Practical Anonymity for the Masses with MorphMix

Marc Rennhard and Bernhard Plattner

Swiss Federal Institute of Technology, Zurich, Switzerland
Computer Engineering and Networks Laboratory
rennhard|plattner@tik.ee.ethz.ch

Abstract. MorphMix is a peer-to-peer circuit-based mix network to provide practical anonymous low-latency Internet access for millions of users. The basic ideas of MorphMix have been published before; this paper focuses on solving open problems and giving an analysis of the resistance to attacks and the performance it offers assuming realistic scenarios with very many users. We demonstrate that MorphMix scales very well and can support as many nodes as there are public IP addresses. In addition, we show that MorphMix is indeed practical because it provides good resistance from long-term profiling and offers acceptable performance despite the heterogeneity of the nodes and the fact that nodes can join or leave the system at any time.

Keywords: anonymity, peer-to-peer mix networks, collusion detection

1 Introduction

MorphMix is a peer-to-peer *circuit-based mix network* [6] to enable anonymous Internet usage for low-latency applications such as web browsing. Unlike traditional circuit-based mix systems such as Onion Routing [14], the Freedom Network [5], JAP[1], and the Anonymity Network [19], MorphMix does not consist of a relatively small set of dedicated mixes that serve many users. Rather, every MorphMix user is also a mix at the same time.

The main goal of MorphMix is to provide practical anonymous Internet access for the masses, i.e. for millions of users. Traditional mix systems – operated commercially or not – may not be the best option to fulfil this task [18]: the experience with the commercial Freedom network has shown it is difficult to offer such a service in a profitable way and systems with mixes run by volunteers may fail to acquire enough mixes for cost reasons and due to potential political and legal pressure. In general, having many mixes operated by independent institutions or persons located in several different geographical and jurisdictional areas is good to increase the resistance to certain attacks because (1) it is difficult for even a well-funded adversary to run a significant portion of all mixes himself and (2) legal attacks are much harder to carry out.

[1] http://anon.inf.tu-dresden.de

The basic ideas behind MorphMix have been published before [17]. In this paper, we answer the questions that were left open (mainly about the threat model, peer discovery, and scalability) and present a full analysis to demonstrate MorphMix is indeed practical system to provide anonymity for the masses. With practical, we mean that it (1) offers acceptable performance despite the heterogeneity of the nodes and the fact that nodes can join or leave the system at any time, and (2) provides good resistance to a realistic adversary. Especially the first property is very important because in anonymity, usability is an essential requirement: hardly anybody will use a system that offers poor performance no matter how well it protects from attacks. But without any users, there is no anonymity at all [2, 1].

In the next Section, we provide a brief overview of MorphMix. Section 3 states the threat model and Section 4 briefly repeats the collusion detection mechanism. Section 5 discusses the peer discovery mechanism and Section 6 discusses why MorphMix scales very well. In Section 7, we analyze the collusion detection mechanism. The performance MorphMix users may expect is evaluated in Section 8 before we compare MorphMix with similar systems in Section 9. Finally, we conclude our work in Section 10. Due to the limited space, we can only present the most important results of our analyses. For a more thorough discussion, refer to the technical report [16].

2 MorphMix Overview

MorphMix is made up of an open-ended set of nodes. A node i is identified by its IP address ip_i and has an RSA key-pair generated locally when a node is started for the first time, consisting of a secret (or private) key SK_i and a public key PK_i. A node that is part of MorphMix is *connected* to other MorphMix nodes, which are its *neighbors*. Two nodes that are connected share a symmetric key, which is exchanged using their public keys.

Basically, MorphMix is a circuit-based mix network and to access Internet hosts anonymously, a node establishes a circuit, which we name *anonymous tunnel*, via some other nodes. The first node in a tunnel is the *initiator*, the last node the *final node*, and the nodes in between are *intermediate nodes*. The total number of nodes in a tunnel is the *tunnel length*. Sending data along a tunnel works similar as in other circuit-based systems such as Onion Routing [14] and makes use of *layered encryption* and *fixed-length cells*. Anonymous tunnels can be used to contact several hosts subsequently or in parallel. To do so, *anonymous connections* that are only visible to the initiator and final node are transported within anonymous tunnels. It should be noted that setting up tunnels is a background process in the sense that when a host should be contacted anonymously, there are always a few tunnels ready to be used (see Section 8).

One key feature of MorphMix is that when setting up a tunnel, each node along the tunnel selects its successor node. This has the advantage that a node only has to manage its local environment consisting of its current neighbors, which is nearly independent of the system size. Neighbors can directly commu-

nicate with each other and exchange control information to learn which nodes have spare resources to accept further anonymous tunnels.

We only give a sketch of the protocol to set up a tunnel because it has already been provided and analyzed in [17] and has only been slightly adapted [16]. The initiator a picks the first intermediate node b among its current neighbors and establish a symmetric key with it that is used for the layered encryption. Then, a tells b to append a node. To prevent b from easily picking any next hop it likes, b must offer a *selection* of possible next hops among its neighbors to a, which selects one of them. This selection plays an important role in the collusion detection mechanism (see Section 4). Assuming a has picked node c, a and c establish a symmetric key via b to be used for the layered encryption. Since a learns c's public key from b as part of the selection, a cannot simply choose a key and encrypt it with c's public key, because this could easily be exploited by b by carrying out a man-in-the middle attack on the layer of encryption between a and c. To prevent this attack, a picks a *witness* w from the nodes it currently knows (see Section 5) and encrypts the key first for c and then for w. The resulting data are sent to b, which sends them to w. Node w decrypts the data and forwards them to c, which decrypts the data again to extract the symmetric key. Appending additional nodes works in exactly the same way until a decides the tunnel is long enough.

