Bitcoin Micropayment Channels

Brian Bloomer and Toshiro Nishimura

Table of Contents

Acknowledgments

We would like to sincerely thank our advisor, Doug Tygar, for his insight and guidance during this project.

We would also like to express our gratitude toward the I School faculty for all that they taught us and the I School staff for their unfailing support.

Lastly, we would like to thank our fellow students for the good times and camaraderie these past two years.

Introduction

Micropayment systems have been proposed for years but none have yet to take hold. The definition of a micropayment has changed from the 1990's to 2015; older academic papers defined a micropayment as five, six, or seven dollars or less. Today a micropayment is typically defined as an amount less than one dollar. As the world continues to get online, become increasingly networked, and more and more objects are given computing abilities, such as "smart" lightbulbs, the need for an efficient payment system that can handle micropayments grows. Digital currencies have been proposed as a mechanism for enabling efficient resource exchange between "internet of things" objects by IBM researchers.[1] Currently implemented payment systems are effective for most transactions, but their costly frictions caused by infrastructure and protocol level constructs make micropayments difficult.

Many valuable and intriguing systems have been proposed by scholars and technologists that require an efficient micropayment system. Media content could be sold via a pay-per-article system rather than the currently existing subscription services. Adding very small fees to sending email wouldn't have any impact on regular people's lives but could potentially be a very effective way to combat spam. Networked, P2P systems that rely on altruistic behavior could instead incentivize nodes through micropayment integration.

With the advent of Bitcoin and the blockchain, it is possible to create new micropayment schemes. The unique nature of the blockchain and the Bitcoin transaction system allows for entities to engage in microtransactions without relying on the pre-existing payment networks that are unable to accommodate microtransactions and require third parties such as credit card companies or banks. The value of not relying on a third party is of particular value, for it means the

[1] Higginbotham, Stacey. "Check out IBM's Proposal for an Internet of Things Architecture Using Bitcoin's Block Chain Tech." Gigaom, 09 Sept. 2014. Web. 06 May 2015.

scheme could be implemented in regions of the world where there is no reliable payment infrastructure yet where people are gaining internet access. Furthermore, it allows any two entities to engage in a transaction without requiring the approval of a third party. The payment cannot be stopped in the way that credit card networks prevent people from donating money to Wikileaks, for example. [2]

For our final project we decided to focus on Bitcoin-based micropayments. We researched the history of micropayment schemes to provide a greater historical context for the new schemes being proposed in the present. We then explored two micropayment schemes that have been proposed using Bitcoin: direct micropayments and micropayment channels. Bitcoin micropayment channels are more efficient than direct Bitcoin network micropayments, so we evaluated a number of possible use cases for Bitcoin micropayment channels and opted to implement one of those use cases: a router selling metered wireless network access. We designed a system for our use case and created a software library that allowed us to implement Bitcoin micropayment channels.

Brief History of Micropayment Schemes

Micropayment schemes have been proposed since the 1990's. When no such scheme was widely adopted interest waned in the 2000's, but recently there has been a resurgence of interest as computing has advanced and digital currencies have been invented. All micropayment schemes are concerned with efficiency, keeping the processing costs of transactions as low as possible, and correctness. Other properties researchers have prioritized for micropayment schemes include anonymity, transferability, culpability, exculpability, scalability, and fairness. Most micropayment schemes represent "coins" as hash chains or messages signed with digital signatures. More recently proposed schemes have more innovative forms of micropayments.

[2] Greenberg, Andy. "Visa, MasterCard Move To Choke WikiLeaks." Forbes. Forbes Magazine, 07 Dec. 2010. Web. 06 May 2015.

In 1995 Lei Tang published "A Set of Protocols for Micropayments in Distributed Systems." The protocols rely heavily on public key cryptography, and like many proposed micropayment schemes, it involves at least three parties: the customer purchasing a good or service (using micropayments), a merchant selling the good or service, and a "billing service center" that debits the purchaser's bank account and credits the merchant's bank account.[3] Three notable issues with this type of model, the customer/vendor/bank model, are that the intermediation of the bank inevitably adds transaction costs to the act of purchasing and therefore is less efficient than a model that doesn't require a third party, introducing a third party such as a "billing service center" creates additional security risks and a single point of failure for the model, and the political costs of having the customer and vendor rely on a third party to enable their transactions: this kind of political cost was highlighted when various payment processors refused to process donations from individuals to Wikileaks.

One of the earliest micropayment schemes is PayWord[4], proposed by Rivest and Shamir. PayWord relies on using hashing functions to chain together payments from an individual buyer, thereby aggregating numerous micropayments into a single transaction that can be processed by the bank. The main problem suffered by PayWord is that the merchant cannot chain together payments from multiple buyers. So if an individual only makes one very small transaction, say a fraction of a cent, that payment would cost more to process than it is worth. A few years later Jutla and Yung proposed PayTree[5], a scheme designed to solve PayWord's problem of not allowing merchants to aggregate payments from multiple buyers. PayTree

---

[3] Tang, Lei. "A set of protocols for micropayments in distributed systems."*Proceedings of the First USENIX Workshop on Electronic Commerce, USENIX*. 1995.

[4] Rivest, Ronald L., and Adi Shamir. "PayWord and MicroMint: Two simple micropayment schemes." *Security Protocols*. Springer Berlin Heidelberg, 1997.

[5] Jutla, C., and M. Yung. "PayTree." *Amortized-signature" for flexible micropayments", in Second USENIX Workshop on Electronic Commerce, Oakland CA*. 1996.

utilized merkle trees to allow merchants to aggregate payments from more than one buyer. Both schemes, however, still suffer from the problem of relying on a bank to process all payments.

Other early micropayment schemes include Millicent, Netcard, MPTP (micropayment transfer protocol), and Micromint. They tended to employ various cryptographic schemes and signatures for greater efficiency. None became widely adopted.

