Proof-of-Work as Anonymous Micropayment: Rewarding a Tor Relay

Alex Biryukov and Ivan Pustogarov

University of Luxembourg,
{alex.biryukov,ivan.pustogarov}@uni.lu

Abstract. In this paper we propose a new micropayments scheme which can be used to reward Tor relay operators. Tor clients do not pay Tor relays with electronic cash directly but submit proof of work shares which the relays can resubmit to a crypto-currency mining pool. Relays credit users who submit shares with tickets that can later be used to purchase improved service. Both shares and tickets when sent over Tor circuits are anonymous. The analysis of the crypto-currencies market prices shows that the proposed scheme can compensate significant part of Tor relay operator's expenses.

Keywords: Tor; Proof of Work; Crypto-currency; Micropayment; Mining pools

1 Introduction

Many open peer-to-peer systems rely on volunteers donating their resources in order to achieve acceptable level of Quality of Service. E.g. in file-sharing applications, latency and failure rate depends on the number of users sharing their resources. In overlay routing systems packet latency depends on relays donating their bandwidth. Many of these systems suffer from free-riding: users consume resources without donating anything back. Obviously, this rational behavior is motivated by that users don't want to degrade their own performance. While not a P2P network in the traditional sense as there is a clear separation between clients and relays, Tor network suffers from the same free-riding problems: only limited number of relays provide decent bandwidth while the client base is rather large.

A number of incentive techniques were proposed to mitigate selfish behaviour of clients for traditional P2P systems. The bottom line of many of them is that a client is incentivized to donate the same type of resources to the network as he consumes. Unfortunately for Tor such incentives are hardly applicable: the majority of Tor users reside behind ISP NAT's and firewalls and thus cannot be checked by Tor authorities for reachability which prevents them from appearing in the Tor Consensus. In fact, for Tor it might be even undesirable to allow very low bandwidth nodes to become a part of the network [2] (and many clients can provide only limited bandwidth).

Another alternative would be to use a cryptocurrency and make direct payments to Tor relay operators. Many cryptocurrencies are not anonymous however which is in conflict with Tor goals. In this paper we propose a method to reward Tor relays. This method is based on crypto-currencies but does not have to involve direct payments; it rather adopts a mining-pool approach: a Tor relay implements mining pool functionality and provides Tor clients with mining jobs. When a client finds the job which meets requested difficulty, he submits the share to the Tor relay and gets priority tickets in exchange. Tor relays can either join a mining pool and delegate jobs to Tor clients or can do solo-mining and try to solve a block. The proposed approach does not require a central bank or a secure bandwidth measurement mechanism. The proposed approach may also help to solve scalability problem. The more users join the Tor network and use "paid" services, the more profitable it becomes to run a relay, and the more relays are expected to join the network.

The rest of the paper is organized as follows. In section 2 we give an overview of previous proposals to incentivize contributing to Tor. In section 3 we provide the necessary background on Tor, Bitcoin, Altcoins, and mining pools. In section 4 we describe the details of our approach. Analysis of the method is given is section 5. Discussion in Section 6 concludes the paper.

2 Related Work

The necessity of developing robust and secure incentives to participate in Tor was first mentioned in the Tor design paper [9]. Since then a lot of research has been done in the area. In this section we provide a short survey of previous approaches. A good summary can also be found in [27].

Androualki et al. [3] propose that Tor clients use e-cash in order to pay relays for high-priority circuits. In their scheme coins are issued by a central bank. In the proposed approach, in order to prevent double spending a relay should deposit coins shortly after a user withdraws them. This creates potential timing linkability attacks. The scheme assumes that clients purchase digital coins from the bank using real money. This might create legal issues for Tor relay operators. Finally, introducing a bank makes another step towards Tor centralisation.

Ngan, Dingledine, and Wallach [22] propose that relays which provide good service to others are assigned a "Gold Star" flag by Tor authorities. A Gold Star relay's traffic is then given higher priority. As was mentioned in [22], the original anonymity set is divided into two: one for gold star users and one for the rest. This has a negative impact on anonymity. In addition, only people who run relays can get improved service.

Jansen, Hopper, and Kim [16] developed BRAIDS, a scheme in which users "pay" relays with tickets. Double spending is prevented by that tickets are "relay-specific". Tickets are produced by central bank using blind signatures and a limited amount of tickets is distributed to clients through Guard nodes. Relays can accumulate clients' tickets and thus receive better performance by having more tickets. The proposed method achieves desirable anonymity properties,

however it still uses a central bank for generating coins. Second, in order to prevent clients from unfairly increasing free ticket income by joining multiple nodes into the system, the ticket distribution is limited per IP address. This creates problems for users behind NAT which share the same IP address. Third, the problem of collusion between clients and tickets distributors is not mitigated completely.

In [17], Jansen, Johnson, and Syverson propose LIRA, an incentive scheme similar to BRAIDS where the problem of double spending is solved by using "relay-specific" tickets. In LIRA, tickets can be obtained either by buying them from a central bank or by simply guessing (the probability to guess a "correct" ticket is a tunable parameter). By providing bandwidth relays receive coins from the bank which they can exchange later for tickets. LIRA has two undesirable properties: (1) it relies on an external secure bandwidth measurement scheme; (2) a central bank is still used.

In [12], Jansen, Miller, Syverson, and Ford discuss an approach which borrows some ideas from BRAIDS and LIRA however they try to avoid reliance on a central bank. This is achieved by distributing bank functionality among several semi-trusted servers. Public accountability is achieved by using protocols from distributed digital crypto-currencies. While a step towards better decentralisation a new bank entity is still added to the system. In addition similar to LIRA, the proposed system relies on a external secure bandwidth measurement scheme.