3 Threat Model

We assume the adversary wants to link communication partners in as many cases as possible to accumulate and possibly sell dossiers about Internet users. Consequently, the goal of MorphMix is to provide very good protection from long-term profiling instead of guaranteeing the anonymity of every single transaction. In fact, considering the open and asynchronous nature of the Internet and powerful attacks on mix systems [2–4, 11, 13, 21, 22], operating such a system such that it is both practical and resistant to powerful adversaries is a very challenging problem. In particular, if a user is suspected to communicate anonymously with a host, then a targeted attack by monitoring both the data sent and received by the user's computer and the host should make it possible to link the two communication partners in most cases by means of traffic confirmation. Cover traffic may protect from such attacks, but especially in mix networks for low-latency applications, they tend to introduce vast amounts of data overhead. In general, the benefit of dummy traffic is still not really understood and therefore, MorphMix does not employ any such mechanisms at this time. However, since MorphMix is essentially a mix network, we state that if efficient cover traffic mechanisms that significantly increase the protection from attacks low will be ever developed, they should be easily applicable to MorphMix.

We say an anonymous tunnel is *malicious* or *compromised* if an adversary manages to link the initiator and the host(s) that are contacted through this tunnel. Since MorphMix does not employ any cover traffic, we assume that a tunnel is compromised if (1) an external observer eavesdrops on both the link

between initiator and first intermediate node and on the route between final node and host(s), or (2) an adversary operating some nodes himself controls both the first intermediate and the final node. Note that in practice, this is not always easy because the chances of the adversary depend on the amount of data exchanged between initiator and host. In addition, one property of MorphMix is *plausible deniability*, i.e. the first intermediate node does not know if the previous node in the tunnel is the initiator or if that node is merely relaying the data for yet another node. However, by analyzing the timing patterns of cells exchanged between initiator and first intermediate nodes and because of the fact that the tunnel length will be a reasonably small number in practice, the first intermediate node should often be able to guess its position in the tunnel. Nevertheless, our assumption about compromised tunnels is a worst case assumption because anything else is difficult to quantify.

An adversary that observes a fraction of 0.1 of all MorphMix traffic succeeds in compromising a fraction of $(0.1)^2 = 0.01$ of all tunnels on average. While large backbone ISPs may indeed be capable of observing so many data, we state the threat from external observers is quite small. Increasing the protection from this adversary depends on the development of efficient cover traffic mechanisms. On the other hand, due to the openness of the system, an internal active attacker controlling a subset of all nodes and compromising a significant fraction of all tunnels is a real threat. Consequently, we must assume there are *honest* nodes, which are nodes that do not try to break the anonymity of other users and there are *malicious* nodes, which may collude with other malicious nodes to break the anonymity of honest users. We have analyzed many different attack strategies [16] for an internal attacker that aims at compromising as many tunnels as possible. Since every node in a tunnel selects its successor node, we have come to the conclusion that the most effective attack to control both the first intermediate and the final node in a tunnel is the one where malicious nodes offer many or only other malicious nodes in their selections during the tunnel setup (see Section 2).

To defend against the internal attacker, MorphMix employs a *collusion detection mechanism* (see Section 4), which exploits the fact that usually, only contiguous ranges of IP addresses are under a single administrative control. We say that all IP addresses with the same 16-bit prefix belong to the same */16 subnet*.[2] Leaving out reserved and multicast addresses, there are exactly 56559 public /16 subnets in the Internet. An adversary owning an entire class B network can still run 65533 MorphMix nodes, but from the point of view of the collusion detection mechanism, they all belong into the same /16 subnet. Controlling nodes in many different /16 subnets is much more difficult than in a single subnet. Even an adversary owning an entire class A network has easy access to only 256 different /16 subnets. Consequently, we assume the adversary can operate nodes only in a small subset of all /16 subnets. It is difficult to specify an upper limit, but we do not believe it is realistic a single adversary will ever be able to run nodes in significantly more than 1000 /16 subnets because even

[2] We have developed a similar concept to support IPv6 [16]

the largest ISPs do not control addresses in so many /16 subnets. The adversary could also try to run nodes in subnets he does not possess, either by himself or by private persons. Again, running nodes in much more than 1000 subnets is very difficult, in particular if the adversary wants to avoid that his activities become public.

4 Collusion Detection Mechanism

The collusion detection mechanism bases on the assumption that the most effective attack is that malicious nodes offer many or only malicious nodes in their selections, i.e. they offer nodes from a relatively small spectrum of all /16 subnets. Honest nodes, on the other hand, choose their neighbors and therefore also the nodes in their selections more or less randomly from all /16 subnets that contain nodes (see Section 5). We name the selections offered by honest nodes *honest selections* and the selections from malicious nodes *malicious selections*. Each node maintains a *extended selections list* L_{ES} that contains the k_{ES} (see Section 6) most recently received *extended selections*. The extended selection is the combination of the 16-bit prefixes of the IP addresses in a selection and of the node that offered the selection. For each new extended selection, the initiator computes a *correlation* by comparing it with all other extended selections in L_{ES}. We do not describe this algorithm here because this has already been done in detail in the original paper [17] and only repeat the main result that this correlation is in general relatively big if the new extended selection contains many or only colluding nodes and relatively small otherwise.

A node remembers the correlations it has computed over time and represents them as a *correlation distribution*. This correlation distribution is used by a node to determines a *correlation limit*, which has the property that if the correlation of a new extended selection is smaller than this limit, then the node that offered the corresponding selection is honest with a high probability. During the setup of a tunnel, the initiator gets an extended selection from each intermediate node. If at least one yields a correlation larger than the correlation limit, the tunnel is considered as *malicious* and is not used. Otherwise, it is considered as *good* and can be used to contact hosts anonymously.

5 Peer Discovery and Selecting Nodes

For the collusion mechanism to work correctly, honest nodes must pick the nodes they offer in their selections as randomly as possible from the set of all /16 subnets that contain at least one node. To do so, honest nodes must (1) frequently change their neighbors and (2) new neighbors must be selected as randomly as possible, which is exactly what the peer discovery mechanism should support.

