MessageVortex Protocol – Sending Messages Anonymously

M. Gwerder[1,2][0000−0003−0296−5079]

[1] University of Basel, Switzerland
[2] University of Applied Sciences of Northwestern of Switzerland
martin.gwerder@fhnw.ch

Abstract. In this paper, we introduce an unobservable message anonymization protocol, named MessageVortex. It is based on the zero-trust and peer-to-peer (P2P) principle and avoids central aspects such as fixed infrastructures within a global network. It scores over existing work by blending its traffic into suitable existing transport protocols, thus making it next to impossible to block it without significantly affecting regular users of the transport medium. It furthermore requires no protocol-specific infrastructure in public networks and allows a sender to control all aspects of a message such as the degree of anonymity, timing, and redundancy of the message transport without disclosing any of these details to the routing or transporting nodes.

Keywords: Messages · Unobservable · Anonymity · Unlinkability · Pseudonymity

1 Introduction

Since whistleblower Edward Snowden disclosed documents, it seems generally accepted that global monitoring of Internet traffic is conducted. A message sent throughout the Internet must, even when perfectly encrypted, disclose at least the recipient to the router transporting a message. The sender can be identified by the return path or is identifiable by following the source of packets. Meta information is valuable because frequency and message size disclose important facts about the association and intensity of the relationship of involved parties.

This paper addresses the problems above-mentioned above of traffic monitoring by introducing a new protocol called MessageVortex. Within MessageVortex we consider the whole network as untrusted except for the sending and receiving node. MessageVortex does not leak routing information as only the immediate peers are known to a node. The protocol can sustain anonymity[22] even under harsh assumptions such as an adversary possessing huge but limited funding, unlimited monitoring capability on the network and a considerable number of own nodes.

Numerous attempts such as in [9,14,19,12,11,13] have been made to anonymize message flow. However, most of them have problems as they rely at least on

the partial trust in the nodes routing the messages, or some central infrastructures [21,1,2,3]. Exit and entry points are essential as they may leak information. By degrading the network, message flows can be redirected and information extracted from the new flows. Additionally, a dedicated transport protocol is easy to block since used ports or some protocol properties can be used to identify nodes. Furthermore, most approaches require infrastructure with fixed addressing within the internet, rendering owners vulnerable. All papers analyzed for this work introduced a new transport layer solving the anonymity problem. Only TOR defined an additional transporting mechanism which may be used as an alternate medium between two defined nodes to avoid detection. In our approach, we decouple the routing layer from the transport layer completely. By doing so, we introduce new degrees of complexity to attack scenarios, as messages may use any common transport protocol of the used network.

Our work consists of a routing layer which is completely P2P based without any central protocol specific infrastructure. Any node is a routing node and may be an endpoint. There is no implicit or explicit trust in any particular system of the network. The original sender of a message controls decoy traffic generation. Even a node generating decoy traffic is unable to differentiate between message and decoy traffic. As transport media, we use well known store-and-forward-based protocols. The routing logic has no affiliation to the transport layer. Any transport endpoint such as a free-mailer email address or chat account may be converted into a transport media for our protocol without any modification required on the server side. The broad availability of such services makes the network very agile on one side at the cost of reliability. To counter this phenomenon, we use a high degree of redundancy if required by the routing block builder. Using the MessageVortex protocol, any device with latent or permanent connectivity to the Internet may act as a routing node. By applying the zero-trust model, we give full control of all traffic to the original sender of the message. He controls message flow, redundancy, the degree of anonymity, timing, and many more aspects of the message transport throughout the whole network. A sending node may do this without disclosing any of these parameters to the participating nodes as they are encoded in the operations and only visible to the node executing them. The operations itself are chosen in such a way that they do not reveal the nature of the traffic. To limit possibilities of denial-of-service (DoS) within the system and guarantee efficient handling of messages, MessageVortex nodes (in short "node") rely on unlinked, ephemeral identities which are created in a proof of work system (PoW). While it is technically easy to use a node, it is hard to carry out traditional attacks against them as all transactions have to be authenticated.

