CPSC 490 Final Report

TorCoin: Proof-of-Bandwidth Altcoins for Compensating Relays

Miles Richardson
Yale University, New Haven, CT
miles.richardson@yale.edu

Abstract. This is the final report for CPSC 490. I collaborated with Mainak Ghosh on a project to submit to the HotPETS 2014 workshop in Amsterdam, which is focused on privacy enhancing technologies. The purpose of this paper is to serve as my individually written final report, which is orthogonal to the paper we submitted to the workshop. In it, I give an overview of our research, describe our results, offer possible applications of the research, provide a timeline of the semester, and discuss future improvements to the paper.

1 Overview

CPSC 490 is the “senior research” course in the Yale Computer Science department. The goal of the class is to accomplish some kind of independent research along with an advisor and optionally other group members. I collaborated with Professor Bryan Ford (advisor) and Mainak Ghosh (partner) to write a paper called “A TorPath to TorCoin: Proof-of-Bandwidth Altcoins for Compensating Relays”, introducing a solution to the problem of incentivizing Tor relays to provide bandwidth to the Tor network, which suffers from notoriously slow speeds.

We successfully submitted the paper to the 2014 HotPETS workshop, which is the workshop for “hot topics in privacy” at the Privacy Enhancing Technologies Symposium (PETS) in Amsterdam this July. We think we succeeded in writing a paper that introduced two legitimately novel concepts.

Since our main deliverable for the term was the paper itself, but CPSC 490 requires Mainak and I to submit separate reports, this report is orthogonal to our finished paper. I doubt many people will ever read this, but if you are one of the lucky few, my goal is to enlighten you as to what we actually did this term, why it’s important, and where we can go from here.

I really enjoyed working with Mainak and Professor Ford, and I hope to continue this research next term. I learned more in this research than I have in any other class. I was exposed to much more of Academia than I had been before, and I’m glad I now have a solid understanding of the process. Now I understand what’s required of a good academic paper, which makes me a much more critical reader of them. Although I plan to go “into industry” after graduation, this term
has given me an appreciation for Academia and I can see it becoming something I’m involved in post graduation, in addition to my industrial contributions.

I want to thank Professor Ford and Mainak for their help throughout the semester. I really enjoyed this project and hopefully we can continue to produce quality research together. Now, enjoy the final report!

2 Problem Statement

2.1 Why is Internet Anonymity Important?

The past decade has seen unprecedented empowerment of global citizens. Never before has it been so easy to share, communicate, and collaborate. Less than a century ago, sending a simple photo from China to the United States took weeks or months. Now, with the press of a button on a mobile phone, a single person in China can distribute a video to millions across the globe in a matter of seconds. This movement is catching authoritarian governments off-guard, because the Internet allows dissent to spread virally throughout populations. The Arab Spring exemplified government resistance to mass dissent. In an attempt to stop it, the governments employed heavy-handed measures like blocking social media. Fortunately, anonymity technologies rendered them fruitless. As such, improving anonymity technologies remains a subject of utmost importance.

2.2 Tor provides anonymity and censorship resistance.

Anonymity ensures that power stays in the hands of citizens, not the government. Specifically, the Tor project’s routing protocol enables anonymity through a special onion routing protocol. It routes requests through thousands of volunteer nodes such that no node or outside observer can de-anonymize the request. This protocol ensures anonymity between all nodes on the network, and incidentally allows users to circumvent censorship because no entity can know their destinations. Tor is a great tool for ensuring anonymity and resisting censorship.

2.3 Tor is great, but slow.

Tor is slow, as a result of oversubscription to the Tor network. Only one Tor network exists, and it depends on volunteers to supply server hardware and bandwidth. Unfortunately, Tor is too popular! More people want to use it than the servers can support. So, essentially, the Tor network suffers from a shortage of relay servers. The obvious explanation for the shortage is a lack of incentive to supply the servers. Volunteers actually pay a cost to assume a risk. They supply expensive bandwidth and servers, in exchange for nothing except the fear of an FBI battering ram smashing through their doors. This lack of economic incentive represents a fundamental flaw in the Tor network. Tor will never become mainstream without enough servers to support more users, and it seems there are not enough willing volunteers to provide sufficient servers. So we need to introduce an economic incentive to hosting Tor relay servers.

2.4 The Problem of Incentivizing Tor Relays

At a basic level, an incentive system must retain anonymity but have the ability to verifiably measure bandwidth and reliably distribute payment to the nodes
that provide it. The system must be resilient to adversaries attempting to identify clients, or fake bandwidth transfer. Effectively, the problem of incentivizing Tor relays is how to verify bandwidth transfer without compromising anonymity.