Ghosh, Richardson, Ford, and Jansen [19] propose the concepts of TorCoin and TorPath. TorCoin is a crypto-currency which uses proof-of-bandwidth instead of proof-of-work in order to mint new coins. TorPath is a bandwidth measurement mechanism which tries to prevent colluding parties from minting coins without providing bandwidth to the network. Despite TorPath mechanism, the proposed scheme is still vulnerable to Sybil attacks which allow a colluding set of malicious servers and clients to generate coins without providing real bandwidth to the network. In addition TorPath requires that clients do not choose circuits by themselves but rather circuits are assigned by "assignment servers". It is a huge step towards network centralisation. Finally during circuit assignment, relay's bandwidth is not taken into account, thus making it easier for an attacker's relays to be chosen at both ends of a circuit (which has a negative impact on the anonymity).

The idea described in [23] is close in spirit to our scheme (though not directly related) and suggests that a client offers a portion of his computation power in exchange for a service.

3 Background

3.1 Tor Anonymity Network

Tor is a volunteer-based low-latency anonymity network built on ideas of onion routing. It allows users to browse Internet anonymously by forwarding their

traffic through a *circuit* of *Tor Relays* in such a way that each relay in the circuit knows only the immediate transmitter of the message and the immediate receiver of the message. Using Tor makes it more difficult for non-global adversaries[1] to trace Internet activity for TCP applications. At the time of writing Tor comprises of more than 6,000 relays while the number of clients is estimated to be more than 500,000 daily [28] (not counting the bots).

Anonymous circuits. In order to make a connection to a server through the Tor network, a client first downloads *Consensus* (the list of running Tor relays) from a small set of *Tor Authorities*. The client then chooses a path consisting of three relays and agrees on a Diffie-Hellman key with the first (*Entry*) node in the path. Then he completes a Diffie-Hellman key exchange handshake with the second *Middle* node by using the first node as a proxy. In this way the Middle node does not know the real initiator of the handshake. The user repeats the same procedure with the third (*Exit*) node but uses the chain of Entry and Middle nodes as proxies. When the client wants to connect to a server on the Internet, he packs his messages in fixed 512-bytes sized *cells* and encrypts each cells with the three keys. When the message travels along the circuit, each relay strips off one layer of encryption. This scheme allows to break linkability between the sender of the message and its destination.

Load balancing. In order to achieve better performance Tor implements a load balancing mechanism. Each relay in the Consensus is assigned a bandwidth weight and the probability for a relay to be chosen by a client is roughly proportional to that weight. In order to assign weights to relays, Tor authorities conduct active measurements by building two-hop circuits to specific URL's and measure download times. This way relays get clients depending on their current load.

Bandwidth management. In order to limit bandwidth usage, a Tor relay implements token bucket algorithm [26]. The relay's operator can specify the token rate (which specifies the average incoming/outgoing bandwidth usage) and the bucket size (which specifies the burst).

3.2 Bitcoin and Other Crypto-Currencies

Bitcoin is a decentralized digital currency and payment system which does not rely on a trusted issuing entity but rather on a peer-to-peer network with peers minting Bitcoins by brute-forcing double SHA-256 hash function. Two main components in Bitcoin are: (1) coins generation, and (2) double-spending prevention. Coins ownership and money transfers are implemented through elliptic curve public key cryptography: a party proves the ownership when he transfers a coin by signing it with his private key.

Money generation. Money generation in Bitcoin is solved by maintaining a public list of blocks, the *block chain*, which starts from the *genesis block*. Bitcoin participants constantly search for new valid blocks, once a new valid block is

[1] A global passive adversary is a type of adversary who can observe all the traffic in the network.

discovered the corresponding peer announces it and generates (and earns) new coins. Valid blocks are created by Bitcoin miners by providing *proofs of work* (PoW). The proof of work consists of finding a cryptographic hash value for a block of transactions which starts with a certain number of leading zero bits. Hash of the previous block is included into the new block, which results in chain of blocks. The Proof-of-Work difficulty (i.e. the required number of zero bits) is adjusted automatically by the network so that the network generates one block every 10 minutes on the average. In case of two forks of the blockchain, the longest[2] fork is adopted by the network.

Double spending. Double spending is prevented by that *transactions* are included into blocks and each peer in the network keeps its local copy of the blockchain. When a payee receives a transaction it checks the blockchain if the same coins were already spent previously. Once a transaction is buried under sufficient number of blocks it becomes computationally impractical to revert it. The very first transaction in each block is called *coinbase transaction* and is used for coins generation. A miner which generated a block puts the hash of his public key into this transaction.

Altcoins. Since Bitcoin's inception in 2009, a number of alternative currencies has appeared (called *Altcoins*). They all share the same basic principles of Bitcoin, but differ in PoW algorithms, difficulty adjustment rules, and amount of coins generated per block. Bitcoin along with Altcoins form the crypto currency market where they can be exchanged into the usual fiat currencies. In addition it is possible to pay directly with Bitcoin for a number of services. Currencies based on ASIC-resistant PoW functions from Password Hashing Competition (PHC) [24] have been recently proposed.

3.3 Mining-pools

Bitcoin mining is an arms race. Initially it was carried out by CPU. As Bitcoin was becoming more popular and new miners were entering the game, mining has moved to GPU. Then with the rise of the Bitcoin price, dedicated ASICs which provided orders of magnitude higher hash rates were developed. The probability to mine a block with a consumer-level GPU became very low at this stage. For miners without powerful dedicated hardware it takes prohibitively long time (years) before they can make a return. Such miners solve this problem by joining their resources in a mining pool. Participants of a mining pool all together generate blocks much faster and receive a portion of the block reward on a consistent basis.