Ronald Rivest proposed a rather creative micropayment scheme a few years later that relied upon "probabilistic payments with 'electronic lottery tickets.'"[6] In this case, a "ticket" that has a one in a hundred chance of winning one dollar has the expected value of one cent. Theoretically, on average all of the users of the micropayment scheme would receive the correct amount of intended dollars spent/received over time even though some of the tickets they spend or receive would provide them with no value at all. This scheme also relies on a customer-vendor-bank model, and is intended to be far more efficient than other micropayment schemes because the bank would only need to process the lottery tickets that "won" and actually transferred funds from a customer to a merchant. While exceptionally clever, many businesses cannot afford to gamble on whether or not a customer's payment is going to go through each time the business provides a good or service. Many businesses do not possess large reserves of cash, and need guaranteed cash coming into their account with every transaction so they can pay their bills at the end of the month. There is also a psychological hurdle introduced by this scheme, because people are accustomed to a cash model where one dollar is always worth one dollar. Reluctance to adopt a technology that changes the model of cash would likely meet resistance. This scheme also possesses the issues previously noted associated with client/vendor/bank models.

---

[6] Rivest, Ronald L. "Electronic lottery tickets as micropayments." *Financial Cryptography*. Springer Berlin Heidelberg, 1997.

In 2003, Yang and Garcia-Molina proposed PPay, a micropayment scheme specifically designed for peer-to-peer (P2P) systems.[7] Part of the motivation for the design of this scheme was the online content piracy occurring at a large scale at this point in time, the fall of these pirating systems, and new P2P entrants to the marketplace that hoped to enable users to legally purchase the content being distributed on these P2P systems. PPay's designers recognized that customer/vendor/bank models possess scalability and performance problems due to the bank element of the model. PPay did not entirely eliminate the bank from the payment model, but drastically reduced its involvement for "when peers open or close accounts, for arbitration, and in limited cases, to perform services on behalf of offline peers." One limiting issue for this type of scheme is that it is only applicable to P2P networks. PPay does not provide anonymity. An extension of PPay called WhoPay was designed to enable anonymity with PPay.[8]

As proposed micropayment schemes failed to gain widespread adoption, researchers began presenting more novel schemes, such as micropayments for Tor, network resource distribution, incentivizing new types of behavior such as motivating individuals to capture and share data from their mobile devices.[9] Micropayment schemes have been expanded to address the needs of "cloud resource economies."[10] Researchers have also investigated using micropayments as a business model for online content consumption (such as news articles), a widely-discussed potential business model that has not yet been adopted by the market.

---

[7] Yang, Beverly, and Hector Garcia-Molina. "PPay: micropayments for peer-to-peer systems." *Proceedings of the 10th ACM conference on Computer and communications security*. ACM, 2003.
[8] Jain, Mohit, Siddhartha Lal, and Anish Mathuria. "A SURVEY OF PEER-TO-PEER MICROPAYMENT SCHEMES."
[9] Reddy, Sasank, et al. "Examining micro-payments for participatory sensing data collections." *Proceedings of the 12th ACM international conference on Ubiquitous computing*. ACM, 2010.
[10] Zhanikeev, Marat. "Coins in cloud drives can use OAuth for micropayments and resource metering alike." *Proceedings of The Ninth International Conference on Future Internet Technologies*. ACM, 2014.

The possibility of using micropayments to incentivize "selfish nodes" in a multihop wireless network is another area of exploration.[11]

Some new schemes stretch the boundaries of the definitions of micropayments, for example, by defining a micropayment as a user choosing to run computations on behalf of the merchant, thereby expending valuable computing power in exchange for a good or service. [12] Microcomputations as micropayments could also be put in the service of projects aimed at advancing certain scientific or socially-minded projects.[13]

Models have been designed to take advantage of mobile networks like 4G, such as PRIPAY proposed by Ntantogian, Gkikakis, and Xenakis.[14] These models are designed with an eye on the future and the rapid proliferation of internet enabled mobile devices throughout the developing world. PRIPAY does not attempt to make each micropayment economically efficient. Instead, in the PRIPAY model the mobile operator aggregates all of a user's payments into a single payment and charges them for a single transaction on their monthly bill. While PRIPAY claims to preserve anonymity, the mobile operators who are charging their customers for purchases will have insight into the customer's activities. Anonymity is maintained between customer and merchant, but not between the customer and their mobile operator.

Many businesses that allow for micropayments do not actually rely on micropayment schemes at all, but simply aggregate multiple transactions that a user

---

[11] Kasmir, Presty, and Sakhi S. Anand. "Survey on Routing in Multihop Wireless Networks Using Micropayment Schemes." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 2.2 (2013): pp-362.
[12] Karame, Ghassan O., Aurélien Francillon, and Srdjan Čapkun. "Pay as you browse: microcomputations as micropayments in web-based services."*Proceedings of the 20th international conference on World wide web*. ACM, 2011.
[13] Karame, Ghassan O., et al. "Microcomputations as Micropayments in Web-based Services." *ACM Transactions on Internet Technology (TOIT)* 13.3 (2014): 8.
[14] Ntantogian, Christoforos, Dimitris Gkikakis, and Christos Xenakis. "PRIPAY: A Privacy Preserving Architecture for Secure Micropayments." *ICSNC 2012, The Seventh International Conference on Systems and Networks Communications*. 2012.

makes on their system into a single transaction that they then charge to the user's credit card, bank account, or other type of billable account. Flattr, a startup launched in 2010, enables "micropayments" to occur between users, but only process an actual transaction when a user wishes to withdraw funds from their Flattr account to their bank account. M-Coin, another "micropayment" system launched in 2010, places micropayment transactions on the user's phone bill, which are then paid for in aggregate. Perhaps the most well known example of this kind of micropayment system is Apple's iTunes store.