1.1 Previous Work

Generally, not many technologies are used to achieve anonymity or unlinkability as defined in [22]. Most analyzed protocols use relays[5], mixes[5], or Dining-Cryptographers-related-networks[6] or their variants to achieve anonymization. Numerous protocols have evolved from these technologies:

- *TOR*[12]: Mixer-based infrastructure for tunneling TCP-based protocol streams. TOR is a near synchronous routing system. The anonymization is based on mixing by using a temporary statical path consisting of an entry node, an exit node, and at least three more intermediate nodes.
- *Mixmaster*[19]: A type-II remailer where all mixes may choose the path on their own.
- *Babel*[14]: Mixer-based remailer where the sender chooses the path and sends an onionised message.
- *Mixminion*[9]: A type-III remailer offering sender anonymity. Unlike their predecessors, it is no longer based on the SMTP transport protocol. This system requires at least a centralized directory infrastructure.
- *Freehaven*[11]: A distributed storage system. The system offers anonymous document storage. A user downloading a document requires the hash of a public key used to sign the document. Known documents may be identified, even if no key is known, and owners of infrastructure might be held responsible if hosting such well-known documents when not conformant to the local jurisdictional zone.
- *Freenet*[7]: Freenet is an anonymous, distributed data storage system. The system does not trust any server. Instead, a reputation system is used. This system has attracted very little attention from the researcher community.
- *Herbivore*[13]: A DC-net-based protocol without client implementation.
- $\mathcal{P}^5$[23]: There is a simulator available for this protocol. Real-world implementations do not exist, and therefore no attack schemes have been elaborated so far.
- I^2P(geti2p.net): P2P-based pseudonymous protocol allowing TCP and UDP streams to be tunneled synchronously or near-synchronously. Unlike TOR, I^2P works pseudonymously and mixes using packet switching.

Our protocol differs from these works in several ways. There is no central network infrastructure. There are no entry or exit nodes which might be blocked. All nodes including the sender and the recipient are treated equally. The number of nodes, the traffic to be generated, anonymity sets, timing of the message, redundancy in message transmission, and size of all packages to be sent along is solely decided by the builder of a routing block. Furthermore, there is no dedicated transport protocol. Instead, MessageVortex messages (in short "vmessages") are embedded in other existing Internet protocols.

2 Methods and Material

The protocol is described precisely in [16]. All necessary details to implement the protocol have been defined in this document.

The protocol is divided into the three layers blending (embedding vmessages into transport), routing (processing vmessages), and accounting (prevents misuse and DoS). These three layers are connected via the fourth layer (transport). The transport layer is based on one or more store-and-forward based, common internet transport protocols. All cryptographic operations such as encryption,

decryption, hashing, or random number generation within the protocol do not rely on a single algorithm. The protocol can signal what capabilities a node has and how exactly a message should be processed. This makes the protocol very robust if a used algorithm is broken.

2.1 Protocol Layers

The transport layers provide the Internet infrastructure. Unlike in most other approaches such as [12,23,7], this layer is not protocol specific. We use already existing, symmetrically built store and forward protocols. Attributes such as anonymity do not rely on the security of this layer. Protocols on this layer are typically well known and frequently used. They have no prerequisite for encryption or privacy and are store-and-forward based protocols with routing capabilities.

The blending layer embeds vmessages from the routing layer in transport protocol messages. Incoming vmessages are extracted from the transport layer and passed to the routing layer. Messages can be identified by picking up a potential vmessage and start deciphering k_{p_N} using its private key $k_{h_N}^{-1}$. If decryption succeeds, a vmessage is found. Protocol features such as anonymity or redundancy do not rely on this level. This layer embeds messages within the transport layer in such a way that an adversary is no longer able to identify vmessages from regular transport layer messages. Good blending is achieved if transport layer censorship measurements such as application-level firewalls are unable to detect the difference between real-world messages and vmessages. In an ideal application, this applies to human and algorithmic censorship.

The routing layer is the mixer of the system. It processes messages extracted by the blending layer and is supported by the accounting layer. The routing layer processes vmessages by recombining payload with defined operations.

The accounting layer protects a node from being overloaded or misused. Every sender must first apply for an ephemeral identity which is limited by lifetime. Nodes must apply a proof-of-work to their messages to get quotas or ephemeral identities. The ephemeral identity is assigned with message and size transfer quotas.