Each miner in a mining pool tries to solve a block with a much lower than the original difficulty. Such simpler block is called a *share*. With some probability the share will also have a solution with the original difficulty in which case the pool mines a block. The block reward is then divided among the participants based on the work they contributed. There exist pools which track crypto-currencies

[2] In terms of difficulty.

market prices and automatically switch to mining the most profitable crypto-currency.

3.4 Partially blind signatures

Blind signature is a from of digital signature scheme that allows a user to get a signature on a message from a party without revealing the content of the message. The common usages include digital cash schemes and voting protocols. Blind signature schemes however do not allow the signer to include necessary information (e.g. expiration date of a digital coin) in the resulting signature. This limitation can be removed by using a *partially* blind signature scheme [1] in which a signer can explicitly include additional attributes (such as timestamps). A concrete scheme from [1] is described in Appendix B.

4 Proof-of-Work as payment for service

4.1 Design goals

The main objective of the proposed scheme is to compensate Tor relays for providing improved service and to encourage server operator's participation in the Tor network. In addition, we require the following properties. First, the scheme should not degrade the anonymity provided by Tor, i.e. it should not introduce new attack vectors. Second, it should not involve direct payments neither with fiat nor with crypto-currencies. The reason for this is that direct payment even with a digital currency like Bitcoin will reduce user privacy[3] and may become a strong psychological obstacle for adopting a scheme for ordinary users. Third, it should not rely on secure bandwidth measurement mechanisms. Fourth, it should not involve a central bank as in [16]. Sixth, the scheme should not require from users to run a Tor relay in order to get improved service. We analyse these properties in more detail in section 5.

4.2 System design

Tor users can get improved service from a Tor relay by producing proof-of-work and sending it to the relay over an anonymous Tor circuit. The relay can then forward this proof-of-work to a crypto-currency mining pool and earn coins. Users are rewarded by relay-specific *priority tickets* which can later be exchanged at the same relay for improved service (higher bandwidth or lower latencies). Tickets are issued by relays using blind signatures [5] and exchanged between users and relays over anonymous Tor circuits. Unlike [16] we do not use any bank entity and tickets are blind-signed by relays themselves.

Setup. In the setup phase a Tor relay first chooses a mining pool, the corresponding crypto-currencies and PoW algorithms (note that the relay can choose

[3] An option of payment via anonymous crypto-currency like ZeroCoin [21] will be discussed in Section 6.

a pool which automatically switches to the most profitable currencies). Second, the relay generates a public/private key pair which will be used in generation of priority tickets (this key pair should be different from the relay's onion and identity keys). The relay then includes this information into its descriptor. A client which plans to obtain improved service chooses relays which announce compatible PoW algorithms.

Protocol 1. Ticket Purchase: Client C obtains a priority ticket from relay R

1: $C \rightarrow R$: `SUBSCRIBE` message.
2: $R \rightarrow C$: `JOB` message.
3: C : start mining a share.
4: C : If share w is found, generate random number x and its hash $H(x)$.
5: $C \rightarrow R$: w, $H(x)$.
6: R : check w, if correct pass it to the mining pool.
7: $R \leftrightarrow C$: Generate partially blind signature S over $\{H(x), d\}$, where d is an assigned by the relay timestamp, which specifies the current day.
8: C: Keeps the ticket $T_R = \{S, d, x, H(x)\}$.

Purchasing priority tickets. A relay will provide improved service for clients in exchange for priority tickets. Priority tickets are relay-specific which means that by default they can only be used to purchase service from the relay which issued them (see Protocol 2 if ticket exchange is required). The protocol for client C to obtain a ticket from relay R is described in Protocol 1. Prior to execution of the protocol, the client establishes an anonymous Tor circuit to the relay. All communications are carried over this circuit, including (optionally) the future client traffic. Client C registers for a new mining job with relay R and the relay sends a reply in which it specifies the PoW algorithm, difficulty per share, and data sufficient to construct a share (steps 1–2). At step 3, the client starts solving a new share. At steps 4–5 (given that the client solved the share), the client generates a random value x and its hash $H(x)$ and sends the share to R. The relay verifies the share and produces a partially blind signature S over $H(x)$ with timestamp d as an added factor according to [1]. The tuple $T = \{S, d, x, H(x)\}$ is a priority ticket which the client can later exchange for the improved service. By reducing the granularity of the timestamp to just the current date makes all clients that got tickets on the same day undistinguishable.

Buying improved service. Every ticket that a client gets can be used to transmit cells with priority access during Δt seconds through the Tor relay which issued the ticket. In order to prevent double-spending, the relay should keep history of spent tickets. To limit the size of this database tickets should expire after e.g. 48 hours.

Priority access. We suggest using Hierarchical Token Bucket Algorithm [18] to provide improved quality of service for users with priority tickets, however other options exist [11]. HTB is a simple algorithm and it is a logical step from the currently employed by Tor Token Bucket algorithm. The priority access scheme

should allocate enough resources for "free" users so that people without funds to buy high-speed computers can still have reasonable QoS with Tor.

In Hierarchical Token Bucket the bandwidth is allocated to one or more classes, and when a class-allocated bandwidth is exceeded, it can temporarily "borrow" unused bandwidth from another class. Classes form a tree structure in which only leaves have corresponding packet queues. Each class C has associated guaranteed rate R_C, ceil rate $Ceil_C$, and priority p_C. Class C is guaranteed to have at least rate R_C. The rate of a parent class should not exceed the sum of the rates of its children classes. $Ceil_C$ specifies the maximum speed that class C can have by borrowing from its parent class. Classes with higher priorities borrow unused bandwidth first.