Once a node is participating in MorphMix and starts setting up anonymous tunnels, it learns about a variety of other nodes through the selections it receives. It remembers these nodes and arranges them in a *most recently seen subnets list* L_S. There is at most one entry in the list per /16 subnet and each entry

contains the corresponding 16-bit prefix and a *most recently seen nodes list* $L_{N,S}$, which contains information about nodes in this subnet that have been received in selections. An entry in $L_{N,S}$ contains the IP address, port, public key, and node level (see Section 7) of the corresponding node. When the initiator learns about a new node, it moves the corresponding entry in L_S to the first position of the list, or inserts an entry at the first position if the /16 subnet has not yet been in the list. Then, the information about the new node is inserted at the first position of the corresponding $L_{N,S}$. If $L_{N,S}$ already contains information about the node, the old entry is simply removed from $L_{N,S}$. Furthermore, to limit the memory requirements, the length of every $L_{N,S}$ is limited to ten entries.

Organizing the information about other nodes in this way has two properties: (1) the nodes belonging to the same /16 subnet are ordered in their respective $L_{N,S}$ such that the more recently a node has been seen, the closer to the first position in $L_{N,S}$ it is, and (2) the subnets in L_S themselves are ordered such that the more recently a node has been seen, the closer the corresponding subnet is to the first position in L_S. After a node has been participating in MorphMix for a while, its L_S will contain entries for nearly all subnets that contain at least one node. Since nodes may join and leave the system at any time a node never knows about all other nodes. However, this is no problem because for honest nodes, it is sufficient to know about nodes in nearly all /16 subnets (e.g. 80%) that contain at least one node to pick them as their neighbors and offer them in selections from a much wider spectrum of /16 subnets than malicious nodes do.

To pick a new neighbor, the initiator randomly selects a subnet from L_S and gets (and removes) the information about the first node in the corresponding $L_{N,S}$. If the node can be contacted and is willing to accept further anonymous tunnels, it is used as a new neighbor. Otherwise, the same is tried using the next node in $L_{N,S}$. If this fails for all nodes in the selected subnet, the subnet is removed from L_S and another subnet is tried. This guarantees honest nodes pick their neighbors, and therefore the nodes they offer in their selections, from a wide variety of /16 subnets that contain MorphMix nodes. Note that witnesses (see Section 2) are basically selected using the same method, but to make sure that a high percentage of attempts to set up an anonymous tunnel succeed, it is desirable that the witnesses the initiator selects are online with high probability. Witnesses should therefore be picked "close" to the first position in L_S, i.e. from the nodes that have been inserted more recently.

The nodes in newly arriving selections are only inserted into the most recently seen subnets list if the corresponding correlation is not above the correlation limit. So we have actually combined peer discovery and collusion detection to minimize the number of malicious nodes in the list. For the adversary to compromise an anonymous tunnel, controlling the first intermediate node is a requirement. To make sure that the nodes he controls are selected as often as possible as first intermediate nodes, he needs to include many or only malicious nodes in the selections. But since the collusion detection mechanism detects these malicious selection with high probability, the adversary cannot advertise malicious nodes as aggressively as he would like.

6 Scalability

The key to scalability in MorphMix bases on the fact that although there may be as many participating nodes as there are public IP addresses, the number of /16 subnets has a strict upper bound.

Our measurements [16] have shown that the effectiveness of the collusion detection mechanism depends on both the number of nodes offered in a selection (n_{sel}) and the number of extended selections in L_{ES} (k_{ES}). Using experiments, we have derived reasonable values for both sizes. They depend on the number s of different /16 subnets that contain MorphMix nodes. The selection size to be used is defined by $n_{sel} = \max(3, \lceil 7.75 \cdot \log_{10} s - 17 \rceil)$. Assuming there are MorphMix nodes in every public /16 subnet, the maximum selection size is given by $n_{sel,max} = \lceil 7.75 \cdot \log_{10} 56559 - 17 \rceil = 20$. This also implies it is sufficient for a node must have at least 20 neighbors that are willing to accept further anonymous being routed through them at any time. If $\overline{n_{sel}}$ is the average number of nodes in a selection, the number of extended selections in L_{ES} is defined by $k_{ES} = \lceil 2 \cdot \frac{s}{\overline{n_{sel}}} \rceil$. There is also an upper bound for the size of L_{ES}, which is given by $k_{ES,max} = \lceil 2 \cdot 56559/20 \rceil = 5656$. We carried out some performance tests on a system with a 1GHz AMD Athlon CPU and 256 MB RAM. With both n_{sel} and k_{ES} set to their maximum values, it takes about 50 ms to compute the correlation of a new extended selection. Assuming an initiator sets up one anonymous tunnel every two minutes (see Section 8) and the tunnel length is five, this only consumes about 0.125% of the computing power available on the system mentioned above, which can be neglected. Similarly, the maximum size of L_{ES} is 5656 entries with 21 IP addresses each, corresponding to less then 0.5 MB memory space, which is hardly an issue for state-of-the-art computers.

Peer discovery also scales well because L_S has at most 56559 entries. The information about a node includes four bytes for the IP address, two bytes for the port, 256 bytes for the RSA modulus, and one byte for the node level. Since there may be up to ten entries in every $\mathrm{L}_{N,S}$, the maximum size of L_S is about 150 MB. While this is not insignificant, it can well be handled by modern systems. In addition, there is always the possibility to reduce the number of entries in a $\mathrm{L}_{N,S}$ to reduce the memory requirements.

7 Analysis of the Collusion Detection Mechanism

Results from our earlier paper [17] have shown that it is not advisable for the adversary to always include only malicious nodes in malicious selections because such a selection is virtually always detected by the collusion detection mechanism. Including fewer malicious nodes makes malicious selections more similar to honest selections and less detectable. We name the number of malicious nodes the adversary offers in malicious selection the *attack level*. We have analyzed several strategies [16] an adversary may employ by varying the attack level depending on the position of a malicious node in a tunnel and have come to the conclusion that the most effective way is to attack always with the same attack

level, i.e. malicious nodes always offer the same number of malicious nodes in their selections. The main reason is that the adversary can get all information to carry out this attack optimally, because observing the system tells him the approximate number of different /16 subnets with nodes in the system, which tells him the optimal attack level. Note that there are strategies that are slightly more effective in theory, for instance attacking only if the adversary controls the first intermediate tunnel. However, these attacks requires a malicious nodes to correctly "guess" its position in a tunnel during the setup, which is very difficult in practice, in particular if the initiator introduces random delays of several seconds between receiving a message and forwarding the next during tunnel setup.