2.2 Message Processing

We define a protocol block which has an inner block structure as shown in Fig 1.

These blocks are passed from node to node. Every block is protected by two symmetric keys key_{peer_N} (in short k_{p_N}) , key_{sender_N} (in short k_{s_N}) and the private part of an asymmetric host key $k_{host_N}^{-1}$ (in short $k_{h_N}^{-1}$). The public host key $k_{h_N}^{1}$ and both symmetric keys are known to the builder of the routing block structure.

The header is protected by the symmetric key k_{s_N} and is found in a preamble to the header protected by the receiving peer's private key $k_{h_N}^{-1}$. The key k_{s_N} is known to the routing block builder and the receiving node only. The receiving node obtains all vital information protected by this key. k_{p_N} is known

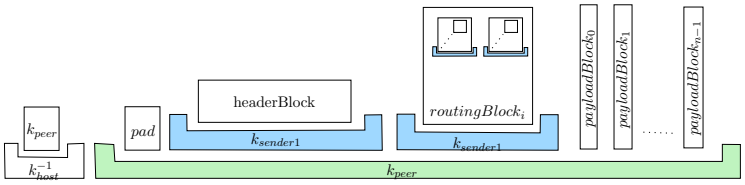

Fig. 1. Protocol block outline.

to two immediate peers and the builder of the routing block. The sending peer obtains k_{p_N} from the routing block, whereas the receiving peer acquires it in the *headerBlock*. The header block contains vital static information for the message disclosed to only one peer of the network. It is protected by key k_{s_N}. The operations within the routing block are designed in such a way that they allow a variance of message size without telling anyone which message part is used later.

All interactions are non-interactive. Interactive operations such as DC-nets do add more complexity to the system. Behavioral analysis can be used to identify interactive operations. This is one of the main reasons, why DC-nets are rarely used in this context. Theoretically, it is possible to reflect them as a single operation by calculating the answer and then broadcasting the answer. In practice, this fails due to the non-existence of efficient, reliable multicast networks. There are attempts to apply DC-nets to real protocols [8]. The protocol features multiple types of operations to enable mixing. Especially noteworthy is the *addRedundancy* operation. It supports the generation of decoy traffic in such a way that the generating node cannot tell decoy apart from real traffic. It is based on a modified Reed-Solomon redundancy function. The general inner workings are described in Fig 2.

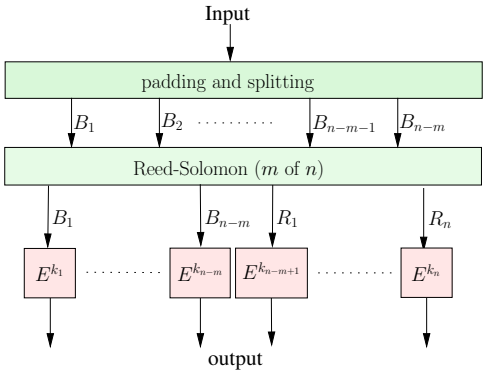

Fig. 2. AddRedundancy Operation

We define a function addRedundancy$_{n,m}(\mathbf{M}, k_1 \dots k_m)$ where M denotes the message, n the number of total output blocks, m the number of redundancy blocks whereas $m < n$, k the encryption key and scheme to be used, and bs_k the block size required to accommodate scheme and key size described by k. It is important to note that any set of blocks with the size of $d = n - m$ may be used to recover the full set of original data blocks. The n output blocks are encrypted with the keys k_1-k_n. The message is length-prefixed with a big-endian 64 bit unsigned integer number and padded in such a way that $8 + \text{len}(M) + \text{len}(padding) \bmod bs_k = 0$. As padding stream, we take the output of $\text{prng}_i\left(\left\lceil \frac{8+\text{len}(M)}{b_s} \right\rceil b_s n\right)$. The first 64 bytes of the message (padded with 0 if required) are taken as initializer i for the PRNG function.

Furthermore, the protocol offers support for onionized encryption, and block splitting and merging.