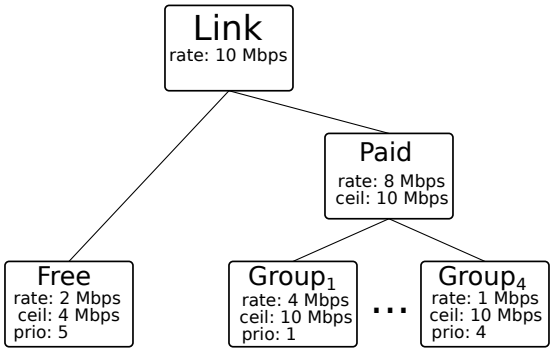

Fig. 1. Hierarchical Token Bucket example: the link of 10 Mbps is divided between paid and free services. Free circuits will share 2 Mpbs or 4 Mbps if there are no paid clients. Paid clients can get the whole capacity of 10 Mbps if there are no free users.

Consider an example in Figure 1 in which a relay is willing to provide up to 10 Mpbs for Paid and Free services in total. The guaranteed rate for Free service is 2 Mpbs; the total rate for Paid service is 8 Mbps which is later divided between different classes of users based on the number of tickets they pay. Consider two examples. In the first example a relay does not have any paid clients in which case the relay increases the bandwidth for Free service to 4 Mbps by borrowing from the Paid class. In the second example the relay has very few free clients which consume only 1 Mbps while classes $Group_1$ and $Group_4$ require 4.8 Mbps and 1.5 Mpbs respectively. In this case the higher-priority class $Group_1$ which normally pays for the rate of 4Mbps will be the first to take the needed 0.8 Mpbs after which the lower priority class $Group_4$ will take the remaining 0.2 Mbps.

Ticket exchange. So far in the proposed scheme a client gets tickets from the same relay R_1 for which he is working, and the tickets are valid at this relay only. Such scheme works best if the client provides proof-of-work simultaneously with sending his data over Tor. Assume now that a client pre-mined priority tickets with an intention to spend them later. He might become frustrated if at

the time when he decides to spend them relay R_1 is off-line. In such a case relay R_1 may team with a backup relay R_2 and ask it to accept its priority tickets. R_2 can later request payment from R_1 in crypto-coins or by redirecting his clients to mine for R_2. Protocol 2 describes how priority tickets issued to client C by relay R_1 can be spent at relay R_2. When relays R_1 and R_2 are both online they synchronise their databases of spent tickets.

Protocol 2. Ticket Exchange: C gets improved service at R_2 by providing a ticket issued by R_1

Client C obtained ticket $T_{R_1} = \{S_1, d, x, H(x)\}$ from relay R_1. R_2 is a backup relay for R_1

1: $C \rightarrow R_2 : T_{R_1}$
2: R_2 : verify signature S_1 and timestamp d.
3: R_2: If correct, register T_{R_1} as spent (sync this with R_1).
4: R_2 : If T_{R_1} is correct, provide priority access.
5: $R_2 \rightarrow R_1$: `PAYMENT_REQUEST` (Once every N served tickets).

Assume that client C has ticket $T_{R_1} = \{S_1, d, x, H(x)\}$ issued by relay R_1. The objective of the Protocol 2 is for the client to be able to get improved service from relay R_2 while preserving the following properties: (1) A colluding client and relay should not produce "free" tickets which can later be used at other relays; (2) Double spending of the same ticket at two different relays should be prevented.

"Free" tickets created by colluding client C and relay R_1 are avoided by that R_2 requests payment for each batch of N served tickets (either in crypto-coins or by delegating new mining work). We can envision that in practice relays R_1, R_2 might be run by the same operator or by two operators, who trust each other. In the second case the amount of trust can be regulated by the size of N. In case R_1 stops paying, relay R_2 will stop accepting its tickets. In order to prevent double-spending of the same ticket at relays R_1 and R_2 they should regularly synchronise their databases of spent tickets.

Mining strategies. The operator of a Tor relay which accepts PoW shares has two possibilities. First, he can decide to do solo-mining, by making his crypto-currency address a part of `JOB` messages sent to clients in the hope that one of the submitted shares will also solve a block. This strategy requires significant computational power at a large number of Tor clients. Second, the Tor relay operator may decide to ask for work from a large mining pool and then delegate this work to clients. The operator then resubmits the shares found by the clients to the mining pool. Note that the mining pool requests the relay to generate a share of difficulty lower than the current block's difficulty in the hope that one share will also solve the block. The Tor relay may use the same strategy towards Tor clients: it may request to generate PoW with difficulty lower than that indicated by the mining pool in the hope that a client's PoW will also solve

the mining pool's share. With this approach the Tor relay may regulate how many tickets are issued to different clients, proportional to their mining power.
Donations. Clients that just want to support Tor relays without requesting any bandwidth can submit shares without requesting anything back.
Implementation considerations. The scheme proposed in this paper requires several modifications to the Tor protocol and bundling Tor with crypto-currencies mining software. It introduces the following new Tor cells:

- `RELAY_MINING_REGISTER`[4]– by sending this message a user asks a Tor relay to send him mining jobs.
- `RELAY_MINING_JOB` – a Tor relay uses this message to send mining jobs to clients.
- `RELAY_TICKET` – used by Tor relays to (1) send material (blind signed) to clients for producing a priority ticket, (2) to notify backup relays about spent tickets; used by Tor clients to send priority tickets and request improved service from a relay.

In addition our scheme requires:

- Replacing currently used Token Bucket algorithm with Hierarchical Token Bucket algorithm.
- Implementing a partially blind signature module.
- Keeping track of spent tickets.
- Synchronizing a relay's spent tickets database with its backup relay.

At the client side, Tor should be bundled with a crypto-currency miner software (e.g. [4] or [7]). At the relay side, Tor should be bundled with both miner and mining pool software (e.g. [10]). Tor control port should also be extended to enable communication between Tor and mining software. Note that it is not necessary to develop new mining software, but rather bundle existing projects with Tor.