Bitcoin

Bitcoin[15] is a protocol invented in 2008 that enables a P2P online payment network that allows agents to transfer bitcoins, a digital currency. Unlike the vast majority of payment systems, including many of the micropayment schemes discussed above, the Bitcoin network allows users to transact with one another without requiring any intermediaries. Transactions are broadcast to the entire network, validated by network nodes, and then recorded in a distributed public ledger called the blockchain. The blockchain allows users to transact with one another without requiring a specific intermediary because anybody can check the ledger to verify that a user actually possesses the bitcoins they are trying to spend, hence preventing double-spending of coins from occurring.

Bitcoins are held at specific addresses, and transactions move bitcoins from one set of addresses to another set. Addresses correspond to unique public/private key pairs, and it is by using a private key to sign a transaction that a user is able to spend their bitcoins. Once a transaction has been validated by a "miner" node in the network, it is added to a "block" – a set of transactions. Once a miner node solves a proof-of-work computation – a mechanism employed by the Bitcoin protocol to ensure the security of the network – the block is broadcast to the rest of the

[15] Nakamoto, Satoshi. "Bitcoin White Paper."

network and added to the blockchain, thus becoming part of the official distributed public ledger of all bitcoin transactions. Operators of mining nodes are rewarded for contributing computational power to solving proof-of-work computations by receiving bitcoins in exchange for their efforts. Miners also receive (optional) transaction fees for including transactions in their blocks. These fees are currently much lower than those imposed by other payment networks, and many miners do not require any transaction fee at all to include transactions in their blocks.

Bitcoin has a built-in scripting language. When generating a transaction, users utilize the scripting language to dictate how the coins that are being spent can be accessed by the next transaction that wants to spend them. The majority of bitcoin transactions currently broadcast to the network simply dictate that the coins are being sent to address *A*, and that the corresponding private key that maps to address *A* must be used to sign the next transaction attempting to spend the bitcoins held at address *A*. More complex transactions are also possible. There is currently a movement for people to use "multisig" transactions. Multisig transactions dictate that for bitcoins to be spent, the spender must provide signatures from *n*-of-*m* private keys (associated with *m* addresses). For example, a 2-of-3 multisig transaction sends bitcoins to three different addresses, and two of the three private keys associated with those addresses must be used to sign a transaction to spend the bitcoins. The scripting language is also what allows us to create Bitcoin micropayment channels, which we implemented for our final project.

Bitcoin Micropayments

Excited by the new possibilities created by Bitcoin, we decided to explore the intersection of Bitcoin and micropayments. First, we researched two alternative ways to perform micropayments on the Bitcoin network: directly broadcasting micropayments to the Bitcoin network, and Bitcoin micropayment channels. After concluding that Bitcoin micropayment channels were more efficient than direct

broadcast, as well as more technically interesting to us, we set out to implement Bitcoin micropayment channels in a prototype.

Although it is possible to directly broadcast micropayments as individual transactions in the Bitcoin network, there are a number of issues that arise from this model of Bitcoin micropayments. As described by the bitcoinj wiki:

1. *If you send too many transactions too fast, they will get down-prioritized or not relayed by various anti flooding algorithms built into the Bitcoin network.*
2. *There is a minimum amount of value a single transaction can send, determined by the number of bytes required to send and claim it along with the fees charged.*
3. *The recipient of the micropayments ends up with a wallet full of "dust" which can be expensive to send, fee-wise.*[16]

Micropayment channels do not solve these problems in general, but do offer a solution when the micropayments are to be made between the same service consumer and producer for a given "micropayment channel session". The key to micropayment channels is that the customer generates rapid micropayment transactions and sends them directly to the vendor, thus enabling metered payments, but ultimately only two transactions need to be broadcast to the Bitcoin network to prevent fraud and for the vendor to redeem all of the micropayments made by the customer. This scheme prevents the issues of down-prioritization of rapid transactions, a build-up of rapid fees to miners, and wallet dust. Furthermore, this scheme allows for the establishment of such channels with zero-trust and negligible risk. Neither party can steal more than a negligible, incremental payment.

---

[16] "Working with Micropayment Channels." Working with Micropayment Channels. N.p., n.d. Web. 06 May 2015. <https://bitcoinj.github.io/working-with-micropayments>.

We opted to implement Bitcoin micropayment channels in the context of a router selling metered access to a wireless network.

Advantages of Bitcoin micropayment channels include:

- Rapid generation of secure payments between customer and vendor, yet does not require the Bitcoin network to process every micropayment transaction, thus reducing friction, fees, and processing costs.
- Not requiring a third party to intermediate the transactions between customers and vendors. The lack of a third party broker or bank in the model reduces complexity. Furthermore, the lack of an intermediary means that the transacting parties cannot be prevented from transacting by the intermediary. This type of interference by an intermediary has occurred, such as when various payment processors blocked their users from donating money to Wikileaks.[17]

Micropayment Channel Protocol

Micropayment Channel Protocol consists of 3 stages: Channel Start, Channel Connected, and Channel End.

The following flowchart (figure 1) describes how a client would establish a micropayment channel with a server, send micropayments, and then close the channel. We then describe each step of the flow chart below:

[17] Greenberg, Andy. "Visa, MasterCard Move To Choke WikiLeaks." Forbes. Forbes Magazine, 07 Dec. 2010. Web. 06 May 2015.