Since MorphMix aims at providing anonymity for a large number of users, we analyze the performance of the collusion detection mechanism when there are nodes in nearly all public /16 subnets. We also take different capabilities of the nodes into account, i.e. some nodes have slow dial-up connections and can only relay few tunnels of others, which means they are chosen less frequently as neighbors (see Section 5) and therefore also offered less frequently in selections. Then there are nodes with very good network connectivity that can relay many data for others. As a basis for the kind of nodes that may participate in MorphMix, we use a measurement study [20] about the peers in the Napster and Gnutella file sharing systems. One main result of the study is the distribution of the bandwidths of the peers, and based on these results, we define a distribution for the bandwidths of MorphMix nodes that we assume to be realistic. To do so, we define six *node levels* and nodes are categorized according to their bandwidths. Depending on the node level, we define acceptance probabilities, which is the probability a node accepts relaying further anonymous tunnels when it is contacted as a new neighbor by another node. The left half of Table 1 illustrates the node levels and their up- and down-stream bandwidths, the distribution of MorphMix nodes over the node levels, and the acceptance probabilities. Note that these assumptions are only valid for honest nodes. We describe a different model for malicious nodes below.

Table 1. Assumed realistic bandwidth distribution of MorphMix nodes and acceptable intermediate and final nodes.

node level	bandwidth (Kb/s) up/down-stream	frac. of all nodes	acc. prob.	acceptable intermediate and final nodes					
				ISDN	$ADSL_{256}$	$ADSL_{512}$	DSL_{512}	T1	T3
ISDN	64/64	10	0.05	•	•	•	•	•	•
$ADSL_{256}$	64/256	0.25	0.1			•	•	•	•
$ADSL_{512}$	128/512	0.25	0.2				•	•	•
DSL_{512}	512/512	0.25	0.5				•	•	•
T1	1544/1544	0.1	0.8				•	•	•
T3	4632/4632	0.05	0.95				•	•	•

Looking at Table 1, we can see that we assign ISDN nodes a very small acceptance probability of 0.05, which implies that these nodes are only capable

of accepting anonymous tunnels in one out of 20 cases when they are picked as a new neighbor by another node. Conversely, we assume fast nodes can nearly always accept being selected as a neighbor and we therefore assign T1 and T3 nodes an acceptance probability of 0.8 and 0.95, respectively. Note that we have not explicitly listed nodes with Cable connections because the bandwidths they offer are the same as ADSL or DSL connections. Therefore, the ADSL and DSL nodes in Table 1 also include nodes with Cable connections.

A second valuable result from the measurement study are the up-times of the peers. It shows that the probability a peer is connected to the Internet at any time is nearly evenly distributed between zero and one, with the exception that hardly any peer is nearly never or nearly always online. Applied to MorphMix, it is reasonable to assume that dial-up nodes are online and participating in MorphMix for only a relatively short time and the fast T1 and T3 nodes are nearly always up. We therefore model the up-times of honest nodes as follows:

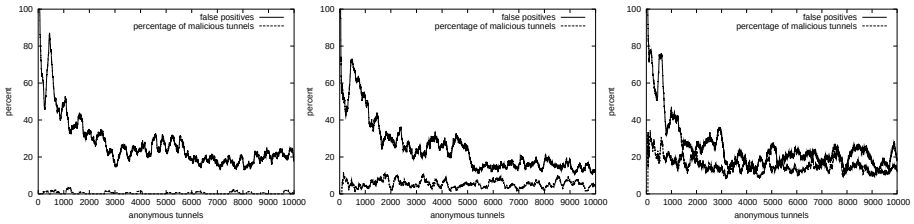
- ISDN nodes are online during one hour a day, which means their up-time probability is 1/24.
- T1 and T3 nodes have an up-time probability of 0.9.
- All other nodes get randomly an up-time between 1/24 and 0.9.

To be most effective, the adversary makes sure that the malicious nodes are participating in MorphMix as often as possible. In addition, to be involved in as many anonymous tunnels as possible, the malicious nodes should always accept further anonymous tunnels, We therefore assign all malicious nodes per default an acceptance probability and an up-time probability of one.

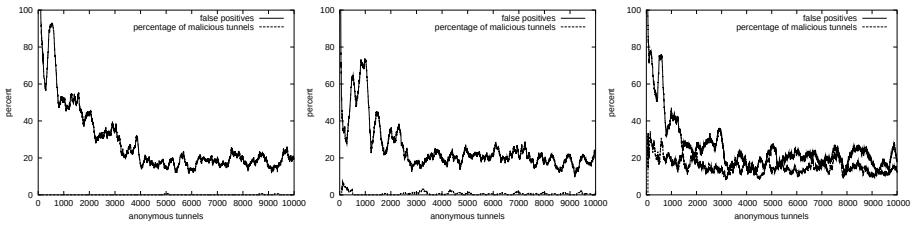
Taking into account nodes with very different bandwidths, we must think about the quality of the nodes along an anonymous tunnel. Basically, the slowest node in a tunnel determines the maximum throughput of the tunnel: if one intermediate node is an ISDN node and all the others, including the initiator, are T3 nodes, the throughput of the tunnel will be at most 64 Kb/s. This is a significant problem because hardly any user is willing to sacrifice her fast Internet connection for anonymity if all she gets is the equivalent of a slow dial-up connection. The only way to cope with this problem is to make sure no slow nodes are present along tunnels of fast initiators. In practice, this means that the initiator specifies a minimum node level for the nodes it accepts and intermediate nodes offer only nodes in selections that meet or exceed this minimum level. The right half of Table 1 specifies reasonable acceptable node levels for the intermediate and final nodes depending on the node level of the initiator.