2.3 Message Building

Using previously defined operations, we may build a message path. This path is typically built by first assigning an identity set I_k where k denotes the target identity. I_k is a static set of n ephemeral identities $I_k\langle eI_1 \dots eI_n\rangle$ which are always used to communicate with k. This set may be enriched with further m ephemeral identities when sending. An identity set is replaced with a different one as soon as ephemeral identities expire. Therefore, we apply a new anonymity set unrelated to the old one with each new set of ephemeral identities. A full message graph including all traffic may have any complexity. Graphs feature partially independent routes from source to the target.

When building the message, it has to be ensured that all nodes in I_k obtain enough information to rebuild the message. If an adversary is capable of identifying the full message flow and knows all the operations applied to the message except for those on the entry and exit node, and at least a subset of $k = |I_{k_{uncompromised}}|$ where $k > 1$ exists then we are still at k-Anonymity as an absolute worst case scenario. Thus, we can prove that attacks, as described in [10], are of limited use.

3 Results

Our protocol may is a toolset for creating and sending anonymized messages. The degree of anonymity and redundancy is controlled when building the routing a building node must do this with care. The protocol has been proven to be very secure against common attacks. In our thesis[15] we analyze various kinds of attacks, such as illicit behaving nodes, hijacking of header and routing blocks, analysis on payload blocks, traffic replay, analysis of infrastructure, and analysis on operations. Results have shown that the protocol is very resistant against most kinds of attacks. We can prove the effectiveness of replay protection even when assuming misbehaving nodes. Hijacking a routing block allows at worst to reduce quotas of the routing block owner. Exchanging routing or header blocks

results in breaking of the messageś path. We can easily show the effectiveness of the tagging and bugging protection. A misbehaving node has no room to tag a message without compromising the message's integrity. A tagged message will be discarded at the first non-misbehaving node. If an interruption of a path is suspected, parts of the message may be obtained by the message block builder at any time. He may do this by introducing fixed diagnostic paths into a routing block, which we refer to as implicit diagnosis, or he may send a second message picking up a block of the message at a node to be tested. We refer to this as explicit diagnostic. Explicit diagnostics may be used as a kind of "receipt" from any node including but not limited to the terminal receiver of a message. Any block at any time of routing may be returned directly or indirectly to the original sender. The arrival of such a packet and content tells the sender at which point a message failed. If a diagnostic packet does not arrive, the routing block builder may build a diagnostic message picking up random packets on any suspected failing node.

4 Discussion

4.1 Comparison to Existing Systems

The following section gives a short comparison to existing systems. It shows that the solution defined in this paper covers a different approach and what problems are solved. It is important to note that this is not a ranking. It just outlines the differences between the system and shows where our system is different compared to existing solutions.

Researchers criticised TOR for several things. Firstly, it is easy attackable if a transported connection is not encrypted and authenticated. It relies on the trust in a centralized directory infrastructure. It is susceptible if more than $\approx 30\%$ of the nodes are controlled by an adversary as shown in [18]. Furthermore, timing analysis on entry and exit nodes are particularly easy because TOR is a low latency network[20,4]. Harvesting of nodes is possible (e.g., https://torstatus.blutmagie.de). Tor nodes are easily identifiable by traffic as shown in [24]. To avoid this detection TOR uses "pluggable transports." between dedicated node tuples. MessageVortex tries to address these problems in multiple ways. First, there is no central infrastructure which defies the trust problem. There are no entry or exit nodes as all participating members are routers at the same time. Therefore, all problems related to entry and exit nodes do not exist. There is no dedicated transport protocol making the presence of vmessages hard to detect. MessageVortex has several downsides compared to TOR. It is not suitable for real-time communication due to its asynchronous operation. It is furthermore a closed system and only participating members may use it.

$\mathcal{P}^5$ experienced so far very little in-depth analysis, as there is no precise protocol specification but only a rough outline available. This outline specifies the messaging and the crypto operations only. It claims to be peer to peer, which would result in NAT (Network Address Translation) circumvention technology. This technology usually relies, at least partially, on a central infrastructure (e.g.,

for hole punching). In contrast, MessageVortex protocol is peer to peer, but the transport layer is not. It misuses already existing infrastructure for transport. This makes it not susceptible to approaches against infrastructure unless our messages are identified and filtered. This may be corrected by applying different blending schemes for the transport layer. It furthermore removes the need for NAT hole punching and similar technologies.