5 Analysis

5.1 Profitability

Motivation. According to the performance statistics maintained by the Tor project[5] [29], it takes roughly between 10 and 15 seconds to download a 5MB file over the Tor network on average (which results in 333 KB/s). While such speeds are likely to be enough for general Web-surfing they might be frustrating for bulk file downloads, watching videos, or having a video conference [15]. The later types of traffic could be the reason why Tor clients may decide to get improved service from Tor relays. This might be especially true for Bittorrent

[4] The described message sequence borrows from Stratum protocol http://mining.bitcoin.cz/stratum-mining
[5] For June – September 2014.

users. Bittorrent over Tor has been problematic for both Bittorrent users and Tor relay operators: users did not get enough speed, and Tor operators are concerned that bulk file downloads consume a lot of bandwidth and thus decrease Quality of Service (QoS) for Web-surfing users.

Another reason why a Tor client would want to have higher capacity/lower delays is to improve QoS for his hidden services. The current version of Tor Hidden Services suffers from high delays and low speeds [14] which significantly reduces the number of users.

Choosing crypto-currencies There are more than 400 different crypto-currencies nowadays [8] (however only few of them achieved noticeable market capitalisation and are less susceptible to huge fluctuations in market value towards fiat currencies). According to [6] and [30] the following PoW algorithms are used in existing crypto-currencies: Blake-256, Groestl, HEFTY1, JHA, Keccak, NeoScrypt, Quark, Scrypt, Scrypt-Adaptive-Nfactor, Scrypt-Jane, SHA-256, X11, X13 (see Table 1).

Profitability of mining a digital currency obviously depends on the miner's hash-rate, price of electricity, the currency's difficulty, and its current market price. The miner's hash-rate can vary significantly depending on hardware. Table 2 shows hash-rates achievable for different algorithms on Intel Core i7-2760QM (4 cores at 2.40GHz). The table also includes maximum revenue[6] for each algorithm for the 1st of September 2014 according to [6] (averaged over multiple observations). Electricity costs are estimated to be 11 cents per day given that max power of the CPU is 45W. During the day we also observed short periods of time when the revenue jumped to 11 cents per day. Also note that hash rates achievable on GPU's can be an order of magnitude higher. We assume that an average user of our protocol does not use ASICs.

Profit estimation. In order to estimate[7] how much a Tor relay can earn using the proposed scheme we first make the following assumptions:

- Among 2,000,000 daily Tor clients (according to the Tor statistics), only 500,000 are real users and the rest belong to botnets [20]. I.e. only 500,000 users can mine.
- Moreover we assume that each user's session takes about 1 hour and every user is willing to mine with a hash-rate similar to that from Table 2. The later implies that clients will spend 100% of CPU on mining during 1 hour period. If clients decide to use less fraction of their CPU, the revenue of a Tor relay will decrease proportionally.

Income of a Tor relay obviously depends on the number of users which establish their circuits through this relay. This in turn depends on the relay's consensus bandwidth. We consider the case in which the scheme motivates running a Tor

[6] Revenue can be smaller when trying to exchange due to small market size.

[7] These are of course very rough estimates: it's not possible to learn the current hardware of Tor users, estimate the fraction of non-botnet Tor users, the number of Tor users which would be willing to mine, and the number of new (Bittorrent over Tor) users.

Blake-256	BlakeBitcoin Blakecoin Dirac Electron Photon
Groestl	Diamond Groestlcoin
HEFTY1	Heavycoin Mjollnircoin
JHA	JackpotCoin
Keccak	365coin Maxcoin Slothcoin Cryptometh
NeoScrypt	Phoenixcoin
Quark	CNotes Quark Securecoin Animecoin BitQuark Diamondcoin
Scrypt	42 Alphacoin Anoncoin Auroracoin BBQCoin Bitbar Bottlecaps Casinocoin Catcoin CHNCoin CryptogenicBullion DigiByte Digitalcoin DNotes Dogecoin Earthcoin Einsteinium Emerald Fastcoin Feathercoin Franko Globalcoin Goldcoin Grandcoin HoboNickels Infinitecoin Klondikecoin Krugercoin Litecoin Luckycoin Lycancoin Megacoin Mincoin Myriadcoin-Scrypt Nautiluscoin Netcoin Noblecoin Noirbits Novacoin Nyancoin Potcoin Quatloo Razor Reddcoin RonPaulcoin Rubycoin Sexcoin Stablecoin Starcoin Tagcoin Teslacoin USDe Viacoin Worldcoin
Scrypt-Adaptive-Nfactor	Entropycoin Execoin GPUcoin Murraycoin ParallaxCoin SiliconValleyCoin Spaincoin Spots Vertcoin VirtualMiningCoin VertCoin
Scrypt-Jane (Scrypt-Chacha)	YaCoin Ultracoin Velocitycoin
SHA-256	Battlecoin Betacoin BigBullion Bitcoin Bytecoin Curecoin Devcoin eMark Fireflycoin Freicoin Ixcoin Joulecoin Mazacoin Myriadcoin-SHA-256 Namecoin OpenSourcecoin Peercoin SaveCoin Takcoin Teacoin TEKcoin Terracoin Tigercoin Unobtanium Zetacoin
X11	ConspiracyCoin Cryptcoin Darkcoin Fractalcoin GlobalDenomination Hirocoin Karma Logicoin Smartcoin Vootcoin X11coin
X13	MaruCoin BoostCoin X13Coin

Table 1. Proof-of-work algorithms and corresponding crypto-currencies

Exit node (currently there are only about 1,000 Exits out of 6,000 Tor relays). The case in which Tor clients pay evenly to all 3 relays in the circuit is considered in Appendix A. The green line in Figure 2 shows the income of an Exit relay under the assumption that each client can mine an equivalent of 3.8 cents per 24 hours of which a fraction of 1/24 is received by the relay during a 1 hour session. For such a case top Tor relays (with consensus bandwidth 200,000 KB/s) can earn about 500 USD per month. A middle-tier relay with consensus bandwidth 10,000 KB/s can earn about 25 USD. The green line in Figure 3 shows monthly incomes assuming 11 cents per client per day (in which case a top Tor relay can earn up to 1,600 USD).