We analyze how well the collusion detection mechanism copes with the realistic acceptance and up-time probabilities defined above. We look at two scenarios: one system with 100000 honest nodes in 50000 subnets and a large system with 1000000 honest nodes in 50000 subnets. We assume the adversary manages to operate 10000 malicious nodes that are located in 1000, 2000, 5000, or 10000 different subnets that also contain honest nodes. We always set up 10000 tunnels, starting with an empty extended selections list, and use a tunnel length of five. Our main measure to assess the effectiveness of the collusion detection mechanism is the percentage p_{a_m} of malicious tunnels among the accepted tunnels.

Besides p_{a_m}, we also show the percentage of *false positives*, i.e. the percentage of good tunnels that were wrongly classified as malicious. The data are represented as a rolling average over the 200 most recently set up anonymous tunnels. Figure 1 illustrates the results for both scenarios with malicious nodes in 1000, 5000, and 10000 subnets, respectively. The table below the graphs give the optimal attack level (oal) and p_{a_m} with and without collusion detection for malicious nodes in 1000, 2000, 5000, and 10000 subnets. We assume the initiator belongs to the four fastest types of nodes in Table 1, which corresponds to the worst case since the spectrum of nodes that can be offered in selections is smallest. The figures in parenthesis give p_{a_m} if no tunnel optimization according to the right half of Table 1 were made, i.e. if every node would accept every other node in its tunnel.



a) 100000 honest nodes, malicious nodes in 1000, 5000, and 10000 subnets



b) 1000000 honest nodes, malicious nodes in 1000, 5000, and 10000 subnets

subnets with mal. nodes	100000 honest nodes			1000000 honest nodes		
	oal	p_{a_m} with collusion detection	p_{a_m} without collusion detection	oal	p_{a_m} with collusion detection	p_{a_m} without collusion detection
1000	4	0.45 (0.28)%	5.74 (5.34)%	7	0.05 (0.05)%	1.79 (1.71)%
2000	5	1.13 (0.91)%	11.26 (10.04)%	8	0.18 (0.14)%	3.16 (2.95)%
5000	7	5.23 (4.47)%	24.04 (22.90)%	11	0.60 (0.52)%	5.71 (5.50)%
10000	10	14.17 (12.42)%	44.64 (42.54)%	14	2.37 (2.20)%	9.51 (9.23)%

Fig. 1. 100000 (a) and 1000000 (b) honest nodes; 10000 malicious nodes.

Figure 1 delivers several interesting results. First of all, it takes setting up about 4000 anonymous tunnels until the rate of false positives reaches and remains at approximately 20%. The reason for this is that the collusion detection mechanism works conservatively in the beginning to keep p_{a_m} small, but at the cost of more false positives. To make sure this learning phase happens only once,

the extended selections list is periodically stored on disk and reloaded in case a node has been offline for a while. We also see that for the adversary, it is much better to operate only one or a few nodes in as many different subnets as possible than several nodes in a smaller number of subnets. This exactly what we wanted to achieve with the collusion detection mechanism operating on the 16-bit prefixes rather than the IP addresses themselves. In addition, Figure 1 illustrates that increasing the number of honest nodes makes the system significantly more resistant to attacks. This can be explained with the way honest nodes pick their neighbors (see Section 5): with 100000 honest nodes, only very few honest nodes are stored in the $L_{N,S}$ per subnet and the probability a malicious node is picked is much larger than with 1000000 honest nodes.

Finally, the measures to improve the throughput of anonymous tunnels only marginally increase the adversaries chances to compromise an anonymous tunnel. At first, this seems surprising because DSL_{512}, T1, and T3 account for only 40% of all nodes, which means the effective number of honest nodes for fast initiators is much smaller because slow nodes are no longer offered in selections to them. But actually, not so much has changed because looking at the acceptance probabilities in Table 1 shows that slow nodes accept relaying tunnels infrequently compared to the fast nodes, which means that even if they were accepted by fast initiators, they would be present in their tunnels rather infrequently. This implies that by requesting a minimum quality for the nodes offered in selection for fast nodes, we have merely removed occasional occurrences of slow nodes in these selections.

Assuming our threat model and looking at the results presented in this section, we conclude that the collusion detection mechanism works indeed well for large systems. It significantly reduces p_{a_m} compared to the case if no such mechanism were employed and it is very difficult for an adversary to compromise a significant percentage of all anonymous tunnels. Even optimizing the throughput of anonymous tunnels must not be paid with a significant increase in the number of compromised tunnels. In large systems, the task for the adversary becomes very complicated, because he cannot simply run many nodes in a few subnets but must be present in a large number of different subnets. Of course it could be the case that the adversary owns a part of the public IP address space, for instance a whole class A network. But this only gives him full control over 256 /16 subnets, which only enables him to compromise very few tunnels. To be effective, the adversary must have nodes under his control in very many different /16 subnets. Assuming a large system with honest nodes in nearly all public /16 subnets, the adversary must control nodes in several 1000 subnets to compromise more than 1% of all anonymous tunnels.

8 Simulation Results

To analyze the expected performance MorphMix offers to its users, we implemented our own simulator, mainly because existing generic network simulators simulate the underlying network protocols in great detail and are therefore not

capable of simulating a large number of nodes (e.g. 1000) over a large simulated time period (several hours) within a reasonable execution time. Our simulator simulates the entire MorphMix protocol and is described in [16].

