Advance Security Project Title

Kelly Nicole Kaoudis
kaoudis@colorado.edu

ABSTRACT

We describe *some cool name here*, a distributed anonymous filesharing network based on insight from studying anonymity-providing services.

First, we present an introduction to and justification of the first two pillars of our design approach: security built in from the start at all possible levels, and realizing the implications of usability (or lack thereof) for the future developer and especially the user.

We detail some relevant portions of the related literature and what we have learned from those systems' failures and successes. Our work originated from an in-depth study of the construction of the Tor network, its performance, and interaction between its clients and its hidden services, but we do not use Tor itself as the foundation for our service. We justify this decision with Tor design documents, metrics taken from our prior study of Tor, and the stated wishes of the users and maintainers of the Tor network. We gained valuable insight into how a practical, widespread implementation of the onion protocol might succeed or fail.

We chose to also take away design lessons from current implementations of peer-to-peer filesharing systems, distributed databases such as Kademlia and Pastry, and two operation (GET/PUT) key-value stores. More recently we have begun considering implications of Byzantine failure and other potential faults for our system. In light of these concerns, the third pillar of our system is simplicity. This means reducing the number of technologies and techniques used, without compromising our other priorities, characteristic security and usability.

The fourth pillar of our completed system design will be modularity, as in the familiar Unix philosophy. Each element should work as a cog in the greater system without disrupting the user's experience of the system as a seamless entity. Components will be independently testable and secured, composable, and re-usable as needed as stand-alone modules in future work.

We detail the implementation to date and possible issues in our continued execution of our design. We conclude with analysis of possible threats to our system and design as it currently stands, and a brief description of the continuation of work on *some cool name here* in the coming months.

1. INTRODUCTION

Alice should be able to transfer her important documents to Bob without allowing anyone else knowledge of what is being transferred or where the documents are located at any point during the process. Neither the adversary nor Bob should know Alice's geolocation or IP. Neither Alice nor the adversary should know Bob's geolocation or IP.

The First Amendment to the United States Constitution is composed of clauses listing certain liberties that Americans have come to expect in all aspects of daily life. Among these are the prohibition of laws abridging the freedom of speech and interfering with the right to peaceably assemble. While the Internet is a multinational, borderless entity, it originated in the United States. Thus we believe that some of the morals and expectations of the American people have become deeply interwoven in the social fabric of the Internet. This is underscored by the worldwide availability of systems such as Tor [8], Freenet [4], I2P [19], and other anonymizing systems which claim to allow their users to transfer information without having to divulge anything they do not wish to.

As anyone who has tried the Tor system before knows, it is generally slower to use than the open web. We argue that this is due more to an imbalance between users and contributors than to overhead from the anonymity precautions.

Due (we hypothesize) to the Tor system's notoriety in mainstream media and additionally the general public's lack of understanding of what Tor is and how it works, the percentage of those using Tor and similar systems for anonymity who also contribute back by running a relay or an exit node or improving the codebase is much smaller than the number of people who want to use the service.

Furthermore it has been shown [3] using the Cisco traffic analysis tool Netflow that it is possible to uncloak around 81% of Tor users, especially in cases where the users did not have enough deep knowledge of the inner workings of the service. While we strongly advocate for an improved Tor, since it is already in widespread use, we propose a single-purpose partial alternative that addresses some of Tor's limitations.

We believe further exploration in this space is necessary and that anonymity is not a solved problem, especially given the number of possible attacks on Tor [11] [14] [2] [5] em-

ployed lately by governments and other entities.

Many entities have attempted to create a reasonably reliable, secure method for information transfer and have partially succeeded, standing upon the shoulders of those who tried and did not succeed quite so well previously. We propose yet another partial solution.

2. BACKGROUND AND RELATED WORK

The Tor system [8] is a mainly volunteer-run project claiming to provide more security than its users would have otherwise conducting business on the open web.