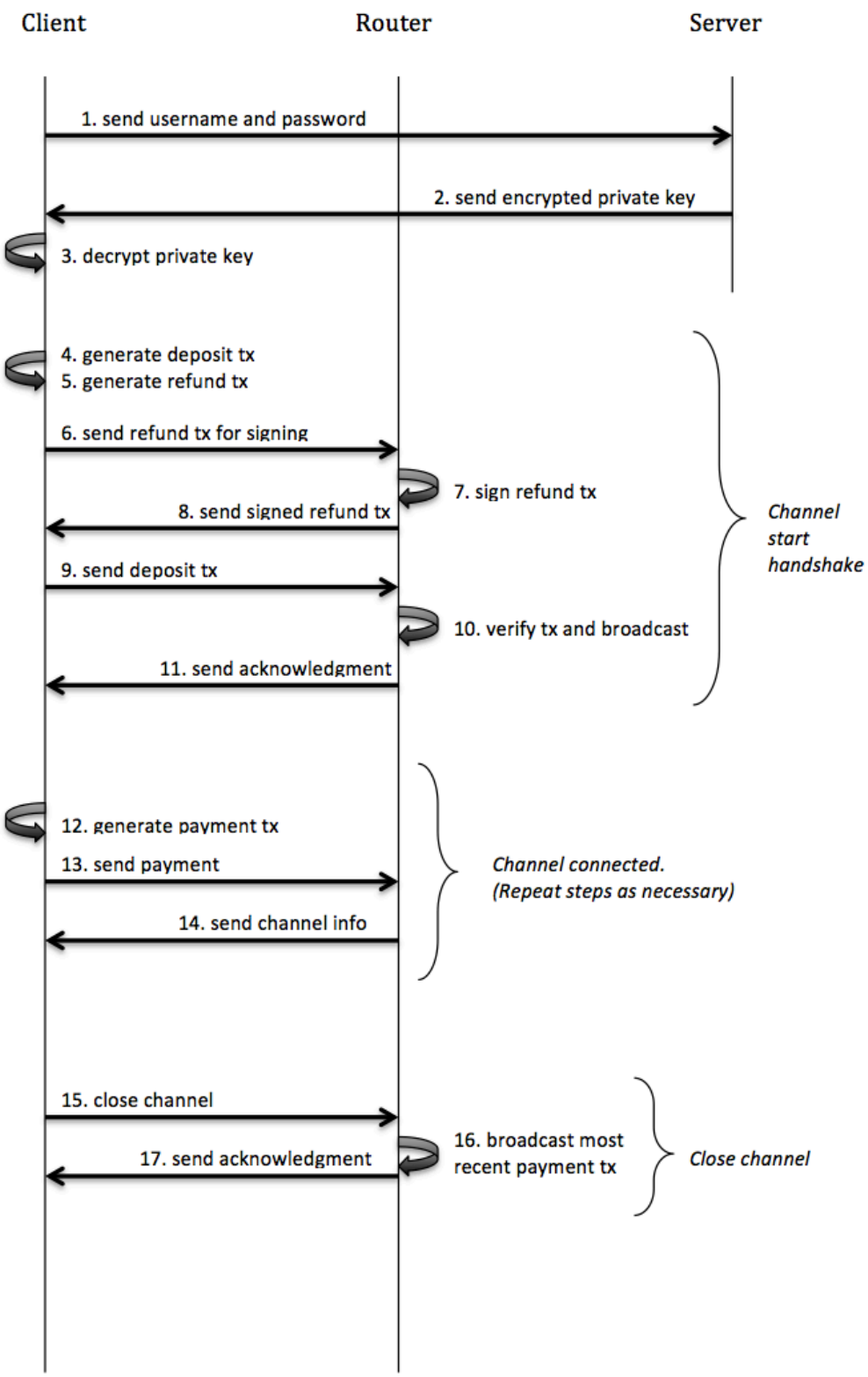

Figure 1: Bitcoin micropayment channel flowchart

Private key retrieval

In our implementation, an additional step that occurs in the system before micropayment channels are established is the client retrieval of a private key that gives them access to their bitcoins. This phase could be accomplished in any number of ways, but for illustration, we are assuming that the client's private key is encrypted and stored on a third-party server. It would not be difficult to adapt this protocol for hardware signers.

1. **Client sends username and password to key-server.**
2. **Key-server replies with encrypted private key.**
3. **Client decrypts private key.**

Channel Start Handshake

The **channel start** phase is the most complicated.

4. **Client generates deposit transaction.** The *deposit transaction* is created by the client to declare to the router that it has a certain amount of Bitcoin, and that it is willing to deposit it in a sort of cryptographic-escrow. In order to spend the funds locked in the deposit transaction, both the client and the router's signatures are needed.
5. **Client generates refund transaction.** The *refund transaction* sends all funds from the deposit transaction back to the client. It is created by the client and sent to the router for its signature *before* the router has seen the deposit transaction. As it spends funds from the deposit transaction, it requires signatures from both the client and the router. It cannot be redeemed for a prespecified time (several hours, say). The client signs its half here. In a successful micropayment channel, the refund transaction is never used.
6. **Client sends refund transaction to router for signing.**
7. **Router signs refund transaction.** Router uses its key to sign the other half of the refund transaction.

8. **Router replies with fully signed refund transaction.**
9. **Client sends deposit transaction.** The client does not send the deposit transaction until it possesses a signed refund transaction, for if the deposit transaction were to be broadcast and the router became unresponsive and the client did not possess the refund transaction, the client's deposit funds would be locked in escrow in perpetuity.
10. **Router checks and broadcasts deposit transaction.** Broadcasting the transaction onto the Bitcoin network locks it into the channel.
11. **Router send acknowledgement.** The channel is now officially open and the router is ready to accept payments.

Channel Connected Phase

During the **channel connected** phase, the client uses the services provided by the router, monitors its own usage and sends payment when needed.

12. **Client generates a payment transaction.** The *payment transaction* sends part of the deposit transaction to the router and the rest back to the client. The portion for the router is a payment. To "pay more" to the router, the client creates another transaction which sends more of the deposit to the router.
13. **Client sends payment transaction to router.** The payment transaction is half-signed at this point and it's in the router's power to sign and broadcast the payment whenever it wants to.
14. **Router replies with channel information.** This reply contains information such the amount of service (megabytes of bandwidth, in our example) consumed, funds remaining, etc.