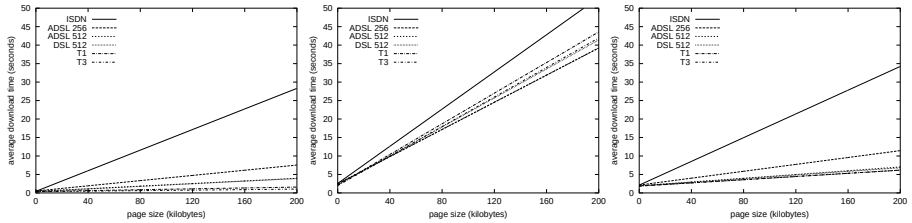
We use web browsing based on HTTP 1.1 for our analysis. The lengths of web requests and replies are modelled using appropriate values from traffic modelling and simulation literature. Web requests have a length of 300 bytes with a probability of 0.8 and 1100 bytes with a probability of 0.2 [12]. The lengths of web replies follow a ParetoII distribution with parameters $k = 2.4$ and $\alpha = 1.2$, resulting in average object size of 12 KB; the number of embedded objects per page also follow a ParetoII distribution with parameters $k = 0.8$ and $\alpha = 1.2$, resulting in an average of four embedded objects per page [9]. Finally, the reading time is defined by the time it takes between having completely downloaded a web page and initiating the next request and is also modelled by a ParetoII distribution with $k = 10$ and $\alpha = 2.0$, resulting in an average of ten seconds.

We have made several assumptions to reflect a realistic scenario. The time it takes for the data to travel between two neighboring nodes or a node and a web server and is selected randomly between 20 and 150 ms for each link. To force nodes to frequently change their neighbors, a newly selected neighbor may be offered in selection for only 30 minutes. The tunnel length is five, and every node sets up a new tunnel every two minutes on average. A tunnel may be used for at most ten minutes after it has been established, which means that at any time, a node has about five tunnels that are ready to be used. We assume it takes ten ms to process a cell in a node; if processing of data includes a DH or RSA operation, we add an additional 100 ms to the processing delay.

There are 1000 nodes in the system. More nodes are possible but the simulation time grows linearly with the number of nodes. However, we argue that even a system with 1000 nodes delivers reasonable information about how a very large system would perform if certain parameters are set accordingly. To do so, we will always use the maximum selection size of 20 (see Section 6), which implies the messages to set up a tunnel have their maximum length. We also make sure that at any time, every node has at least 30 neighbors that are willing to relay more anonymous tunnels, which implies that 20 nodes can easily be offered in selections at any time. So even if the system consisted of a million nodes, the tunnel setup messages would not be longer and the local environment every node has to handle would not be larger. The nodes' capabilities and up-times are chosen according to Section 7. For ISDN nodes, we assume their owners are browsing the web whenever the nodes are online. For all other nodes, we assume their owners browse the web during two hours a day. If the system were ten times bigger, there would also be ten times as much traffic, but also ten times as many nodes to handle it. Since the distribution of the nodes' capabilities and up-times would be unchanged, we could expect the simulation results to be very similar.

We analyze the download times for a complete web page depending on whether the web server is accessed directly or through MorphMix. In the latter case, we also compare the results with and without tunnel quality optimization according to Table 1. We simulate four hours of real time. Since the page down-

load time is nearly linearly dependent on the page size, we use linear regression to plot the graphs. Figure 2 illustrates the results.



a) direct download b) no tunnel optimization c) with tunnel optimization

Fig. 2. Download times when accessing the web server directly and through MorphMix.

The results are split into the six node levels defined in Table 1. Comparing Figures 2(a) and (b), we see that the download times get significantly longer if the web server is accessed through MorphMix without tunnel optimization. In particular, the end-to-end performance of any node drops below the performance ISDN nodes experience if the web server is contacted directly. We strongly believe that a performance loss so significant would be unacceptable for most users with reasonably fast Internet connections and hinder MorphMix from acquiring a critical mass.

Using tunnel optimization and looking at Figure 2(c), the end-to-end performance could be significantly improved. We can also clearly state that the benefits from optimizing the throughput of anonymous tunnels greatly outweighs the small increase in the number of compromised tunnels (see Figure 1). Compared to Figure 2(a), the download times have increased about 20% for ISDN nodes and about 50% for ADSL_{256} nodes. All other nodes only accept nodes with at least DSL_{512} speed in their tunnels and the performance they experience is therefore approximately equal. Their download times are now about 50% longer than those of ADSL_{512} or DSL_{512} nodes when the web server is contacted directly. Since Figure 2 does not take the time to completely display a page in the browser into account, the actual performance loss experienced by the user should be even smaller. We believe that for many users, this is an acceptable price for getting anonymity.

We now analyze the bandwidth usage and the data overhead of MorphMix assuming the web browsing scenario above. We distinguish between six different types of data: (1) web requests sent and web replies received at the initiator, which corresponds to the the data sent and received if the web server is contacted directly. (2) Cell headers and padding bits to generate fixed-length cells. (3) Forwarding of cells containing web requests and replies for other nodes. (4) Tunnel setup overhead, which includes all data sent and received to establish and tear down anonymous tunnels. (5) End-to-end (e2e) ping/pong overhead from regularly testing the quality of a tunnel. (6) Link message overhead, which

includes all messages exchanged between two neighbors to set up a link and exchange keys, for link status information, and for flow control messages.

The first three types of data are needed to fulfil the prime task of a mix network: to send and receive user data through anonymous tunnels. We therefore do not count the cell headers and padding bits to generate fixed-length cells from the user data and forwarding these cells along anonymous tunnels as overhead, because they are essential properties of any mix system. We collectively identify these three types of data as *tunnel data* The other three types are needed to provide the anonymous tunnel infrastructure and are therefore *data overhead*.

We first analyze how much of the available bandwidth is actually used by MorphMix using the scenario in Figure 2(c). We distinguish between data sent and received and between tunnel data and overhead. Figure 3(a) shows the bandwidth usage for all nodes together and for the different node types.

