server. Sending and receiving messages ties up network bandwidth, and processing the messages ties up computation power, which leads to a negative immediate profit for servers. Also, if a company did offer a collection of servers to act as service providers, from the perspective of the user the company should still only be treated as a single service provider. After all, the company controls the logs on all its internal computers (supposing you trust that it really does have multiple servers) and if it wanted to tell the source to somebody it could go through all of its internal servers until it found the source and let the adversary know who the source was.

The first thought that comes to mind is to offer a payment for services being offered[1]. There exist techniques that can provide anonymous payments and yet provide the security required for digital cash [CPS96, Bra95, BGK95]. The structure for sending messages changes only slightly, becoming:

$$E_2(coin_2, A_3, E_3(...(coin_{N-2}, A_{N-1}, E_{N-1}(coin_{N-1}, A_N, m))).$$

Each server A_i earns the e-cash $coin_i$ from the message that it processes. This now stimulates people to declare themselves *mix-net service providers* and to

[1]This was also discovered independently from us in [XNJS04].

accept to receive messages, but does nothing to encourage them to really send it down to the next service provider.

3.3 Utilizing the message structure to encourage nodes to forward their data (the handshake mix-net)

In a network, when a server A_i receives a message M from server A_{i-1}, it knows that A_{i-1} sent the message. Because A_i is not the final destination, A_i also knows where the A_{i+1} is located. We can use this to provide an incentive for A_i to forward the data to A_{i+1} by moving A_i's coin to the private message for server A_{i+1}, encrypting it with the with A_i's public key. Therefore we replace the coin with: $E_{i+1}(E_i(coin_{i-1}))$. This forces A_i to send the message to A_{i+1}, since A_i expects its payment from A_{i+1}.

One further optional improvement can be made to the system where every message being exchanged between servers has a digital signature made. This can be considered to slightly weaken the anonymity of the sender, because the ability for a server to deny the sending of message becomes impossible. However, given that Mallory needs to still get the cooperation of all the servers in order to find the source, it doesn't weaken the security of the problem by much.

3.4 Another improvement (the reply-pay mix-net)

The system proposed above can improve localization of fraud by the source of the message; however the sender loses money that are difficult to retrieve. Since we are assuming people are adding servers in order to make money, we can use this to our advantage to correct this problem. Rather than providing the digital coins at the time the message passes through the server, we instead move the coins to the end so that the message has to be delivered before any server will get its cash. To ensure that the participating servers have incentive to forward payment back down the list, the method of doubly encrypting coins is going to be applied here, the only difference is encryption is set up as if the destination sent the message rather than the source. The structure for a message being sent between two servers, A_{i-1} and A_i, has two elements, looking like:

$$\{E_i(A_{i+1}, E_{i+1}(...(A_{N-1}, E_{N-1}(A_N, E_N(coins, m)))...)),$$
$$E_i^1(A_{i-1}, E_{i-1}^1(...(A_2(E_2^1(A_1, dummy\_path))))))\};$$

and *coins* has the structure:

$$\{E_{N-1}(E_{N-2}(E_{N-1}(coin_{N-1})), E_{N-2}(E_{N-3}(E_{N-2}(coin_{N-2})), .$$
$$E_3(E_2(E_3(coin_3)), E_2(E_1(coin_2, dummy\_coins))).))))\}.$$

The E^1 represents the fact that the return addresses do not need to be encrypted with the same algorithm/key as the one used to compose the original message. Only the server encrypting the return addresses needs to be able to decrypt it. Because of this, a symmetric encryption algorithm can be used in place of the slow asymmetric encryption algorithm.

Note that in the structure presented above we assume that the destination server A_N operates reliably for free, as expected for voting servers. However, in situations where the destination is a payed service, it can be stimulated to send the digital coins with payments backwards by presenting him his payment, $coin_N$, wrapped as with the other mix-net service providers, namely in $E_{N-1}(E_N(coin_N, m))$. The server A_{N-1} will remove his encryption only when the destination sends the other coins.