Channel End Handshake

The **channel end** phase can either be initiated by the client manually, or automatically by the router when the client runs out of funds.

15. **Client sends close end request router.**
16. **Router broadcasts the most recent and highest valued payment transaction.**
17. **Router sends acknowledgement.**

Protocol Peculiarities

Peculiarities of the this entire protocol are that:

- The router, if it wants to get paid at all, *must* sign and broadcast its payment transaction before the lock time of the refund transaction. Otherwise, the client will broadcast its refund and get all of its funds back.
- The client cannot use its deposit for any other purpose during the session. The protocol itself provides no guidance on what the deposit or session time should be.
- The most that could be lost by the client is one incremental payment, assuming that it is paying attention to the service being delivered and promptly halts sending payment transactions in case the service stops.
- Because only the final payment transaction and the deposit transaction needs to be broadcast, transaction fees are minimized.
- The client cannot really end the session itself, since it has no way of forcing the router to spend the final payment transaction, freeing the rest of the funds held in the deposit transaction. However, as stated above, the longest the router can hold the remaining amount "hostage" is the predefined lock time.

Implementation

After deciding we wanted to implement Bitcoin micropayment channels, we needed to decide how we wanted to implement them. Bitcoin micropayment channels are particularly intriguing for metered payments. We researched goods and services

that could be distributed using metered payments, including online content, utilities, street tolls and parking (in a world of smartcars), and bandwidth. Based on the resources available to us and our skillsets, we decided to create a prototype router that sold bandwidth access to clients in metered payments.

Wifi access is typically provided to clients in two different ways: either for free, like at many coffee shops, or for a set amount of money for a set amount of time, as in airports and hotels. In some instances the wifi access is sold in various packages, providing the user a choice between purchasing an hour of wifi access or 24 hours worth. These models poorly accommodate the elasticity of demand for wifi. Cafes and similar establishments that operate in a highly competitive, low-margin market and could use additional revenue streams are unable to monetize their wifi because there is no easy way to charge for small amounts of wifi. Meanwhile, airports and hotels are missing out on potential revenue from people who want a low amount of bandwidth and are unwilling to pay for more than they need.

Protocol Design Challenge – Storing Private Keys

One of the biggest questions we faced as we designed our prototype was how to store the bitcoins that our client needed to spend to open the micropayment channel. Designing a secure storage system for private keys is a final project's worth of work in its own right. We explored existing wallet software, including Coinbase, blockchain.info, and Armory. However, we found that none of these wallets were designed to interface with software designed to sell goods or services utilizing Bitcoin micropayment channels. For this reason we decided to build our own server that would function as a "web wallet" for our demo. This server stores private keys associated with testnet bitcoin addresses holding testnet bitcoins (testnet is a parallel Bitcoin network that is explicitly designed for testing. Testnet bitcoins can be acquired via "faucets," which dispense testnet bitcoins for free). Usernames and passwords we created grant our prototype users access to these private keys so that our implementation can be demonstrated.

Designing a robust, secure web wallet did not fall within the scope of our project. The username/password system we have set up to retrieve these private keys is sufficient for our demonstration purposes but is not intended to be production ready or to possess market-grade security.

We hope that one day there will be wallet software on the market designed to accommodate micropayment channels.

We are aware that by implementing a username/password system that retrieves private keys from a centralized database, our system does not take full advantage of the decentralized potential promised by Bitcoin. Ideally, a variety of secure wallets available on the market would have micropayment channel capabilities. If that were the case we would have designed our system to allow users to open micropayment channels with the router without being required to store a private key with our server. However, working within the limitations of wallet software as it exists today, and with the primary focus of our project being the research of micropayments and implementation of Bitcoin micropayment channels, we opted to implement a username/password system.

One advantage of our implementation is that it allows users of the metercoin system to use a single username and password to access the system from any client, providing users with a high level of convenience. Furthermore, it allows consumers to use the metercoin system from devices that do not have access to other sources of bitcoins beyond their metercoin wallet.

Interface Design and Development

While the primary focus of this project was the research of micropayment schemes and the implementation of Bitcoin micropayment channels, we determined that an effective way to demonstrate Bitcoin micropayment channels would be to create a web-based user interface rather than rely on the command line alone.

We used Balsamiq to create lo-fi prototypes of our interface to come to consensus on the design of our interface.