Tor offers users the ability to set up rendezvous points known as "hidden services", or alternatives to sites on the open web offering anything servable over a SOCKS proxy. While the idea is that operating such a service would be anonymous in that the operator's IP address and physical location should never be revealed, it has been proven that this assumption is flawed in some cases [5]. These hidden services, by way of a "hidden service descriptor", which consists of the service's public key and a summary of its introduction points, advertise themselves in a globally available distributed hash table directory. The descriptor is what's first found when a user types the service's DESC.onion address into their Tor browser's URL bar. The "DESC" is a base-32 encoding of a 10-octet hash of the service's public key.

Tor uses an incremental path-building design, which means that each successive hop in a circuit (a path between a user's Tor client's entry point into the network and her destination) has to negotiate its own session keys. Tor multiplexes many TCP streams on a single circuit to allow for a noisier background and also provide more efficiency than just one stream on a circuit would have. Therefore the next step from the client point of view in navigation to DESC.onion once the service descriptor has been fetched from a randomly chosen hidden service directory is as follows. The client picks one of the service's introduction (rendezvous) points, and establishes a rendezvous circuit which will facilitate connection to that introduction point. Once the introduction point has been established by the client, the *service* builds itself a new Tor circuit ending at that point as well. The rendezvous point authenticates and relays cells between the client circuit and the server circuit.

Our aim is to provide a service that facilitates anonymous, distributed sharing of encrypted information in a peer-to-peer fashion. The Tor FAQ [20] states that filesharing is not desired within the Tor network, especially using P2P protocols, which are not anonymized by Tor and SOCKS proxies in general, so we chose not to create another layer on top of Tor. Not implementing our service over Tor gives us the freedom to use UDP at the transport layer where desired and also to employ specially tailored peer-to-peer protocols for communication among our "directory servers" and also for uploading and downloading files from the network.

Similarly to Tor, I2P [12] is an anonymous network designed to sit below typical internet functions such as email, HTTP, file sharing, and so on, though it is message-based and one must use libraries which allow TCP and UDP traffic on top of it if desired. Like Tor, I2P encrypts all communication end-to-end. I2P also employs El Gamal for eepSite to eepSite dialogue where Tor uses RSA for hidden service to hidden service communication, and operates over a different routing scheme: "garlic" to Tor's "onion".

The general public expects the convenience of being able to launch any sort of application with little fuss. Additionally, popular virtual private storage / storage as a service providers such as Dropbox and Apple's iCloud do nothing to dissuade the general belief that properly securing data is trivial, or that one's data will be secure once entrusted to the service provider.

We argue that neither of these suppositions are currently true, especially in light of the high-profile breaches [13] of such services that have occurred in the last few months, spreading the 'private' data of public figures across the internet. However, if a service requires extra effort and time compared to 'the usual', it is unlikely that most human beings will put in that effort even if it is understood that they *should* (e.g. avoiding dentist appointments, and similar sociological phenomena).

Essentially, we wish to balance usability with privacy and create a service that is both simple to use and as secure as we can make it.

3. THREAT MODEL AND PREMISE

We absolutely do not advocate for those with no use case for such information outside personal privacy and data safety to spend their time in acquisition of in-depth working knowledge of networks and cryptography as a prerequisite for simply existing in the modern world. While such knowledge may become more widespread over time, as a simple example of the current situation, the author's grandmother is not very likely to spend significant time studying the particulars of TCP/IP when that time could be invested in her garden and great-grandchildren.

The connected world should not have to become computer scientists and network engineers if they do not otherwise wish to, simply to avoid being hacked.

Since the general expectation is already that data is secure once entrusted to the cloud, we propose a simple system to build on what has come before and better fulfill this supposition.

Given that Snowden supposes the governmental adversary is capable of at least one trillion hashes per second [9], we do not at this time include such an adversary in our model.

Our idea is therefore to provide an accessible service that allows the general public to share files once (or multiple times) using at least some level of encryption and randomization to keep the adversary, Eve, from having knowledge of the mapping between *any* users' personal information such as location and real name, and the encrypted data temporarily stored within the service in a randomized fashion.