3 Summary of Research

This section serves as a summary of our research. Some is copied from our research paper. For more details, please read the paper itself.

3.1 Brief Description of Tor

Tor anonymizes web traffic by a method called “onion routing”, which routes traffic through a circuit of three relays (entry, middle, exit) in such a way that no relay on the circuit can identify the source or destination of a packet (except obviously the exit relay can identify the destination).

TorCoin is a method for compensating those relays. TorPath is a method of assigning circuits to clients so that the circuits can verifiably sign the TorCoins.

3.2 Research Overview

Our research goal was to identify a method of compensating Tor relays for transferring bandwidth, without compromising the “status-quo” of anonymity on the Tor network. Our research introduced TorCoin, an “altcoin” based on the Bitcoin protocol, that relays can “mine” by transferring bandwidth over a Tor circuit, and sell on any existing altcoin exchange. In order to verify that a TorCoin is legitimate, we introduced TorPath, a circuit assignment scheme that assigns clients and relays to circuits in such a way that each circuit can sign a Torcoin for authenticity. Anyone can verify the legitimacy of a Torcoin by simply checking the blockchain.

3.3 Key Technical Challenges

We might envision a naive bandwidth-measurement scheme using blinded cryptographic tokens to signify bandwidth transfer. Suppose in this scheme, a client is able to give each relay a token for the amount of bandwidth it provides. Relays are then able to convert these client-signed tokens into some form of an incentive. Such a scheme would be vulnerable to colluding groups of clients and relays, however, who can simply sign each other’s transfer tokens without actually transferring any bandwidth.

We attempt to counter this through the TorPath scheme, which restricts clients’ ability to choose their own path, ensuring that most paths include at least one non-colluding participant (the client or at least one of the three relays). Assignment servers bundle large groups of clients and relays into groups that collectively choose paths. Even in the relatively rare event that a path constructed this way consists entirely of colluding clients and relays, an upper bound on the number of coins each path can mint rate-limits potential loss to such entirely-colluding paths.
3.4 TorCoin

TorCoin is an alternative cryptocurrency, or altcoin, based on the BitCoin protocol. Unlike BitCoin, its proof-of-work scheme is bandwidth-intensive, rather than computationally-intensive. To “mine” a TorCoin, a relay transfers bandwidth over the Tor network. Since relays can sell TorCoin on any existing altcoin exchange, TorCoin effectively compensates them for contributing bandwidth to the network, and does not require clients to pay for access to it.

3.5 Algorithm Summary

The TorCoin algorithm allows relays sharing a circuit to “mine” TorCoins. It uses an encryption method similar to onion-hashing to get all circuit participants to agree when a TorCoin is mined. The algorithm is below:

1. Each client and relay creates a temporary key R and its hash $R' = Hash(R)$.
2. Every m Tor packets, the client sends a tuple $(\text{coin#}, R'_C)$, where $\text{coin#}$ is the number of Tor packets sent in this circuit.
3. Each relay adds its own temporary hash to the tuple and sends it to the next relay in the circuit.
4. The exit relay adds its own hash to the tuple to create the coin blob $B = (\text{coin#}, R'_C, R'_E, R'_M, R'_X)$.
5. The exit relay then signs the blob B to create $S_B^X$ and sends the tuple $(B, S_B^X, R_X)$ to the middle relay.
6. Each of the relays, $i$, signs the blob B to create $S_i^B$ and adds their signature and their temporary key to the tuple, and forwards it to the previous component in the circuit.
7. The client receives the tuple $\text{candidate} = (B, S_X^B, S_M^B, S_E^B, R_X, R_M, R_E, R_C)$ and verifies its correctness, thus proving bandwidth transfer.
8. To check if a TorCoin has been minted, the client checks if the lower-order bits of $Hash(\text{CS}_i, \text{candidate}) == 0$. If so, the client adds the tuple $(\text{CS}_i, \text{candidate})$ to the blockchain to mint a “TorCoin”.
9. The client then pays each relay in the circuit $\frac{1}{3}$ of the TorCoin.
After a circuit mines a TorCoin, all of the information necessary for verifying it is stored in the block chain. Any interested party can then verify that the route signature is authentic and refers to the correct group by referring to the public log. They can also verify that the blob B was signed by the correct relays by verifying the signatures, as well as verify that the temporary keys R actually hash to the claimed values $R'$.  