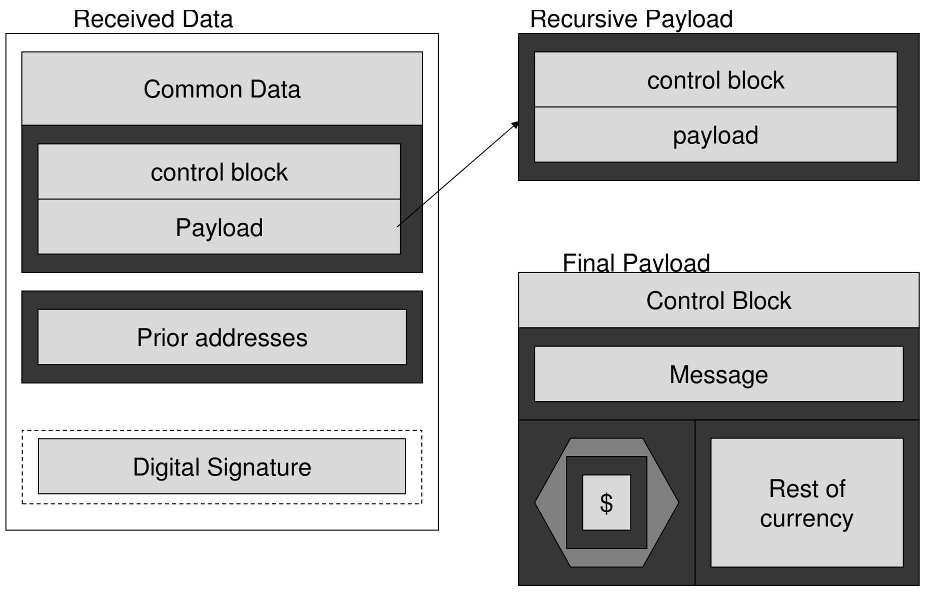

Figure 3: A pictorial view of the data structure used for a submitted message.

Figure 3 shows a particular implementation of the above equations. Because the servers may come from all over the Internet and as such have no affiliation with each other, the *Common Data* section offers a means to have directories of servers and just index into the directory to reduce message size. The *control block* contains the next hop, which may be an entry into the directory in the common block, or may be an IP address not in the common block. It also specifies whether the current node is the actual destination. The next field is the payload. As described above, the payload is a recursive definition of a control block and payload, until the final payload which also has the currency for the entire set of participants. These are followed by the return block. The *prior addresses* record the path took so far by the message, recursively encrypted such that it can be used for returning the digital cash payment or detecting fraud from sender.

3.6 Advantages of the proposed method

This method has the following advantages. First, if for some server's fault the message does not reach the destination, the sender does not loose any money, and can resend the package. Then, the previous servers know who they have sent the message to and so know to expect payment from that server. Of course, that server may not have received the message, but that means it also didn't get payed, and now pressure will be created to find out where their money is. Colluding servers could say they never received the message, but at least one of the colluders will be isolated as he conflicts with a non-colluding neighbor. This allows to build bad reputation desired in [XNJS04]. Third, because the destination has to send the currency back down the line anyhow, it becomes very easy to send an acknowledgement to the sender. Finally, because of the structure of the message, the sender could pretend to be just another node in the list and to have received the message from some other server [GRS96].

4 Conclusion

We have presented a method that not only uses payment as a means of enticing people to create servers to participate in a mix-net, but also provide a means of validating the fact (or lack thereof) of the system working. End-users do not lose money if they do not get the service, improving user satisfaction. The method improves the ability of a server to prove that it has operated correctly in the case where fraud is suspected. Or, should a server find that they have fraudulent data, they can show that the data they received is fraudulent. A reputation system becomes therefore possible, leading to a usable system. In conclusion we believe that we have finally reached the situation where larger deployments of mix-nets for ensuring anonymity are probable to become reality.

References

[BGK95] Ernest F. Brickell, Peter Gemmell, and David W. Kravitz. Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In *SODA'95*, 1995.

[Bra95] Stefan Brands. Off-line electronic cash based on secret-key certificates. In *LATIN'95*, pages 131–166, 1995.

[CE86] D. Chaum and J. Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In *CRYPTO*, pages 118–167, 1986.

[Cha81] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Com. of ACM*, 24(2):84–88, 1981.

[CPS96] Jan Camenisch, Jean-Marc Piveteau, and Markus Stadler. An efficient fair payment system. In *ACM'96*, pages 88–94, 1996.

[FM02] Freedman and Morris. Tarzan: a peer-to-peer anonymizing network layer. In *ACM CCS'02*, 2002.

[GRS96] D. Goldschlag, M. Reed, and P. Syverson. Hiding routing information. In *Information Hiding*, number 1174 in LNCS, pages 137–150, 1996.

[JJR02] M. Jakobsson, A. Juels, and R. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX*, 2002.

[Mer83] M. Merritt. *Cryptographic Protocols*. PhD thesis, Georgia Inst. of Tech., Feb 1983.

[RR98] Reiter and Rubin. Crowds: Anonymity for web transactions. *ACM Trans. on Information and System Security*, 1(1):66–92, 1998.

[Sim96] D.R. Simon. Anonymous communication and anonymous cash. In *CRYPTO*, number 1109 in LNCS, pages 61–73, 1996.

[XNJS04] S. Xu, W. Nelson Jr., and R. Sandhu. Enhancing anonymity via market competition. information assurance and security. In *IEEE ITCC'04*, 2004.