Rethinking Mobile Money Security for Developing Regions

Kiron Lebeck, Temitope Oluwafemi*, Tadayoshi Kohno, Franziska Roesner
University of Washington

Technical Report UW-CSE-2015-12-01

ABSTRACT

Throughout the developing world, countless people live without access to basic financial services, and physical banking infrastructure is often inaccessible. These constraints have led to the rise of mobile money, or branchless banking, systems that offer financial services to people who may not have access to conventional banks. Despite their utility, these services do not always integrate strong computer security principles into their designs. In this work, we explicitly delineate security and functionality goals for these systems, and we explore new directions for secure branchless banking towards achieving these goals in the context of unique developing world challenges. We introduce Braavos, a mobile money system that meets these goals by combining existing primitives in novel ways (namely cryptocurrencies like Bitcoin, secure hardware, and data-over-audio communication). In addition to securely providing common branchless banking functionality, our approach enables new functionality such as secure offline transactions. We ground our exploration in a concrete architecture and prototype implementation.

1. INTRODUCTION

The economic conditions in many parts of the developing world have led to the rising popularity of *mobile money* — or *branchless banking*[1] — services that allow users who lack access to traditional financial institutions to use mobile phones as instruments for exchanging funds. People use these systems for a range of functions, including savings and interpersonal transactions [30]. Mobile phones are relatively ubiquitous throughout parts of the developing world [14], and given the benefits of mobile money, it is perhaps unsurprising that mobile money services like M-Pesa [36] in Kenya and Eko [21] in India have millions of users.

Mobile money systems are thus a critical class of mobile systems, with an important role and high adoption in the developing world. Unfortunately, there are identified vulnerabilities and hence known security risks in today's mobile money systems [6, 39, 45]. Some of these issues have manifested in the wild: SMS spoofing has led to the loss of tens of thousands of dollars by M-Pesa agents, and caller-ID spoofing presents a real threat to end users [37, 42]. Motivated by the existence of such vulnerabilities, prior work [39] called for the integration of stronger end-to-end security protections into the designs of mobile money systems. Recent work [45] also identified systemic security weaknesses in Android mobile money applications used in the developing world, exemplifying the inability of today's systems to adequately protect users and highlighting the dire need to explore new directions in mobile money security.

Motivated by the importance of mobile money systems, we ask whether there are new ways to design these systems — ways that might leverage modern computer security and cryptography research — to derive solutions resilient to the threats that plague existing approaches. In exploring this question, we find that it is not only possible to create a mobile money system that meets the security goals that most users might desire, but that a fresh approach can enable new functionality unachievable in existing systems (such as the ability to securely conduct transactions offline).

To conduct this inquiry, we rigorously evaluate the threat model and concretely define security goals for mobile money systems — goals that, with some exceptions [39], were largely implicit to date. In security, concrete definitions are critical to create a basis for design and evaluation. Innovation in the mobile money space is complicated, however, by numerous challenges unique to the developing world. For example, many users have only basic "dumb" phones without data or Internet capabilities, and several users (e.g., a family) may share a single phone. Ecosystems of novice users with shared and out-of-date phones are also more likely to be vulnerable to attack, requiring that a mobile money system provide security even in the face of potentially untrustworthy phones. We discuss these and other challenges in Section 2.1.

Our Approach and Key Insights. To address the shortcomings of existing solutions, we introduce Braavos, a secure mobile money framework that leverages secure trusted hardware, secure cryptocurrencies (specifically Bitcoin, in our prototype), and data-over-audio. Our key insight is that, by identifying established primitives and combining them in novel ways, we can design a secure system that not only overcomes the above challenges, but also enables unique new capabilities (e.g., secure offline transactions).

Cryptocurrencies. A key challenge for mobile money systems is how to ensure the authenticity and integrity of transactions, for which basic communication encryption (e.g., TLS)

*This author is now at Intel.

[1]"Branchless banking" refers to financial distribution strategies that do not depend on physical bank branches, and "mobile money" refers to branchless banking schemes that leverage mobile phones for conducting transactions. However, these terms are often used synonymously (e.g., [45]), and we likewise use them interchangeably.

is insufficient. We observe that cryptocurrencies provide important security benefits that can be leveraged to combat the risks present in existing mobile money systems, including providing transaction authenticity and integrity. For our prototype, we use Bitcoin [38] as a tool for reasoning concretely about how a mobile money system could integrate cryptocurrencies. There has also been increasing interest in the potential for Bitcoin adoption in developing regions [27], especially for those with weak currencies and unreliable banks [13, 25], further motivating our choice of Bitcoin as a relevant case study.

Secure hardware. The developing world context brings additional challenges to secure mobile money systems: a user's device may be untrustworthy and devices may be shared among users. Similar to prior work for mobile health applications [48], we propose equipping users with small trusted (and trustworthy) hardware dongles that interface with phones to conduct transactions. By decoupling and isolating users' funds from their phones, the dongles both secure transactions against modification by malicious phones (i.e., allowing encrypted transactions to be tunneled through untrustworthy phones) and separate ownership of funds from phone ownership. We discuss models for distributing these dongles as well as form factor considerations in Section 8.