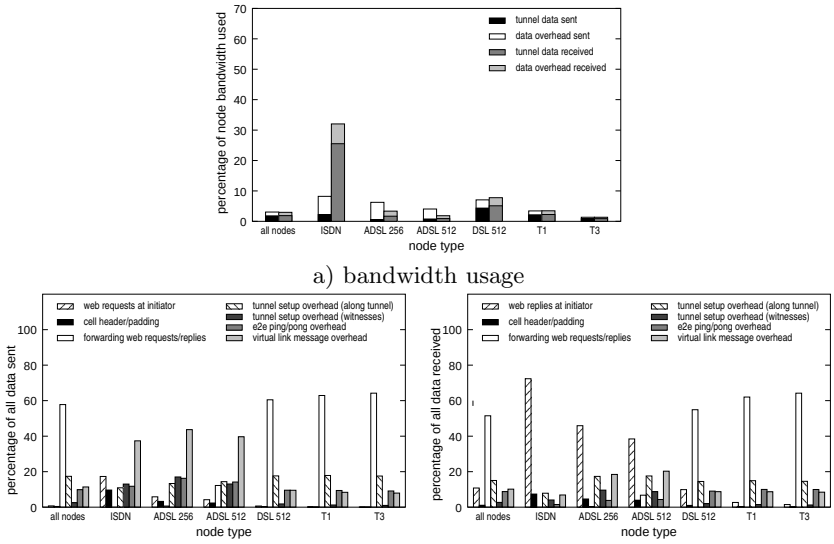


Fig. 3. Bandwidth usage and data sent and received by the nodes.

Overall, about 3% of the total bandwidth available is used by MorphMix. This is quite a small total burden and Figure 3(a) shows that all nodes with at least a ADSL_{256} connection can easily run a node without noticing a significant drop in terms of network performance for other applications. The reason why relatively much of the down-stream bandwidth of ISDN nodes is used is that their bandwidth is generally quite small and that we assume that ISDN users are always browsing when they are online. About 61% of all data are tunnel data and 39% are overhead. The overhead is therefore relatively large but since the total load on the nodes is so small, it can easily be dealt with.

To analyze the data sent and received by the nodes in more detail, Figures 3(b) and (c) illustrate how much of the used bandwidth is spent on which type of data. The biggest part – about 55% – is spent on forwarding the web requests and replies of other nodes and only relatively little is spent to handle the own data, which is reasonable. Tunnel setup and teardown overhead is responsible for about half of all overhead and for about 19% of all all data. About 16.5% stem from the nodes along the tunnel and 2.5% from the witnesses when appending a node. End-to-end ping/pong messages are responsible for about 9% and the various link messages exchanged between neighbors for about 11% of all data. Looking at the different node types, the bandwidth that is spent for handling the data of other nodes gets the bigger the faster the Internet connection to the node is. This is reasonable because according to our assumptions about realistic capabilities and up-times in Section 7, faster nodes accept relaying anonymous tunnels more frequently and are online more often.

We have also analyzed the impact of nodes that frequently crash or that can temporarily not be reached for any reason because this renders the corresponding tunnels useless and therefore also stops ongoing web page downloads along these tunnels. Without going into the details [16], we only state that MorphMix is still able to deliver satisfactory performance for application such as web browsing, although a small percentage of the web pages must be requested again.

For our analyses in this section, we have assumed that all nodes relay data for others. However, many peer-to-peer systems, especially those for file sharing, suffer from the "free rider" problem because there is often no real incentive to provide services to others because everything is for free and the systems seem to work well enough even if most users are free riders. If 90% of all MorphMix nodes were free riders the, load on the other 10% them could get quite high and the performance may suffer. However, the advantage of MorphMix compared to other peer-to-peer systems is that MorphMix provides incentive to relay the data of others. This has to do with the fact that the first intermediate node in a tunnel cannot easily learn that it is the first intermediate because no such information is leaked during the setup of the anonymous tunnel. So if a node is accused of having contacted a host anonymously, its operator can claim she only relayed the data for another node (plausible deniability). Traditional mix systems do not have this property because the clients and mixes are strictly separated. On the other hand, if a node a is a free rider in MorphMix, other nodes can learn about this by trying to pick a as a new neighbor. If this always fails or if a never accepts relaying tunnels, it can be concluded with very high probability that all data sent or received by a belong to tunnels of which a is the initiator. This implies that a cannot plausibly deny being the initiator of a tunnel and reduces a's anonymity compared to other nodes that relay the data of others.

9 Comparison with Similar Systems

We compare MorphMix with two other peer-to-peer based system aiming at anonymous low-latency Internet access: Crowds [15] and Tarzan [10].

Crowds requires a centralized lookup server to keep track of the nodes. This is a major drawback, first of all because it provides a single point of failure and attack and second because the lookup server must inform all participating nodes about any membership changes. The latter makes Crowds not well suited to support many nodes (e.g. several 1000s) that come and go. Crowds also neither employs layered encryption nor a collusion detection mechanism. Assuming the requester picks a malicious node to which it forwards the request and that node can find out that its predecessor is indeed the requester, it has broken the anonymity. To protect from this attack in the case of web browsing, the last node in the circuit retrieves the page including all embedded objects before sending it back to the requester. This prevents the malicious node from easily making use of a timing attack to learn whether it is directly following the requester or not because embedded objects would be requested by the browser automatically. The disadvantage is that the last node must parse HTML objects to get all embedded objects, which is impossible if HTTPS is used. There are additional possibilities for a requester to leak information (clicking on a hyperlink, HTTP redirects) that can be used for a timing attack by the node directly following the requester. The Crowds' designers propose to introduce random delays to complicate this attack, but this reduces the end-to-end performance and could refrain potential users from using the system. In general, it is always possible to introduce such application-dependent measures, but they also imply limiting the capabilities of the system a bit. The collusion detection mechanism as employed in MorphMix is a much cleaner solution because it tries to guarantee that anonymous tunnels are "secure" with high probability *before* any information about hosts to be contacted anonymously are revealed to the final node. Consequently, no such measures as employed by Crowds are required.

