Route Fingerprinting in Anonymous Communications

George Danezis
K.U. Leuven, ESAT/COSIC
Kasteelpark Arenberg 10
B-3001 Leuven-Heverlee, Belgium
George.Danezis@esat.kuleuven.be

Richard Clayton
University of Cambridge
Computer Laboratory, JJ Thomson Ave.
Cambridge, CB3 0FD. United Kingdom
Richard.Clayton@cl.cam.ac.uk

Abstract

Peer discovery and route set-up are an integral part of the processes by which anonymizing peer-to-peer systems are made secure. When systems are large, and individual nodes only gain random knowledge of part of the network, their traffic can be detected by the uniqueness of the information they have learnt. We discuss this problem, which occurred in the initial design of Tarzan, and other related problems from the literature.

1. Introduction

In Chaum's original work on mix networks [1] it is assumed that if every participant in the mix network also acts as a mix for others, this will improve the overall security of the network. Recent interest in peer-to-peer networking has influenced some researchers to re-examine networks with a large number of transient mixes.

In particular, Freedman *et al* designed *Tarzan* [4], a peer-to-peer network in which every node is a mix. A node initiating the transport of a stream through the network creates an encrypted tunnel to another node, and asks it to connect the stream to the next node that the initiator has chosen. By repeating this process a few times it is possible to have an onion-encrypted connection, relayed through a sequence of intermediate nodes.

The original Tarzan design, presented at IPTPS 2002 [5] only required each node to learn about a random subset of other nodes in the network. This is clearly desirable because of the very dynamic nature of peer-to-peer networks, and the volatility of node presence. We found two attacks against this strategy at that time and communicated them privately to Freedman and his colleagues. The final version of Tarzan [4] avoids these attacks, but does so partly by requiring each node to learn the identity of all others, which is clearly less practical in a peer-to-peer setting.

We have recently noted that other anonymity designs either fail to discuss in detail the relevant issues around node discovery and route (re)construction, or have avoided the attacks by means of a fortunate side-effect of some other mechanism, introduced for some other purpose. We therefore believe that recording our attacks in the literature, even at this late stage, will prevent other system designers from falling prey to them.

2. Attacks against the original Tarzan design

The Tarzan system aims to provide anonymous transport for streams of data. To achieve this it implements the ideas of mixes [1] over a peer-to-peer network.

Traditional mixes are dedicated routers that batch together many messages, or streams, in such a way that inputs are difficult to unambiguously link to outputs. Public-key cryptography is used to ensure that an observer cannot match the messages based on their contents, and the timings of messages are disrupted to foil traffic analysis. In addition, a message or stream is typically relayed through many mixes, and if even just one of them is 'honest' some anonymity is provided. In order to relay messages through these chains of mixes, a client needs to learn their network addresses and their public keys. Centralised servers, often called 'directory servers', provide this information for all mixes in the network. The security of the system relies on the lists of mixes and keys not being manipulable by an adversary. Such an adversary could attempt to exclude all honest mixes from the list, or impersonate the honest ones by substituting their public keys.

Tarzan allows all nodes, in a potentially very large (millions of machines) peer-to-peer network to mix traffic for each other. In line with the typical assumptions of peer-to-peer networks, all nodes are assumed to be transient. Both the number of nodes, and the consequent high churn, make keeping track of all nodes in the network very difficult and expensive, even for a well-resourced centralised directory. Such a centralised directory is also counter to the spirit of

peer-to-peer networking that tries to decentralise all functionality. Therefore, the original Tarzan design assumed that each node would only learn about a small, randomly chosen, subset of other network nodes. This design decision leads directly to our *node knowledge profiling* attack.

A high rate of churn of peers means that there must be frequent reconstruction of routes, which is potentially very expensive. So if one of the nodes making up the route fails or leaves the network, the initially presented Tarzan design opted for 'mending' just the part of the route with a problem. This is, in theory, less costly than reconstructing a broken route from scratch, but it also leads directly to our *route reconstruction attack*.

2.1. Route reconstruction attack

The early design of Tarzan [5] proposed that nodes should start off by learning the identity of a number of nodes by performing random lookups on a Chord ring [11]. This list of nodes is periodically added to by further lookups, perhaps once a minute. When the Tarzan network is to be used for anonymous communication, some nodes are randomly chosen from the set that has been learnt and a route is then constructed through these nodes.