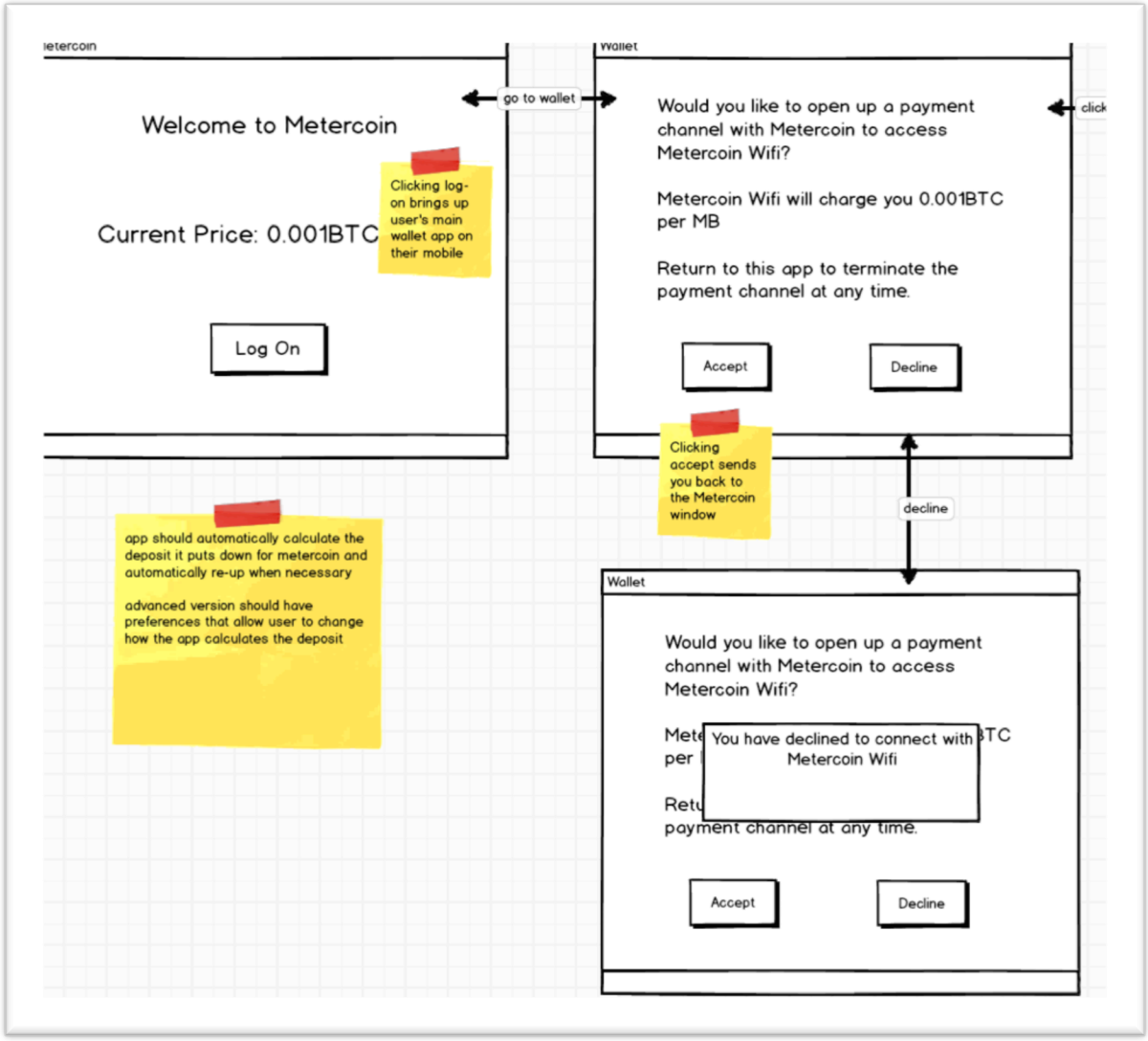

Figure 2: Early Balsamiq mockups

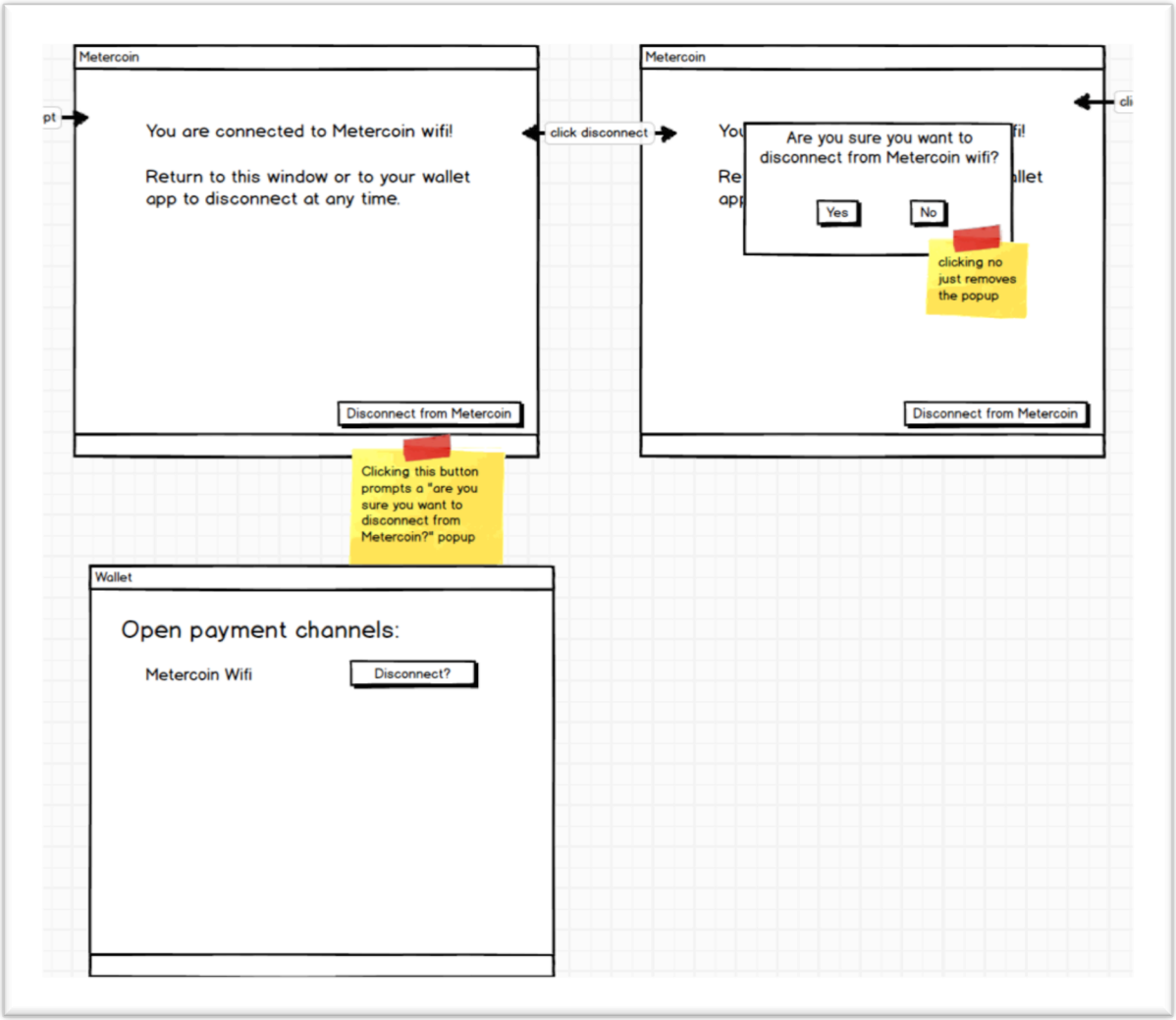

Figure 3: Early Balsamiq mockups

As we iterated on our designs and came upon technical constraints, namely the handling of private keys discussed above, we opted to reduce the complexity and number of views in our final prototype.