I^2P has not attracted as much attention as TOR so far. It is thus hard to judge its real qualities. Unlike TOR, anonymity is not fully granted. Instead, pseudonymity is used. In [17] an attack specific to I^2P is presented. As I^2Ps security model is chosen based on IP addresses, the authors propose to use several cloud providers in different B-Class networks. By selectively flooding peers, an adversary may extract statistical information. The paper proposes an attack based on the heuristic performance-based peer selection. The main criticism of the paper was that the peer selection might be influenced by an adversary enabling him to recover data on a statistical base. MessageVortex does only allow a routing block builder to choose routes and amount of traffic. Due to the replay protection and the trust model, we do not rely on any node. We show in [15] that attacks on this level are ineffective.

Freenet is not a messaging but a distributed storage system. It has many useful features adapted by MessageVortex. Like in Freenet a MessageVortex node may deny being the owner of specific information unless the key for the respective ephemeral identity can be found on the system. As the key is only required for building routing blocks but not for message assembly and sending, this makes it a valuable feature comparable to the deniability of Freenet.

5 Conclusion

The MessageVortex protocol outlined in the previous sections does not solve all privacy issues which might arise. Furthermore, it is complicated to implement and involves a considerable amount of bookkeeping at runtime which is left to the sender of a message and the mixing nodes.

On the positive side, we have a new protocol which addresses privacy in a holistic approach leaving minimal attack surface. If handled with appropriate care by the sender and receiver, the protocol allows a sender-controlled, high degree amount of anonymity. Message paths are diagnosable, may be built redundant and do not build on the trust of any third party systems including all involved mixes except the senders and receivers. Even closed group communication or broadcasting to multiple identities involving a specific subset of mixes is possible if desired by the sender.

In [15] we show that the protocol is very secure. It is hard to block as messages may be redundant, hard to identify as messages are covered within message flows which may not be blocked without a massive impact on existing systems. It is hard to apply censorship in a real-world scenario as messages are tough to detect.

MessageVortex has some flaws which must be outlined. We always considered algorithmic censorship. If human censorship is applied, we must assume that

at least some of the messages are being identified as potential MessageVortex messages. If we assume a white-listing, human, censoring adversary (everything which is not identified by a human as compliant is censored) we must conclude that at least some messages will fail to be delivered. Some of the participating transport nodes may be identified and blocked. These flaws may be compensated with redundancy in message transmission. Messages transported by MessageVortex generate vast amounts of decoy traffic. Unlike other systems which control decoy traffic on a "per peer" base, MessageVortex does not dynamically reduce decoy traffic as decoy traffic is not identifiable. This results in a considerable traffic overhead.

References

1. Barbera, M.V., Kemerlis, V.P., Pappas, V., Keromytis, A.: CellFlood: Attacking Tor onion routers on the cheap. In: Proceedings of ESORICS 2013 (Sep 2013), http://www.cs.columbia.edu/~vpk/papers/cellflood.esorics13.pdf
2. Biryukov, A., Pustogarov, I., Weinmann, R.P.: TorScan: Tracing long-lived connections and differential scanning attacks. In: Proceedings of the European Symposium Research Computer Security - ESORICS'12. Springer (Sep 2012), http://freehaven.net/anonbib/papers/torscan-esorics2012.pdf
3. Biryukov, A., Pustogarov, I., Weinmann, R.P.: Trawling for tor hidden services: Detection, measurement, deanonymization. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy (May 2013), http://www.ieee-security.org/TC/SP2013/papers/4977a080.pdf
4. Chakravarty, S., Stavrou, A., Keromytis, A.D.: Traffic analysis against low-latency anonymity networks using available bandwidth estimation. In: Proceedings of the European Symposium Research Computer Security - ESORICS'10. Springer (September 2010), http://www.cs.columbia.edu/~sc2516/papers/chakravartyTA.pdf
5. Chaum, D.: Untraceable electronic mail, return, addresses, and digital pseudonyms. Communications of the ACM (1981), http://www.cs.utexas.edu/~shmat/courses/cs395t_fall04/chaum81.pdf
6. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology **1**, 65–75 (1988), http://www.cs.ucsb.edu/~ravenben/classes/595n-s07/papers/dcnet-jcrypt88.pdf
7. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A distributed anonymous information storage and retrieval system. In: Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability. pp. 46–66 (Jul 2000), https://freenetproject.org/
8. Corrigan-Gibbs, H., Ford, B.: Dissent: Accountable anonymous group messaging. In: Proceedings of the 17th ACM Conference on Computer and Communications Security. pp. 340–350. CCS '10, ACM, New York, NY, USA (2010). https://doi.org/10.1145/1866307.1866346, http://doi.acm.org/10.1145/1866307.1866346
9. Danezis, G., Dingledine, R., Mathewson, N.: Mixminion: Design of a type iii anonymous remailer protocol. In: Proceedings of the 2003 IEEE Symposium on Security and Privacy. pp. 2–15 (May 2003), http://mixminion.net/minion-design.pdf
10. Danezis, G., Serjantov, A.: Statistical disclosure or intersection attacks on anonymity systems. In: Proceedings of 6th Information Hiding Workshop (IH 2004).