This route is set up by communicating with each node in turn (using the route as set up so far) and recording a flow identifier, the identities of the adjacent hops and an encryption key for return traffic. This route is then used by creating an 'onion' for each packet of data. The intermediate nodes each remove a layer of the onion (using their knowledge of their own private key) and use the flow identifier to obtain the next hop for the traffic. Eventually the packet will reach the far end. The reverse traffic is passed back along the route, but is re-encrypted at each stage of the return journey. When it emerges from the system the multiple layers of encryption are removed by the tunnel-builder and the data exposed.

In order to achieve resilience against intermediate nodes leaving the network, or proving unreliable, a route reconstruction protocol is provided. This protocol is designed to be low cost and just routes around the failed node. Rebuilding the entire route would be expensive, so the working hops are retained and only the failing connections are replaced.

However, by the use of active attacks on intermediate nodes, an attacker who can inspect traffic leaving these nodes will be able to exploit the protocol so as to infiltrate the chain and to ultimately compromise the entire path through the Tarzan network.

Let us assume that the attacker controls a fraction c of subverted nodes. The initial probability that a chain of length l is totally controlled by an attacker is, for a large network, c^l.

However, if the attacker can cause one of the good nodes (that he does not control) to fail, then the Tarzan route reconstruction protocol will replace that single node by making another selection from the pool.

The attacker can induce such a failure either by launching a denial of service attack directly across the Internet or by overloading the node by routing large amounts of traffic across the Tarzan network, with all of this traffic going through the node. The latter scheme would be preferable because the attacker should find it simpler to hide.

The attacker can then cause the replacement node to fail in a similar manner until eventually the user selects one of the subverted nodes.

The attacker then turns their attention to the next node along the path (which they will now be aware of) to try to make the user select a malicious node for that position as well. This attack can be mounted in a linear fashion either from the source towards the destinations it is accessing, or from the destination towards the sources of traffic.

The attack can be made even simpler, once the attacker controls one node in the path. There is no longer a need to mount active denial-of-service attacks, since the malicious node can merely discard traffic packets. Once the source realises that its traffic flow has ceased it will send out 'ping' packets to attempt to determine which hop has failed. The attacker also discards pings for subsequent hops and the user will erroneously conclude that the next hop has failed, and will try to route around it.

The attack is rather faster to mount than might naïvely be expected. The expected number of nodes that must be disabled before a subverted node is selected has a geometric distribution with a mean of $m = 1/c$. The attack operation needs to be repeated l times, once for each node along the chain. Therefore, on average, the attacker needs to disable l/c nodes until the path is fully controlled.

Note that the attack can be applied to subvert all traffic from all users or, by using knowledge about which nodes a particular user has learnt about, it can be directed at one particular user's traffic.

2.2. Node knowledge profiling

In the early design of Tarzan the user discovers the identity of participating nodes by using a pseudo-random number generator as an index into a Chord ring that returns the node identities. Tarzan uses this random selection process because it is intended to scale to networks so large that it would not be feasible for every node to have a complete view of the network.

An attacker is able to observe the node selection process, either by seeing the messages from the Chord ring or by inspecting the traffic subsequently exchanged with the nodes that the Chord ring has identified. Clearly if the user solely established the identity of the nodes that were to be used

to forward messages then this would compromise the path. Tarzan avoids this trap by arranging to discover a large number of nodes but only using a small subset for a particular path. Unfortunately, the attacker can exploit their knowledge of which nodes were known to the user to probabilistically identify traffic as it passes across the Tarzan network.

Let us suppose that there are N nodes in the Tarzan network and the user establishes the identity of a random subset of size k. The attacker can now inspect traffic at any of the k nodes that the user is aware of. There is no need to decode any traffic – just inspect the source and destination. The attacker needs to determine if any traffic is arriving from a node that is known to the user and also if any traffic is departing to another node known to the user. If the node is not both sending and receiving such traffic then the user is not currently using it. If it *is* both sending and receiving such traffic then the user *may* be using it – and where k is small compared to N it is *very likely indeed* that it is the user's traffic being observed and not some other participant in the network.

We will calculate the expected number of nodes that could have constructed a path including three observed intermediate nodes (the triplet).