The front-end of our implementation is written with HTML, CSS, and javascript. We use normalize.css to standardize the look of the front-end across browsers. We explored to of the most popular front-end frameworks, Bootstrap and Foundation, but found that they were very feature-heavy and not particularly lightweight. Instead we opted for TOAST, a lightweight responsive CSS grid system, to make our

front-end responsive. We leveraged jQuery for user interactions for it simplifies the amount of code that needs to be written compared to pure javascript in many instances. Please refer to the appendix for selected images of the front-end of our prototype.

The first screen the user lands on provides the user with information about the metercoin service (selling bandwidth for bitcoin). The server sends the client information about the current price of bandwidth, which is then rendered in the interface. A username and password form is presented to the user.

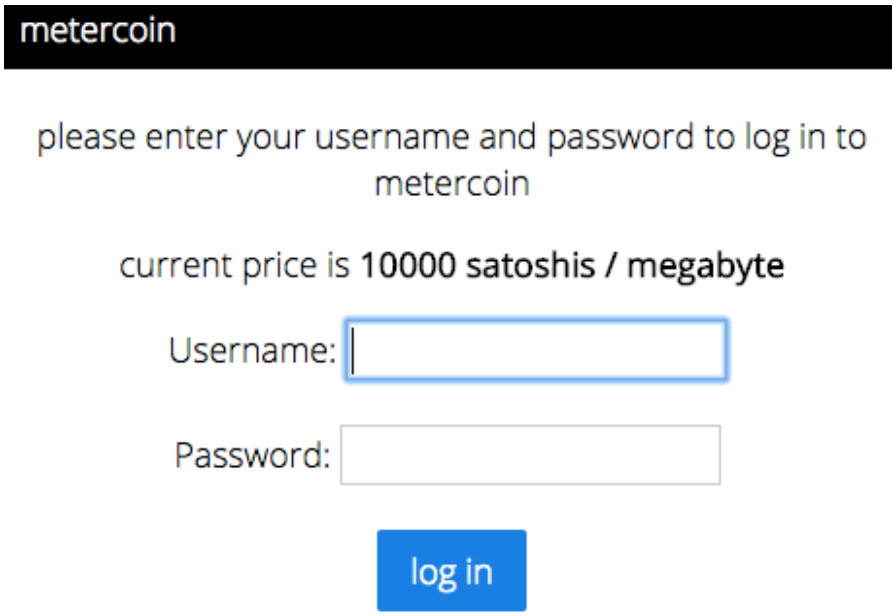

Figure 4: Log-in screen

Once the user logs in, with their associated wallet pre-loaded with testnet bitcoins, the process to open a micropayment channel with the router immediately begins. Once a channel is established the client is able to utilize bandwidth provided by the router. The interface presents usage statistics to the user that are updated very frequently. The client continually requests information from the router in order to update usage statistics. The usage statistics also allow the client to know when it has almost used up the allotted amount of bandwidth it paid for and to send another micropayment to the service. When the user logs off or closes the metercoin page, the micropayment channel is closed and the router broadcasts the final payment transaction sent to it by the client.

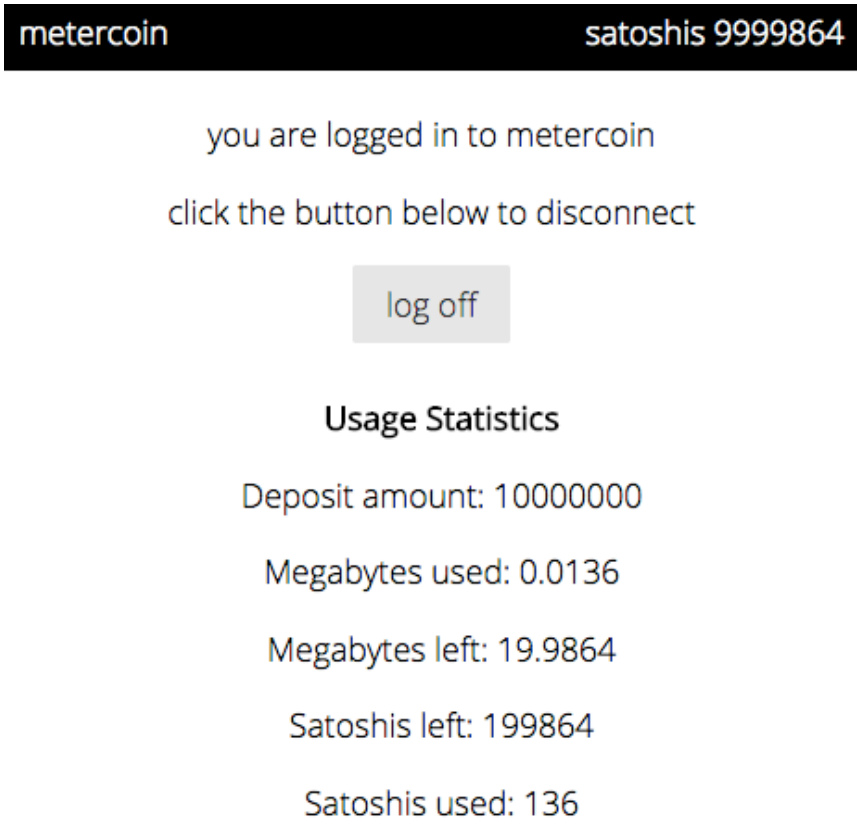

Figure 5: Logged in view

Back-End

The back-end of our implementation is written in Node.js. We utilized the Express framework. The Bitcoinjs and HelloBlock libraries were used to speed up development, so we did not need to interface with the Bitcoin testnet at the protocol level.