Running a high-bandwidth Tor relay obviously means high costs. In order to estimate the incurred costs we assume that the rental price is: 25 EUR per month for a relay with consensus weight less than 15,000; 40 EUR for weight between 15,000 and 50,000; 70 EUR for consensus weight larger than 50,000. In

Hashing algorithm	Rate on Intel Core i7-2760QM	Currency	Revenue per day
Blake-256	9,6 Mh/s	Blakecoin	n/a
Groestl	1 Mh/s	Diamond	2.1
HEFTY1	128 Kh/s	Heavycoin	n/a
JHA	308 Kh/s	Jackpotcoin	2.2 cents
Keccak	5.2 Mh/s	Maxcoin	0.7 cents
Quark	300 Kh/s	CNotes	3.8 cents
Scrypt	40 Kh/s	42	0.8 cents
		Litecoin	0.65 cents
		Dogecoin	0.26 cents
Scrypt-N	20 Kh/s	Vertcoin	2.3 cents
Scrypt-Jane	360 h/s	Yacoin	n/a
SHA-256d	9.6 Mh/s	Peercoin	0.01 cents
		Bitcoin	0.008 cents
X11	360 Kh/s	Smartcoin	3.8 cents
		Darkcoin	2.5 cents
X13	104 Kh/s	Marucoin	n/a

Table 2. Hash rates of the proof-of-work algorithms on Intel Core i7-2760QM

addition we assume that 10 TB of traffic is included into the server's price and one has to pay 2 EUR per additional 1 TB [13]. It is important to note that we consider costs which Tor relays already have regardless whether they use the proposed rewarding scheme or not. Note also that in order to compute traffic costs of a relay we take its consensus bandwidth (which represents the relay's speed in KB/s), and assume that the relay constantly transmits with such speed which results in upper bound of traffic costs.

Costs to run an Exit relay of specific bandwidth and corresponding profitability of running such a relay (given the income produced by mining clients) are shown in Figures 2 and 3 with blue and red lines. A Tor relay partially compensates its costs in case of 3.8 cents per day per client; when clients mine an equivalent of 11 cents per day, the relay's costs are lower than its income. Additional income can be used for the server upgrade or to provide better free services.

5.2 Anonymity

In this section we discuss anonymity of the proposed scheme. In Protocol 1, after client C mined a share he sends it to the corresponding Tor relay along with the hash of a random number (to be blindly signed). All communications are done over anonymous circuits, so that the Tor relay does not learn the originator of the messages (unless it is a Guard node). In addition blind signatures prevent the Tor relay from distinguishing client C from other clients. Finally shares generated by client C contain a Bitcoin address of either the Tor relay or a mining pool (the client is even not required to have a crypto-currency account), thus they

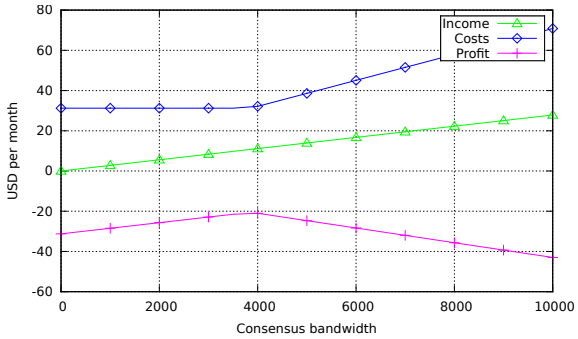

Fig. 2. Income, costs, and profit of an Exit relay in case of 3.8 cents per day per miner.

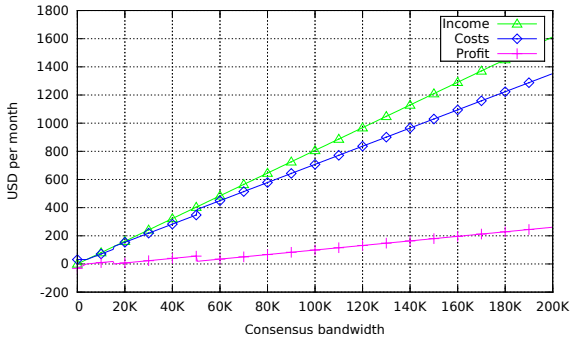

Fig. 3. Income, costs, and profit of an Exit relay in case of 11 cents per day per miner.

don't reveal the identity of the client in spite of known attacks against Bitcoin (and hence Altcoins) anonymity [25].

A curious relay can however learn the hash rate of a client, thus it may recognize repeated connections from the same client. In order to mitigate such an attack a client is advised to randomize its hash rate. The same holds if a client decides to pre-mine bandwidth tickets from a relay.

We also note that a powerful miner can try to DoS the paid traffic of a relay, by taking all the paid traffic of a relay for itself. However such behavior is not rational, since it is economically more reasonable for such miner to just earn shares in the mining pool.