1. We assume, for simplicity, that all participants choose k nodes from the total set of N. Tarzan ensures this selection is uniformly random by requiring users to query a Chord ring.
2. Each node can generate $\binom{k}{3}$ triplets out of the $\binom{N}{3}$ that exist, where $\binom{n}{m}$ is the number of possible ways of selecting m elements from n. Therefore given a random triplet of nodes the probability it is known to a given node is $p = \frac{\binom{k}{3}}{\binom{N}{3}}$.
3. For any particular triplet, the number of nodes that could choose that triplet will follow a binomial distribution with parameters p and N. Each out of the N nodes has a probability p of knowing the particular triplet. The expected number of nodes[1] μ that know the particular triplet is:

$$\mu = N\frac{\binom{k}{3}}{\binom{N}{3}} = \frac{k(k-1)(k-2)}{(N-1)(N-2)} \approx \frac{k^3}{N^2} \qquad (1)$$

We can now calculate when a node is vulnerable to the attack. If we wish the expected number of nodes that know a particular triple to be one or less (i.e. they are, on average, uniquely identifiable) then we need the result of equation (1) to be less than or equal to 1.

$$\frac{k^3}{N^2} \leq 1 \Rightarrow k \leq N^{2/3} \qquad (2)$$

[1]The variance is $\sigma^2 = N\left(\frac{\binom{k}{3}}{\binom{N}{3}}\right)\left(1-\frac{\binom{k}{3}}{\binom{N}{3}}\right) \approx \frac{k^2N^3-k^5}{N^5}$

This result is extremely strong. For example, if there are 1000 nodes in the network, then if all nodes learn the identity of 100 or less participants then any triple they use will (more often than not) be unique to them.

Of course an attacker may be able to establish that traffic is flowing through a sequence of hops. Generalising equation (1) we can see that the average number of nodes that will know about a particular hop sequence of length l is:

$$\frac{N\binom{k}{l}}{\binom{N}{l}} = \frac{k(k-1)(k-2)\ldots(k-l+1)}{(N-1)\ldots(N-l+1)} \approx \frac{k^l}{N^{l-1}} \qquad (3)$$

Thus, if the attacker has the ability to observe many nodes they will be able to test all possible combinations of routes against the target node profiles. Even though the combinations of potential routes to trace might seem to grow exponentially, most of the paths can be discarded during the early stages of analysis.

3. Vulnerability of other designs

The final Tarzan design [4] was immunised against our attacks (after a private communication with the authors). The route reconstruction protocol has been abandoned in favour of reconstructing routes from scratch. Naturally, this eliminates our route reconstruction attack. Nodes in Tarzan are also assumed to discover and maintain knowledge of the full network of peers, instead of just a subset, which eliminates the node knowledge profiling attack. The need to know all other peers does not scale well in the size of the network and is likely to be very expensive in practice.

Tor [3] has a very similar architecture to Tarzan, since they both relay bi-directional streams, yet it embraces a client-server architecture. As in a traditional mix system, a (distributed and redundant) directory service provides a complete list, network addresses and public keys, for every Tor router. This approach is feasible because those routers are assumed to be reliable, with a high up-time. Clearly, having full information defeats the node knowledge attack, and routes are rebuilt from scratch after (relatively unexpected) failures so the route reconstruction attack does not apply either.

However, many other peer-to-peer anonymizing networks have been proposed in the literature [9, 12, 8]. All of them need to address the problem of peer discovery, but most designs do not provide any details as to how they envisage that this would be implemented! In particular many designs make reference to the fact that routes through the mix network might be constructed by nodes selected "at random", which suggests that if these designs were to mature and be implemented they would be faced with the attacks we describe.

Aside from the difficulty in choosing nodes "at random" out of large, distributed and transient population, such a system would also be faced with the difficulty of making the selection mechanism resistant to attack. Tarzan proposed using a random look-up into a Chord ring, but this is vulnerable to an attacker flooding the network with malicious nodes, or using malicious nodes in the Chord ring to make the look-up return a malicious node [2].