4. SYSTEM ARCHITECTURE

Our system has two main parts: a public key,value store analogous to Tor relay directories, and a (secure) temporary data rendezvous point [7] [16] which would likely be a thin layer of encryption and randomization for data over storage space located on SaaS platforms and contributed by volunteers or as a requirement for uploading data in the first place (i.e. the "pay to play" model). These two parts will both operate on the peer-to-peer model assumption that *data is only sent when it is asked for by the receiver* [1].

In direct contradiction to methods employed by Onionshare [15], Anonabox [6], and similar systems overlaid on the existing Tor system, we rolled our own underlying system which integrates only the features we really needed from Tor, mainly the hidden service/rendezvous point model, and the secrecy given by the latest iteration of onion routing.

Despite hope that Tor might one day be more robust (as it stands, it is one of the most well-known free anonymity services), we cannot count on volunteers signing up to support Tor soon enough or in large enough number to meet demand for anonymous information sharing that requires less bandwidth *and* additionally support the general public sharing files over Tor as over tracked torrent clients.

Alice wants to send a file to Bob. She adds herself to the network; she first registers an identifier [10] (which could also be a cryptographic signature) with her client application. A .onion address [17] (that isn't currently in use) is calculated. The hash of the file she would like to send is also calculated. Her .onion address and the hash of the file she would like to send are PUT over the onion protocol as values with her chosen identifier and the recipient's identifier concatenated as the key (i.e., PUT$\langle id_A, id_B, .\text{onion}_A, \text{hash}\rangle$) to our key,value store.

The store chooses a destination point from its list of currently available rendezvous(es) and returns (over the onion protocol/over Tor) a tuple containing the rendezvous' id, .onion, and a reasonably current estimate of service statistics such as free space on that particular rendezvous: PUT$\langle id_R, .\text{onion}_R, \text{MB}\rangle$.

Alice (or the store?) issues PUT$\langle id_A, id_B, .\text{onion}_R, hash\rangle$ to the rendezvous. The rendezvous will then GET the file from Alice via a P2P connection. Ideally, only Bob, who has the ability to generate id_B and also knows id_A by virtue of having access to the key,value store (possibly he might know id_A beforehand), will be able to retrieve the file marked with both from the rendezvous point via untracked peer-to-peer download.

If we would like the file Alice has uploaded to the rendezvous point to persist for other downloads, we propose adding another value to Alice's initial PUT tuple, something analogous to a TTL value that gets set to "infinite" or decremented with each download. Once the TTL is zero, the rendezvous may write over the memory allocated to the file with random data, and then report that space as free, or as an alternative it will simply report that space as free and write over it in the course of hosting other anonymous P2P downloads. If Alice chooses a TTL greater than 1 in preparing her information for upload, perhaps she contatenates a dummy id_B that multiple people have the ability to replicate.

When Bob wants to retrieve the file Alice has left for him, he issues a GET$\langle id_A, id_B\rangle$to the key,value store. The store returns the .onion where the file is located, and the hash/checksum of the file. Bob now is armed with all the information he really needs to get his file.

Bob opens a GET$\langle id_A, id_B$, hash $\rangle$ on the .onion address of the rendezvous, which starts a peer-to-peer download from the rendezvous to Bob of the encrypted file. As Bob reaches certain points in the file, he checks against his hash to ensure he's got it all correctly. Potentially the rendezvous point already did the same when Alice was uploading the file as an integrity check. This also would eliminate the possibility of Alice sending a file to the rendezvous point that does not match the hash she already registered to the network.

Rendezvous points will mostly be run by volunteers who use the service regularly. This model could easily be extended to multiple stores which are updated through P2P every time someone uploads their information tuple to the store most convenient to them. A multiple-store model would provide both greater fault tolerance and additionally reduce the number of hops needed to query the network, hopefully adding efficiency. We also propose a study on what proportion of rendezvous points to regular users to key,value stores would make this architecture most efficient.