### 3.6 TorPath

The TorPath protocol assigns Tor circuits to clients, overriding Tor directory servers with assignment servers, which form decentralized consensus groups. The protocol guarantees that no participant on a circuit can identify all other participants, and that each circuit includes a publicly verifiable signature. We use TorPath to “sign” each TorCoin, so that anyone can verify a TorCoin by comparing its signature to a global circuit history.

#### Requirements

The TorPath protocol adheres to the following constraints:

- No client can generate its own circuit.
- Every circuit has a unique, publicly-verifiable signature.
- No client can know the circuit of another client.

#### Protocol Description

The protocol consists of three sequential steps:

1. **Group Initialization.** Assignment servers form a consensus group. Clients and relays provide public keys to join the group.
2. **Verifiable Shuffle.** The consensus group performs a decentralized, verifiable shuffle of all the public keys, resulting in a circuit assignment for each client.

3. **Path Lookup.** The assignment servers publish the result of the shuffle, such that each client can only identify its entry relay, and each relay can only identify its immediate neighbor(s).

**Drawbacks** The TorPath network is not backwards compatible with the existing Tor network, due to the fundamental differences of route assignment and access control, which are missing in Tor, but are necessary for the TorPath and TorCoin schemes to work.

However, any given physical relay or server can run both services at the same time. TorCoins will, of course, be generated only for measured TorCoin traffic.

### 3.7 High Level Architecture

![High level TorCoin system architecture for clients and relays.](image)

*Fig. 2.* High level TorCoin system architecture for clients and relays. A *TorPath Client* assigns Tor circuits to clients via the TorPath protocol, described in the next section. A *TorCoin Miner* mines TorCoins and stores them in a *TorCoin Wallet*. Each *Tor Client* and *Tor Relay* operates as usual, but on circuits assigned via the TorPath protocol.

TorCoin runs as a standalone service, and requires little modification of the core Tor codebase. Tor clients and relays operate as usual, but receive circuit assignments from *assignment servers* instead of directory servers. Separately, a *TorCoin Miner* on each machine mines TorCoins by monitoring the throughput of the local Tor TLS tunnel, and communicating with its circuit neighbors via the TorCoin algorithm.
3.8 Results

For the results portion of the paper, we experimented with the overhead generated by the TorCoin protocol. We built a lightweight simulation of the protocol using the Python Twisted Framework. The simulation launches a four nodes and passes around TorCoin packets. We found that although TorCoin introduces an acceptable amount of overhead, especially since the exact number is a tunable variable.

The total overhead from one round of successful TorCoin mining (i.e., one entire round trip from client through all the relays and back again) results in a total TorCoin packet overhead of 1752 bytes.

![Graph showing TorCoin packet overhead](image)

**Fig. 3.** TorCoin packet overhead. The TorCoin algorithm only runs after every $m$ Tor packets have been sent, and it is a tunable number.

Each round of TorCoin generation and verification happens only after $m$ Tor packets have been sent. Each standard Tor cell is 514 bytes long, so each round trip on the network requires transmission of $514 \times 6 = 3084$ bytes. Thus, if $m \geq 10$, the TorCoin protocol overhead is around 5%. The value of $m$ can be calibrated in further experimentation and as needed in order to achieve the sweet-spot of transmission efficiency and incentive maximization for relay providers.

It is possible for the system to tune the value of $m$ to decrease during times of high usage to incentivise relay providers to temporarily provide more servers to the network.

4 Potential Applications

TorCoin has many potential applications, outside of simply monetizing the existing Tor network, which would require the entire network to adopt the Tor-
Path protocol. In this section I will discuss two of them: 1) a way to use TorPath to decrease the latency of Tor Hidden Services (.onion websites), and 2) a way to bring TorCoin to market and generate revenue from it.

4.1 Decrease Latency of Tor Hidden Services

TorPath enables a single 3-relay circuit between Tor clients and hidden services (.onion). Currently, a Tor Hidden Service (.onion) connection requires two separate 3-relay circuits, from both the client and server, so that they cannot identify each other. With TorPath, this connection only requires a single 3-relay circuit, because the middle relays of the circuit are collectively chosen. So, any TorPath middle relay will protect the client from the hidden service, and vice versa. This significantly increases throughput of the .onion network, by halving the number of relays required for its operation.

4.2 Privacy Service Providers

The best research has far-reaching benefits, beyond the resulting paper. After all, what’s the point of researching something if it generates no practical results, and nobody uses it? I am interested in bringing this Torcoin research to market, and here I present a way to do it.