Rennhard and Plattner [10] introduced a peer-to-peer anonymizing overlay, *MorphMix*, which shares a very similar architecture and threat model with Tarzan. A crucial difference is that the route through the network is not specified by the source but chosen by intermediate nodes, observed by user specified and trusted witnesses. While our attacks do not apply directly to route selection, variants might apply to the choice of witness nodes. Other systems such as WonGoo [6, 7] and SAS [13] also allow nodes along the route to determine the path, and are therefore, by good fortune, immune to the node knowledge profiling attacks.

MorphMix realises that leaving the intermediate nodes to choose the route through the network might lead to *route capture*, whereby the first subverted mix on the path only chooses amongst other subverted mixes. Therefore MorphMix includes a *collusion detection* mechanism, that monitors for any cliques in the selection of nodes. This prevents subverted nodes from routinely running attacks on the network but does not provide security in every case.

4. Conclusions

Our analysis of the node knowledge profiling attacks shows that the random selection of nodes, out of a small subset, to provide routing through the network is extremely unwise. A very significant fraction of the nodes must be discovered; i.e. k must be large enough that the attacks become impractical, although it should be noted that any value short of N will always cause some information to leak. Our attacks could be prevented if the discovery of participant nodes was to be made unobservable, or the subsets of nodes known by users are correlated.

Similarly our path reconstruction attack is not possible if the entire path is reconstructed after a failure. This defeats Tarzan's original design goal of avoiding excess expense in such circumstances. Alternative designs avoid the problem by distributing the work of reconstructing routes, but most systems seem to just assume that nodes are reliable and so the amortized cost of reconstructing all routes from scratch is not too excessive.

Acknowledgements

We would like to thank the Cambridge-MIT Institute for supporting our visit to MIT, which led to this paper, and to Michael Freedman, Frans Kaashoek, Robert Morris and Emil Sit for their hospitality and valuable discussions.

References

[1] D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[2] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson. Sybil-resistant DHT routing. In S. D. C. di Vimercati, P. F. Syverson, and D. Gollmann, editors, *ESORICS*, volume 3679 of *Lecture Notes in Computer Science*, pages 305–318. Springer, 2005.

[3] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[4] M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In V. Atluri, editor, *ACM Conference on Computer and Communications Security (CCS 2002)*, pages 193–206, Washington, DC, November 2002. ACM.

[5] M. J. Freedman, E. Sit, J. Cates, and R. Morris. Introducing Tarzan, a Peer-to-Peer Anonymizing Network Layer. In P. Druschel, M. F. Kaashoek, and A. I. T. Rowstron, editors, *International workshop on Peer-to-Peer Systems (IPTPS)*, volume 2429 of *Lecture Notes in Computer Science*, pages 121–129, Cambridge, MA, March 2002. Springer.

[6] T. Lu, B. Fang, Y. Sun, and X. Cheng. Performance Analysis of WonGoo System. In *CIT*, pages 716–723. IEEE Computer Society, 2005.

[7] T. Lu, B. Fang, Y. Sun, and L. Guo. Some Remarks on Universal Re-encryption and A Novel Practical Anonymous Tunnel. In X. Lu and W. Zhao, editors, *ICCNMC*, volume 3619 of *Lecture Notes in Computer Science*, pages 853–862. Springer, 2005.

[8] A. Mislove, G. Oberoi, A. Post, C. Reis, P. Druschel, and D. S. Wallach. AP3: Cooperative, decentralized anonymous communication. In *Proc. SIGOPS-EW*, Leuven, Belgium, September 2004.

[9] O. Landsiedel, H. Niedermayer, and K. Wehrle. An Infrastructure for Anonymous Internet Services. In *IWI2005*, Chiba/Tokyo, Japan, May 2005.

[10] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *Workshop on Privacy in the Electronic Society (WPES 2002)*, Washington, DC, USA, November 2002.

[11] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM 2001)*, pages 149–160, San Diego, CA, USA, 27-31 August 2001. ACM Press.

[12] L. Xiao, Z. Xu, and X. Zhang. Low-Cost and Reliable Mutual Anonymity Protocols in Peer-to-Peer Networks. *IEEE Transactions on Parallel and Distributed Systems*, 14(9):829–840, 2003.

[13] H. Xu, X. Fu, Y. Zhu, R. Bettati, J. Chen, and W. Zhao. SAS: A Scalar Anonymous Communication System. In *Proceedings of ICCNMC*, pages 452–461, 2005.