6 Conclusion and discussion

Mining Bitcoins or Altcoins on consumer-grade hardware, GPUs or even first generation ASICs (for Bitcoin) is not profitable nowadays. This is due to the fact that the difference between the price of mined coins and the electricity costs is negative. Delegating mining (and thus electricity costs) to others while

keeping the earned coins obviously makes it positive[8]. In this paper we propose a scheme to reward a Tor relay in which it subscribes for mining jobs at a crypto-currency mining pool and delegates these jobs to Tor clients (thus clients indirectly pay for electricity). The Tor relay then keeps all earned coins and in turn issues priority tickets and sends them to the clients. Priority tickets can be exchanged for the improved service at the same relay. The proposed scheme has four desirable properties: (1) it does not rely on a central bank; (2) it preserves user anonymity; (3) it removes a psychological barrier since clients do not pay directly (and thus the risk of their money being stolen is removed); (4) Tor relays are rewarded with crypto-currency coins which can be exchanged for fiat currencies and partially cover their operational expenses. A relay's income can vary significantly depending on crypto-currency exchange fluctuations, number of Tor clients willing to mine, hardware, etc. In a concrete example, assuming that clients mine for Exit relays only and if each client is able to mine an equivalent of 11 cents per day and mines 1 hour per day, an Exit relay with Consensus bandwidth 100,000 KB/s can earn 800 USD per month; such revenue should completely cover the relay's traffic costs and may allow the operator to upgrade to a more powerful server.

The proposed scheme does not decrease anonymity provided by the Tor network. All shares submitted by clients are anonymous and contain a Bitcoin address of either a Tor relay or a mining pool, thus attacks against Bitcoin anonymity become inapplicable. A curious relay can however learn a client's hash rate. Also in the case of pre-mining for later usage the relay will learn that the same user tries to go through it later on the same day.

Finally we would like to mention that if altcoins with strong anonymity (ex. Zerocoin [21]) become widely adopted it would be easy to integrate such payments into our scheme. A client will need to send together with the payment the blinded value for signing. The relay will need to broadcast a transaction with this value signed, from which the client will be able to derive the signature and thus the priority ticket.

Usages other than Tor The proposed scheme can be used not only to reward Tor relays. The same approach can be adopted by entities which accept payments. We note, that for this scheme to be successful it may be useful to go for memory-hard proofs of work, which would have no advantage in GPU or ASICs. Scrypt function used in some alt-coins (ex. Litecoin) comes close to be adequate for this purpose, though more energy-optimal tradeoff-resistant proof-of-work functions can be designed for this task. For example we refer the interested reader to the recent PHC competition [24], where several candidates like Argon, Lyra2, yescrypt provide strong ASIC resistance. We envisage that widespread use of such CPU mining in exchange for services may become a basis for a widely used micropayment system, which in turn becomes a strong alt-currency used

[8] Our scheme thus also gives an interesting use case for the old mining gear which is otherwise obsolete. This might be the only way to buy lots of priority traffic on Tor relays.

by consumers (what is currently lacking in the Bitcoin universe, where the main activities are mining and hoarding of coins).

References

1. Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) Advances in Cryptology CRYPTO 2000, Lecture Notes in Computer Science, vol. 1880, pp. 271–286. Springer Berlin Heidelberg (2000)
2. Alsabah, M., Bauer, K., Elahi, T., Goldberg, I.: The path less travelled: Overcoming tor's bottlenecks with traffic splitting. In: Proceedings of the 13th Privacy Enhancing Technologies Symposium (PETS 2013) (July 2013)
3. Androulaki, E., Raykova, M., Srivatsan, S., Stavrou, A., Bellovin, S.M.: Par: Payment for anonymous routing. In: Borisov, N., Goldberg, I. (eds.) Privacy Enhancing Technologies, Lecture Notes in Computer Science, vol. 5134, pp. 219–236. Springer Berlin Heidelberg (2008)
4. ASIC and FPGA miner in c for bitcoin: https://github.com/ckolivas/cgminer (2014)
5. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R., Sherman, A. (eds.) Advances in Cryptology, pp. 199–203. Springer US (1983)
6. CoinWars: Crypto Currencies: http://www.coinwarz.com (2014)
7. CPU miner for bitcoin: https://github.com/jgarzik/cpuminer (2014)
8. Crypto-Currency Market Capitalizations: http://coinmarketcap.com (2014)
9. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (August 2004)
10. Eloipool - FAST Python3 pool server: https://bitcointalk.org/index.php?topic=61731.0 (2014)
11. Evans, J.W., Filsfils, C.: Deploying IP and MPLS QoS for Multiservice Networks: Theory & Practice. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2007)
12. From Onions to Shallots: Rewarding Tor Relays with TEARS: http://dedis.cs.yale.edu/dissent/papers/hotpets14-tears.pdf (2014)
13. Hetzner Online Server Auction: https://robot.your-server.de/order/market (2014)
14. Hidden Services need some love: https://blog.torproject.org/blog/hidden-services-need-some-love (2014)
15. How much bandwidth does Skype need?: https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need (2014)
16. Jansen, R., Hopper, N., Kim, Y.: Recruiting new Tor relays with BRAIDS. In: Keromytis, A.D., Shmatikov, V. (eds.) Proceedings of the 2010 ACM Conference on Computer and Communications Security (CCS 2010). ACM (October 2010)
17. Jansen, R., Johnson, A., Syverson, P.: LIRA: Lightweight Incentivized Routing for Anonymity. In: Proceedings of the Network and Distributed System Security Symposium - NDSS'13. Internet Society (February 2013)
18. Linux HTB Home Page.: http://luxik.cdi.cz/ devik/qos/htb/ (2014)
19. M., G., Richardson, M., Ford, B., Jansen, R.: A TorPath to TorCoin: Proof-of-Bandwidth Altcoins for Compensating Relays. In: 7th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs) (July 2014)
20. Massive spike of Tor users caused by Mevade botnet: http://www.net-security.org/secworld.php?id=15530 (2014)