5. CONTROL INTERFACE

We propose a dual model for the client interface: a minimal browser extension that will work on both ordinary Firefox and the Tor browser bundle (which is based on Firefox), and a web site optimized for touchscreen (as well as keyboard) use. We do not wish to create actual client-side applications that run at a deeper layer than appended to the browser, if possible, to minimize any need for porting across platforms.

Rendezvous point operators should be able to see how much of their total space is being used, how many P2P connections they currently have open, and how much bandwidth that's taking up. Furthermore, the key,value store operator will have additional functionality (similar to that of the rendezvous operator) that allows him to keep an eye on the store and make sure it isn't being overwhelmed.

A snapshot of this information could be provided in JSON format from the key,value store as part of the response to adding oneself to the network in order to upload a file, or it could be available only upon request in keeping with the theme that no action of automated parts of this system shall happen without a single, authorized human request to facilitate it.

6. IMPLEMENTATION

7. CONCLUSIONS AND FUTURE WORK

We present some justification for and description of a service that allows anyone to protect the privacy of their data while sharing it as desired.

8. REFERENCES

[1] M. Belshe, R. Peon, and M. Thomson. Hypertext transfer protocol version 2. In *HTTPbis Working Group*, Fremont, CA, USA, July 30, 2014. IETF (Internet Engineering Task Force).

[2] T. Brewster. Swedish researchers uncover dirty tor exit relays. `http://www.techweekeurope.co.uk/workspace/malicious-tor-exit-relays-136828`.

[3] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis, and A. D. Keromytis. On the effectiveness of traffic analysis against anonymity networks using flow records. In *Proceedings of the 15th Passive and Active Measurements Conference (PAM '14)*, March 2014.

[4] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*, pages 46–66, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[5] N. Cubrilovic. Analyzing the fbi's explanation of how they located silk road. `https://www.nikcub.com/posts/analyzing-fbi-explanation-silk-road/`.

[6] C. Dewey. This debunked kickstarter project may be the biggest crowdfunding fail to date. *The Washington Post*, October 16 2014.

[7] R. Dingledine, N. Mathewson, and P. Syverson. Rendezvous points and hidden services. `https://svn.torproject.org/svn/projects/design-paper/tor-design.html#sec:rendezvous`.

[8] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[9] A. Greenberg and L. Poitras. These are the emails snowden sent to first introduce his epic nsa leaks. `http://www.wired.com/2014/10/snowdens-first-emails-to-poitras/`.

[10] S. Han, V. Liu, Q. Pu, S. Peter, T. E. Anderson, A. Krishnamurthy, and D. Wetherall. Expressive privacy control with pseudonyms. In D. M. Chiu, J. Wang, P. Barford, and S. Seshan, editors, *SIGCOMM*, pages 291–302. ACM, 2013.

[11] R. Jansen, F. Tschorsch, A. Johnson, and B. Scheuermann. The sniper attack: Anonymously deanonymizing and disabling the Tor network. In *Proceedings of the Network and Distributed Security Symposium - NDSS '14*. IEEE, February 2014.

[12] jrandom (Pseudonym). Invisible internet project (i2p) project overview. Design document, August 2003.

[13] N. Kerris and T. Muller. Apple media advisory: update to celebrity photo investigation. http://www.apple.com/pr/library/2014/09/02Apple-Media-Advisory.html.

[14] E. Kovacs. Operation against tor dark markets raises security concernts. http://www.securityweek.com/operation-against-tor-dark-markets-raises-security-concerns.

[15] M. F. Lee. Onionshare.

[16] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 85–100, New York, NY, USA, 2011. ACM.

[17] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, SP '97, Washington, DC, USA, 1997. IEEE Computer Society.

[18] Tails. Tails: the amnesiac incognito live system. https://tails.boum.org/.

[19] J. P. Timpanaro, I. Chrisment, and O. Festor. A bird's eye view on the i2p anonymous file-sharing environment. In *Proceedings of the 6th International Conference on Network and System Security*, Wu Yi Shan, China, November 2012.

[20] Tor. Tor project: Faq. https://www.torproject.org/docs/faq.html.en#FileSharing.