The Router

The router is a Linux machine with two network interfaces, one acting as the hotspot for clients and the other connected to the internet. Before establishing a channel, a client is only given access to the daemon on the router itself, the private key server, and a few independent Bitcoin nodes that allow the client to independently verify and access the state of the blockchain. After the channel is established, the client is given unrestricted network access. The access is controlled and monitored by simple scripts which use Linux's IPTables firewall application.

Future Work

Going forward, additional work must be undertaken to make our implementation production ready. The biggest challenge is designing a secure mechanism to enable anybody with bitcoins to open up a micropayment channel with our router in order to purchase bandwidth via metered payments. One approach would be to implement micropayment channel capabilities in open source wallet software. We could also build out our own web wallet.

User research and user testing fell outside the scope of our project. Going forward, that work would be valuable and the results would inform future iterations of our user interface design.

Additional research is waiting to be done to fill in the gaps of our history of micropayment schemes.

Bibliography

Carbunar, Bogdan, Yao Chen, and Radu Sion. "Tipping pennies? privately practical anonymous micropayments." *Information Forensics and Security, IEEE Transactions on* 7.5 (2012): 1628-1637.

Chen, Yao, Radu Sion, and Bogdan Carbunar. "XPay: Practical anonymous payments for Tor routing and other networked services." *Proceedings of the 8th ACM workshop on Privacy in the electronic society*. ACM, 2009.

Geidner, Nick, and Denae D'Arcy. "The effects of micropayments on online news story selection and engagement." *New Media & Society* (2013): 1461444813508930.

Georgescu, Cristian. "Simulating Micropayments in Local Area Networks."*Procedia-Social and Behavioral Sciences* 62 (2012): 30-34.

Greenberg, Andy. "Visa, MasterCard Move To Choke WikiLeaks." Forbes. Forbes Magazine, 07 Dec. 2010. Web. 06 May 2015.

Hauser, Ralf, Michael Steiner, and Michael Waidner. *Micro-payments based on iKP*. IBM TJ Watson Research Center, 1996.

Higginbotham, Stacey. "Check out IBM's Proposal for an Internet of Things Architecture Using Bitcoin's Block Chain Tech." Gigaom, 09 Sept. 2014. Web. 06 May 2015.

Jain, Mohit, Siddhartha Lal, and Anish Mathuria. "A SURVEY OF PEER-TO-PEER MICROPAYMENT SCHEMES."

Jutla, C., and M. Yung. "PayTree." *Amortized-signature" for flexible micropayments", in Second USENIX Workshop on Electronic Commerce, Oakland CA*. 1996.

Karame, Ghassan O., et al. "Microcomputations as Micropayments in Web-based Services." *ACM Transactions on Internet Technology (TOIT)* 13.3 (2014): 8.

Karame, Ghassan O., Aurélien Francillon, and Srdjan Čapkun. "Pay as you browse: microcomputations as micropayments in web-based services."*Proceedings of the 20th international conference on World wide web*. ACM, 2011.

Kasmir, Presty, and Sakhi S. Anand. "Survey on Routing in Multihop Wireless Networks Using Micropayment Schemes." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 2.2 (2013): pp-362.

Micali, Silvio, and Ronald L. Rivest. "Micropayments revisited." *Topics in Cryptology—CT-RSA 2002*. Springer Berlin Heidelberg, 2002. 149-163.

Nakamoto, Satoshi. "Bitcoin White Paper."

Ntantogian, Christoforos, Dimitris Gkikakis, and Christos Xenakis. "PRIPAY: A Privacy Preserving Architecture for Secure Micropayments." *ICSNC 2012, The Seventh International Conference on Systems and Networks Communications*. 2012.

Quibria, Nasreen. "The contactless wave: A case study in transit payments."*Federal Reserve Bank of Boston* (2008).

Reddy, Sasank, et al. "Examining micro-payments for participatory sensing data collections." *Proceedings of the 12th ACM international conference on Ubiquitous computing*. ACM, 2010.

Rivest, Ronald L. "Electronic lottery tickets as micropayments." *Financial Cryptography*. Springer Berlin Heidelberg, 1997.

Rivest, Ronald L. "Peppercoin micropayments." *Financial Cryptography*. Springer Berlin Heidelberg, 2004.

Rivest, Ronald L., and Adi Shamir. "PayWord and MicroMint: Two simple micropayment schemes." *Security Protocols*. Springer Berlin Heidelberg, 1997.

Tang, Lei. "A set of protocols for micropayments in distributed systems."*Proceedings of the First USENIX Workshop on Electronic Commerce, USENIX*. 1995.

van Someren, Nicko, et al. "Does anyone really need micropayments?."*Financial Cryptography*. Springer Berlin Heidelberg, 2003.

"Working with Micropayment Channels." Working with Micropayment Channels. N.p., n.d. Web. 06 May 2015. <https://bitcoinj.github.io/working-with-micropayments>.

Yang, Beverly, and Hector Garcia-Molina. "PPay: micropayments for peer-to-peer systems." *Proceedings of the 10th ACM conference on Computer and communications security*. ACM, 2003.

Zhanikeev, Marat. "Coins in cloud drives can use OAuth for micropayments and resource metering alike." *Proceedings of The Ninth International Conference on Future Internet Technologies*. ACM, 2014.

Zongkai, Yang, Lang Weimin, and Tan Yunmeng. "A new fair micropayment system based on hash chain." *e-Technology, e-Commerce and e-Service, 2004. EEE'04. 2004 IEEE International Conference on*. IEEE, 2004.