21. Miers, I., Garman, C., Green, M., Rubin, A.D.: Zerocoin: Anonymous distributed e-cash from bitcoin. In: IEEE Symposium on Security and Privacy (2013)
22. Ngan, T.W.J., Dingledine, R., Wallach, D.S.: Building Incentives into Tor. In: Sion, R. (ed.) Proceedings of Financial Cryptography (FC '10) (January 2010)
23. Ostrovsky, R.: A proposal for internet computation commerce: How to tap the power of the web. In: Presentation at CRYPTO '98 rump session (1998)
24. Password Hashing Competition: https://password-hashing.net (2014)
25. Ron, D., Shamir, A.: Quantitative analysis of the full bitcoin transaction graph. In: Financial Cryptography and Data Security (FC'13). Springer (2013)
26. Specification of Guaranteed Quality of Service: http://www.rfc-editor.org/rfc/rfc2212.txt (2014)
27. Tor incentives research roundup: GoldStar, PAR, BRAIDS, LIRA, TEARS, and TorCoin: https://blog.torproject.org/category/tags/incentives (2014)
28. Tor Metrics: https://metrics.torproject.org/ (2014)
29. Tor Metrics: Performance: https://metrics.torproject.org/performance.html (2014)
30. Windows GPU Miners for the More Commonly Used Crypto Algorithms: http://cryptomining-blog.com/2595-windows-gpu-miners-for-the-more-commonly-used-crypto-algorithms/ (2014)

APPENDIX

A Profit estimation

In this section we consider the case when a Tor client pays evenly to all 3 relays in the circuit. Profit estimations for this case are shown in Figures 4 and 5 (red lines). In case of 3.8 cents per client per day the top Tor relays (with consensus weight 200,000 KB/s) can earn up to 160 USD. A middle-tier relay with consensus bandwidth 10,000 KB/s can earn about 8 USD.

For the case when a client mines for all 3 relays in the circuit, a relay starts to be profitable if each client mines an equivalent of 36 cents per day as shown in Figure 5. We note that in all cases the proposed scheme covers a part of relay renting costs.

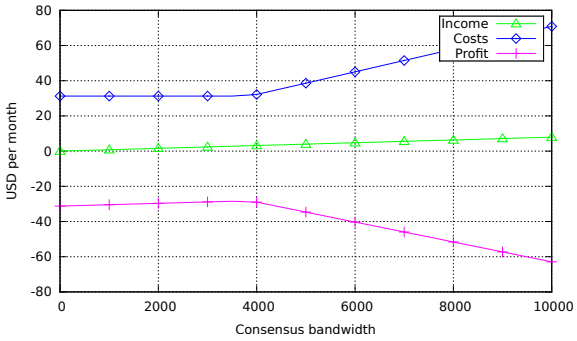

Fig. 4. Income, costs, and profit of a relay in a 3-hop circuit in case of 3.8 cents per day per miner.

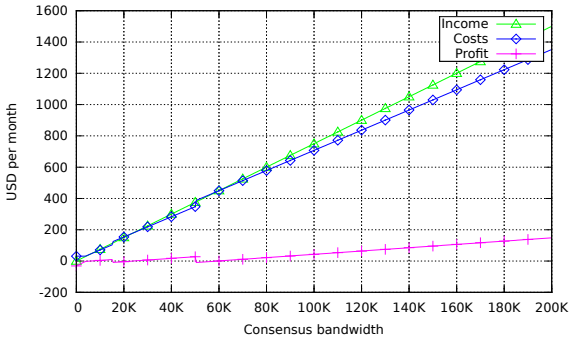

Fig. 5. Income, costs, and profit of a relay in a 3-hop circuit in case of 36 cents per day per miner.

Thses estimates assume 1 hour of Tor usage per client per day and assume 100% CPU mining during this 1 hour. In practice 25% CPU would be more realistic.

B Partially blind WI-Schnorr signature scheme

In this section we describe a partially blind signature scheme from [1]. User U wants to get a partially blind signature over message `msg` attributed by common information `info`. For the case of the micro-payment protocol described in this paper, the common information `info` is a timestamp chosen by signer S. User U and signer S execute signature issuing protocol described in Protocol B.1.

Protocol B.1. Partially blind WI-Schnorr signature issuing protocol
User U obtains a partially blind signature over `msg` with added factor `info` from signer S

Signer S : $(p, q, g, x, \texttt{info})$
User U : $(y = g^x, \texttt{info}, \texttt{msg})$

1: S : $u, s, d \in_R \mathbb{Z}_q$
$z = \mathcal{F}(\texttt{info})$
$a = g^u,\ b = g^s z^d$
$S \to U$: a, b
2: U : $t_1, t_2, t_3, t_4 \in_R \mathbb{Z}_q$
$z = \mathcal{F}(\texttt{info})$
$\alpha = a g^{t_1} y^{t_2}$
$\beta = b g^{t_3} y^{t_4}$
$\varepsilon = \mathcal{H}(\alpha \| \beta \| z \| \texttt{msg})$
$e = \varepsilon - t_2 - t_4 \bmod q$
$U \to S$: e
3: S : $c = e - d \bmod q$
$r = u - cx \bmod q$
$S \to U$: (r, c, s, d)
4: U : $\rho = r + t_1 \bmod q$
$\omega = c + t_2 \bmod q$
$\sigma = s + t_3 \bmod q$
$\delta = d + t_4 \bmod q$
$\omega + \delta \stackrel{?}{=} \mathcal{H}(g^\rho y^\omega \| g^\sigma z^\delta \| z \| \texttt{msg})$
signature$=(\rho, \omega, \sigma, \delta)$

In Protocol B.1, p and q are large prime numbers such that $q|p-1$, and g is an element in $\mathbb{Z}_p^*$ whose order is q. The protocol uses two public hash functions $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_q$ and $\mathcal{F} : \{0,1\}^* \to \langle g \rangle$ ($\langle g \rangle$ denotes a subgroup in $\mathbb{Z}_p^*$ generated by g). In addition $x \in \mathbb{Z}_q$ is a secret key and $y = g^x$ is the corresponding public key.