TorCoin can be brought to market by leveraging its biggest drawback, which is that it requires independent Tor networks to form, where each relay, client, and assignment server runs the TorCoin software on top of the traditional Tor software. Convincing the Tor community to adopt the TorCoin software is unrealistic. Instead, I propose a radical rethinking of the economic structure of the Tor anonymity network. Specifically, I introduce the idea of privacy service providers (PSPs), analogous to Internet service providers (ISPs) in current nomenclature. A PSP charges clients for access to a scalable, fast, independent Tor network.

A PSP abstracts away the entire notion of Torcoins, Torpaths, and the Tor network. Effectively, a PSP is simply a Torcoin exchange, allowing clients and relays to trade Torcoins with each other, and taking a percent of each transaction. It sets a specific exchange rate of Torcoins so that it can charge clients for access to the network per-mbit of bandwidth, then use that money to pay relays for the Torcoins they mine. This way, all the clients need to do is pay money and download software. All relays need to do is download software, and receive money. They don’t need to worry about selling the Torcoins themselves.

None of this trading has to happen manually. It can all be automated by specific software on the client and relay nodes, provided by the PSP. See the figure below for a possible software architecture. In this model, relays and clients have an automatic TorCoin trader to interact with the exchange. After installing the software, clients can use the secondary anonymity network in the same way they may already use the Tor network.

The PSP model radically alters the current notion of the Tor network, but it is also the cleanest, most frictionless way to bring Torcoin to market. It requires no additional work from clients or relays, and provides a way for clients to pay
Fig. 4. High level overview of the software architecture required for privacy service providers to charge for access to an independent Tor network. In this scheme, each PSP acts as a Torcoin exchange, providing liquidity by paying relays real currency for Torcoins, using money that clients pay for access to the network. Relays and clients install an automated TorCoin Trader to trade TorCoins over the exchange.

for fast access to real privacy, and relays a way to earn money by transferring bandwidth.

5 Contributions and Timeline of Semester

Mainak and I worked together on this project, and a lot of our contributions were the same. But CPSC 490 requires specific individual contributions, so we made sure that some parts of the project were done purely by one of us. Specifically, Mainak wrote half of the code, and I wrote the other half. For the paper, I was responsible for all diagramming and a specific set of sections, and Mainak was responsible for another set. Then we edited each other’s work. We definitely divided the work 50/50, and I think both of us learned way more in CPSC 490 than we have in any other class. I’m really glad we had this experience, and both Mainak and Professor Ford have been a pleasure to work with.

5.1 Finding a Friend (First week)

At the beginning of the semester, I was working alone on this project. I knew that I wanted to research Tor, and originally I wanted to work on an algorithm for groups of nodes to share the addresses of relays in such a way that no single node had access to the whole list. The idea was that a hostile state actor, like China, could obtain the list of Tor relays and simply block outgoing connections to any of them. A method of passing around this list without anyone having access to the entirety of it would guarantee that some nodes would not be blockable by the
government. This is an interesting problem and one that has been researched, and at some point I may come back to it.

As I was developing that idea, Professor Ford introduced me to Mainak, who was interested in doing research on Bitcoin. We had a brief discussion and realized we had many of the same interests. So we set to work on finding a way to use Bitcoin as a method of compensating Tor relays.

5.2 Developing the Idea (Next two weeks)

Our first iteration of the Torcoin algorithm was actually pretty good. We were on the 3rd floor of the Zoo, drawing circuits on whiteboards and writing various hash functions, when the algorithm just sort of “walked into our heads and said hello,” to quote J.K. Rowling. I think Professor Ford was surprised that we actually came up with an algorithm that could be worked into something legitimate.

When we discussed TorCoin with Professor Ford, he pointed out that the biggest gap was that we did not have a way to verify a Torcoin wasn’t produced by a colluding group of circuits. So he helped us form the idea of Torpath, which he had been kicking around as a method for DISSENT nodes to form groups.

5.3 Writing Some Code (Weeks prior to spring break)

Before spring break, we were focused on writing code to implement the TorCoin algorithm. Actually coding a workable system was probably an ambitious goal, but after a few sessions we completed a system for simulating the TorCoin algorithm, which we knew we could use for running experiments to generate data for the paper.

Our code was relatively simple. We used the Python Twisted framework to write code that could act as both a server and a client. When it received a Torcoin packet, it modified it as per the Torcoin algorithm and sent it to the next node. Depending on whether the Torcoin packet was being generated or verified, each node sent it either to the left or to the right. The client initiated and terminated the circuit.

In terms of coding contributions, Mainak and I split up the code so that we could each work on specific parts of it. I worked on the server code, and he worked on the client code and writing tests to run the experiments and measure the bandwidth overhead.

Once we had some experiment results, we brought them to Professor Ford and told him that we were interested in turning this product into a paper, and hopefully a publishable one. We started working on a rough draft just before Spring Break.