Tarzan builds an universally verifiable set of neighbors (the mimics) for every node, which requires a node lookup mechanism that can keep track of all nodes currently participating in Tarzan. This makes it unlikely Tarzan can function well in a large and dynamic environment where nodes come and go. Apart from this drawback, the fact that every node selects its neighbors in a pseudo-random way means all nodes along a circuit are also chosen pseudo-randomly from the set of all nodes. Consequently, when we talk about *collusion detection* in MorphMix, we can identify the mechanism employed by Tarzan as *collusion prevention*. However, this also means there is only little room for throughput optimization because the potential next hop nodes are limited to a node's mimics.

10 Conclusions

We have shown that MorphMix indeed provides practical anonymity for low-latency applications for a large number of users. In particular, MorphMix offers acceptable performance and provides good protection from long-term profiling. An important advantage of MorphMix compared to similar peer-to-peer based systems is that it does not rely on a lookup service that must keep track of all nodes that are currently participating, which makes it highly robust to mem-

bership fluctuations. In addition, MorphMix scales very well because every node handles only its local environment, which is nearly independent of the number of nodes in the system. Finally, the collusion detection mechanism also scales well because its complexity is bounded by the maximum number of /16 subnets.

As always, some open issues remain. Since nodes tend to fail or disappear more often than the mixes in traditional mix systems, MorphMix is less well suited for applications using long-standing TCP connections such as remote logins. Possible solutions are to bypass nodes that have failed but doing so could enable an attack where malicious nodes claim that their honest successor node along a circuit has failed. Another problem is exit abuse. What if a Yahoo account is accessed through MorphMix to send a threatening e-mail message? Will the last node in the chain be accused? This problem seems more significant in peer-to-peer-based than in traditional mix systems, because in the latter, the operator can "more plausibly" argue about not having sent the message himself. One way to solve this problem are exit policies using blacklists, but it is difficult to keep them up-to-date. Another potential problem are DoS attacks by malicious nodes that simply do not forward data for others. To solve this problem, one could couple MorphMix with a reputation system. Research on reputation systems is still in its infancy, but initial studies to make mix networks more reliable through reputation have been carried out [7, 8]. Finally, a lot of research remains to be done to develop efficient cover traffic mechanisms that significantly increase the protection from targeted attacks.

References

1. Alessandro Acquisti, Roger Dingledine, and Paul Syverson. On the Economics of Anonymity. In Rebecca N. Wright, editor, *Proceedings of Financial Cryptography (FC '03)*. Springer-Verlag, LNCS 2742, January 2003.
2. Adam Back, Ulf Möller, and Anton Stiglic. Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems. In *Proceedings of 4th International Information Hiding Workshop*, Pittsburg, PA, USA, April 2001.
3. Oliver Berthold and Heinrich Langos. Dummy Traffic Against Long Term Intersection Attacks. In *Proceedings of the 2nd Workshop on Privacy-Enhancing Technologies*, San Francisco, CA, USA, April 14–15 2002.
4. Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The Disadvantages of Free MIX Routes and how to Overcome them. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.
5. Philippe Boucher, Adam Shostack, and Ian Goldberg. Freedom Systems 2.0 Architecture. White Paper, `http://www.homeport.org/~adam/zeroknowledgewhitepapers/Freedom_System_2%_Architecture.pdf`, December 2000.
6. David L. Chaum. Untraceable Electronic Mail, Return Adresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.
7. Roger Dingledine, Michael Freedman, David Hopwood, and David Molnar. A Reputation System to Increase MIX-net Reliability. In *Proceedings of 4th International Information Hiding Workshop*, pages 126–141, Pittsburg, PA, USA, April 2001.

8. Roger Dingledine and Paul Syverson. Reliable MIX Cascade Networks through Reputation. In *Proceedings of Financial Cryptography 2002*. Springer-Verlag, March 2002.
9. Anja Feldmann, Anna C. Gilbert, Polly Huang, and Walter Willinger. Dynamics of IP Traffic: A Study of the Role of Variability and the Impact of Control. In *Proceeding of SIGCOMM '99*, Massachusetts, USA, September 1999.
10. Michael J. Freedman and Robert Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, D.C., USA, November 2002.
11. Dogan Kesdogan, Dakshi Agrawal, and Stefan Penz. Limits of Anonymity in Open Environments. In Fabien Petitcolas, editor, *Proceedings of Information Hiding Workshop (IH 2002)*. Springer-Verlag, LNCS 2578, October 2002.
12. Bruce A. Mah. An Empirical Model of HTTP Network Traffic. In *Proceeding of Infocom 1997*, pages 592–600, Kobe, Japan, April 1997.
13. Jean-François Raymond. Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.
14. Michael Reed, Paul Syverson, and David Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, May 1998.
15. Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, November 1998.
16. Marc Rennhard. Anonymity for the Masses with MorphMix (available at `http://www.tik.ee.ethz.ch/~rennhard/publications/morphmix_tr2.pdf`). TIK Technical Report Nr. 159, TIK, ETH Zurich, Zurich, CH, May 2003.
17. Marc Rennhard and Bernhard Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *Proceedings of the Workshop on Privacy in the Electronic Society*, pages 91–102, Washington, DC, USA, November 21 2002.
18. Marc Rennhard and Bernhard Plattner. Practical Anonymity for the Masses with Mix-Networks. In *Proceedings of the IEEE 8th Intl. Workshop on Enterprise Security (WET ICE 2003)*, Linz, Austria, June 9–11 2003.
19. Marc Rennhard, Sandro Rafaeli, Laurent Mathy, Bernhard Plattner, and David Hutchison. An Architecture for an Anonymity Network. In *Proceedings of the IEEE 6th Intl. Workshop on Enterprise Security (WET ICE 2001)*, pages 165–170, Boston, USA, June 20–22 2001.
20. Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.
21. Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an Analysis of Onion Routing Security. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.
22. Matthew Wright, Micah Adler, Brian Neil Levine, and Clay Shields. An Analysis of the Degradation of Anonymous Protocols. In *Proceedings of ISOC Network and Distributed System Security Symposium (NDSS 2002)*, San Diego, USA, February 2002.