LNCS (May 2004). https://doi.org/10.1007/978-3-540-30114-1-21, http://www.cl.cam.ac.uk/~aas23/papers_aas/PoolSDA2.ps
11. Dingledine, R., Freedman, M.J., Molnar, D.: The free haven project: Distributed anonymous storage service. In: Federrath, H. (ed.) Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability. Springer-Verlag, LNCS 2009 (Jul 2000), http://freehaven.net/doc/berk/freehaven-berk.ps
12. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (Aug 2004), http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA465464
13. Goel, S., Robson, M., Polte, M., Sirer, E.G.: Herbivore: A scalable and efficient protocol for anonymous communication. Tech. Rep. 2003-1890, Cornell University, Ithaca, NY (Feb 2003), http://www.cs.cornell.edu/People/egs/papers/herbivore-tr.pdf
14. Gülcü, C., Tsudik, G.: Mixing E-mail with Babel. In: Proceedings of the Network and Distributed Security Symposium - NDSS '96. pp. 2–16. IEEE (Feb 1996), http://citeseer.nj.nec.com/2254.html
15. Gwerder, M.: Messagevortex – transport independent messaging anonymous to third parties (Dec 2017), PhD thesis writing in progress
16. Gwerder, M.: Messagevortex protocol. IETF (2018), https://messagevortex.net/documentation.html
17. Herrmann, M., Grothoff, C.: Privacy implications of performance-based peer selection by onion routers: A real-world case study using i2p. In: Proceedings of the 11th Privacy Enhancing Technologies Symposium (PETS 2011) (Jul 2011), http://freehaven.net/anonbib/papers/pets2011/p9-herrmann.pdf
18. Jansen, R., Tschorsch, F., Johnson, A., Scheuermann, B.: The sniper attack: Anonymously deanonymizing and disabling the tor network. Tech. rep., DTIC Document (2014)
19. Möller, U., Cottrell, L., Palfrader, P., Sassaman, L.: Mixmaster Protocol — Version 2. IETF Internet Draft (Jul 2003), http://tools.ietf.org/pdf/draft-sassaman-mixmaster-03.pdf
20. Murdoch, S.J., Danezis, G.: Low-cost traffic analysis of Tor. In: Proceedings of the 2005 IEEE Symposium on Security and Privacy. IEEE CS (May 2005), http://www.cl.cam.ac.uk/users/sjm217/papers/oakland05torta.pdf
21. Øverlier, L., Syverson, P.: Locating hidden servers. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy. IEEE CS (May 2006), http://tor-svn.freehaven.net/anonbib/cache/hs-attack06.pdf
22. Pfitzmann, A., Hansen, M.: A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf (Aug 2010), http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, v0.34
23. Sherwood, R., Bhattacharjee, B., Srinivasan, A.: P5: A protocol for scalable anonymous communication. Journal of Computer Security **13**(6), 839–876 (2005), http://www.cs.umd.edu/projects/p5/
24. Winter, P., Lindskog, S.: How the great firewall of china is blocking tor. In: Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI 2012) (Aug 2012), https://www.usenix.org/system/files/conference/foci12/foci12-final2.pdf