5.4 Collaborating with the NRL (3 weeks after spring break)

Over spring break, Mainak and I received an email from Professor Ford telling us that Rob Jansen, a “heavy-hitter” in the Tor community, was interested in collaborating with us on a paper for submission to the HotPETS workshop (which we ultimately submitted to). Rob works with the Naval Research Laboratory and
has published multiple papers researching Tor. Of course we were very excited by the idea of working with him, so we agreed.

After spring break, we had weekly phone calls with Rob for the following three weeks. Initially, we were very excited to be working with him. But we soon realized that he had barely read our proposal, and the paper was spiraling out of control. Instead of providing any novel contributions to the paper, Rob and his other collaborators were stringing together various ideas in a way that made no sense. On our weekly phone calls, Rob would mention a major problem with one of his schemes, and wonder how to solve it. Most of the time it was a problem not present in our proposal, and one that we could solve. He rarely listened to anything we had to say, and any of our writing was to be relegated to the appendix section.

With two weeks to the submission deadline, we realized that this paper was a complete mess, way longer than necessary, and would likely not be accepted to the HotPETS workshop. So we made the decision to break away from his paper.

5.5 Pulling Together a Paper (Final 2 weeks)

We had two weeks to pull together a paper submittable to the HotPETS workshop, and essentially had nothing because we could not incorporate any of the paper that Rob had worked on. During the last two weeks, Mainak and I worked for over 40 hours each to put together the paper, pulling multiple all nighters and enjoying lots of late night writing sessions in the Zoo. We went through about six different drafts with Professor Ford, and ultimately arrived on our final draft, which we are decently happy with, but could definitely be improved.

6 Future Improvements to the Paper

Currently the paper suffers from a lack of clarity. We originally focused on TorCoin/proof-of-bandwidth, then introduced TorPath to solve the problem of verifying TorCoins. But now, the structure of the paper emphasizes TorPath as the paper’s main contribution. This makes sense strategically because our TorPath writing is more developed, so we should focus on it.

When I imagine a complete paper, I see TorCoin taking the lead role over TorPath. After all, TorPath is only necessary because TorCoin needs it to sign coins. Yes, TorPath has stand-alone qualities – increased circuit anonymity, decreased .onion latency, etc – but these are orthogonal to the ability to verifiably sign TorCoins. I wonder if we should ditch the “TorPath” moniker all together, and simply describe its protocol as another component necessary for TorCoin to function. Emphasizing TorCoin would increase clarity of purpose for the reader.

Also, we need to address the massive drawback of TorPath: it requires independent Tor networks to form, where clients, relays, and servers all run the same TorCoin/TorPath software. This requirement is a big hurdle to adoption. If we describe TorPath independently of TorCoin, the orthogonal benefits to circuit anonymity do not warrant adopting a new Tor network. But if we introduce TorPath as enabling TorCoin, we can argue that the adoption hurdle is much smaller because TorCoins provide a monetary incentive for independent Tor networks to develop.
Currently we briefly mention this drawback at the end of the paper. But it’s such a huge drawback that it warrants more focus. That was why I tried to frame it as a “contribution” in the first draft of this paper. I wanted to come out guns blazing, with sentences like ”we introduce the idea of independent Tor networks funded by a proof-of-bandwidth Torcoin...” I realize this is an unabashedly overzealous approach, but I also think it’s radical enough to attract attention.

In the “privacy service provider” economic architecture, described in the “applications” section, “PSPs” run Torcoin exchanges that abstract the entire process of circuit assignment and Torcoin minting from clients and relays. They simply pay relays for bandwidth with money they receive from clients paying for access. To me, this economic architecture seems like the most frictionless approach to Torcoin adoption. It gives a reason to bring the research to market. If consumers demand privacy, and a company can provably sell it, Torcoin will come to fruition.

This summer I’ll be in Taiwan working on a VPN company with a partner who has experience in the area. A primary criticism of VPN privacy is the centralization of providers, i.e. there’s no guarantee that the VPN provider is not logging your traffic. Torcoin addresses this concern, by replacing VPNs with a Tor network that actually has enough resources to support its own traffic. In the long term vision for this company, I can see Torcoin replacing VPNs as a privacy mechanism.

Until then, this summer I’m focusing on bringing the VPN service to market so that I can create a revenue stream as soon as possible. But I want to continue developing this research because I can see a situation in which I could bring it to market, and backing it with peer-reviewed research would immediately increase credibility. I graduate in December, so I have one more term in which I could work on this while at Yale. This is definitely something I’m interested in doing.