Data-over-audio. Mobile devices in the developing world often come in simpler, less expensive form factors, such as "dumb" or feature phones, rather than smartphones. Thus, it is critical that a mobile money system interface with different and basic devices, and a traditional smart phone application would not satisfy our design goals. Braavos leverages data-over-audio, an established technology (e.g., [3]). To use Braavos, the user plugs a dongle into the audio jack of a phone, calls a phone number, and creates a transaction on the dongle; the dongle can then exchange data, via audio, with the party at the other end. Data-over-audio enables Braavos to work with any phone, including dumb phones, and it has the potential for higher bandwidth than SMS [19].

Summary and Contributions. Despite their importance, current mobile money systems suffer from systemic security issues. Existing systems have known vulnerabilities, and while security best practices could improve these systems, such practices are not sufficient. Our goal, therefore, is to explore novel approaches to securely enabling both traditional mobile money operations as well as new functionalities, such as the ability to securely conduct transactions offline and recover funds. Specific contributions include:

1. We explicitly define *security and functionality goals* for secure branchless banking in the developing world, and we consider these goals in the broader context of *unique developing world challenges*. Security definitions are valuable to provide a concrete basis for design and evaluation.
2. We propose and explore new directions for secure branchless banking by *identifying and combining existing primitives* (cryptocurrencies, secure hardware, and data-over-audio) *in novel ways*.
3. We introduce *new secure functionality*, namely offline transactions and wallet recovery, and explore methods to achieve these new functionalities.

To better inform our new directions and surface potential design challenges, we also present a concrete architecture and prototype implementation, which we call Braavos.

2. BACKGROUND AND MOTIVATION

We begin with background information and then specify our target goals for secure mobile money systems in the developing world. Because concrete goals are critical for the structured analysis of a system, we view the explicit delineation of these goals as a contribution of this paper. We then review limitations of existing mobile money systems with respect to our goals to further motivate our inquiry.

2.1 Context

Branchless Banking. Increased access to mobile devices in developing regions, combined with inadequate financial infrastructure, has spurred the growth of branchless banking services. Common use cases for branchless banking include *remittances*, in which someone transfers money to a recipient in his or her hometown or country; *person-to-person* transactions, in which two co-located people exchange currency for goods or services; and *savings*, in which someone stores money over a longer period of time in the system [30].

ICTD Challenges. An entire research field, known as Information and Communication Technologies for Development (ICTD), focuses on technologies for the developing world. Many research efforts in this field aim to achieve technical goals under the resource or social constraints in different developing regions. Key challenges we face in designing a secure branchless banking system include:

- *Phone-to-user ratio.* Some developing regions see different ratios of phones to users. For example, some users may have multiple phones for different carriers [4], or multiple users (such as multiple family members) may all share a single phone [4, 46].
- *Type of phone.* While some users in developing regions may have smartphones, many users still use so-called "dumb" or feature phones — phones that only support basic call or SMS operations, or that do not support application platforms like Android or iOS [41].
- *Trustworthiness of phones.* Due to the use of shared and older phones, it may not be possible for users to trust their phones. For example, even if a user purchases a new phone, that user may not regularly update it over time. Once the phone is no longer supported by the manufacturer, it may not receive vulnerability patches and hence may become untrustworthy.
- *Connectivity type and cost.* We cannot assume that users have direct access to the Internet. As recently highlighted by the Internet.org effort [29], users may not have free access to the Internet — in fact, data plans can be prohibitively expensive [20], so some users may only have access to SMS and voice calls.
- *No connectivity at all.* There may be times when users have no connectivity at all. We desire a secure mobile money system that allows person-to-person transactions between physically co-located individuals even when there is no cellular connectivity.

These challenges significantly complicate the design of a secure mobile money system for the ICTD context. While we focus on exploring secure mobile money alternatives for the developing world, we observe that — as with other ICTD systems — a solution under the above constraints may also

provide benefits or lessons in developed world contexts (e.g., providing secure mobile banking functionality in the face of potentially compromised smartphones).

Cryptocurrencies. Cryptocurrencies are digital currency schemes that employ cryptographic mechanisms to generate currency units and validate transactions. A key insight of this work is that cryptocurrencies can be leveraged to improve mobile money security. We focus our discussions on Bitcoin [38], a recently popularized cryptocurrency, and we briefly introduce relevant Bitcoin vocabulary[2] here. Users have *Bitcoin wallets* corresponding to a public-private key pair and an *address* used to receive funds. They exchange bitcoins via *transactions* that are cryptographically signed with a user's private key, propagated via the peer-to-peer *Bitcoin network*, and recorded in a public, distributed ledger called the *blockchain*.

Bitcoin provides a number of security benefits. For example, a user's transactions are authenticated using cryptographic signatures and are unforgeable without knowledge of the user's private key. We provide additional background on Bitcoin throughout the paper as necessary, discussing its security properties in Section 4 as well as its possible disadvantages in Section 8. Bitcoin allows us to begin a concrete dialogue on the possible role cryptocurrencies could play in improving mobile money security, but other cryptocurrencies are available, many with slightly different properties.

2.2 Goals for Secure Branchless Banking

Building in part on prior work [39], we explicitly delineate a set of **security goals** for mobile money systems that were largely implicit to date, and we summarize the ways in which Braavos achieves them in Figure 1. These goals can serve as the basis for the design and evaluation of future systems.

1. *Remove Trust from Phones:* The user's mobile phone may be running untrusted applications or be otherwise compromised (e.g., even if a mobile money app uses cryptography best practices, the phone could still be rooted). Thus, a desirable goal for branchless banking is to limit or eliminate trust from a user's phone.
2. *Resilience to Device Theft:* Physical theft of a user's mobile money device (e.g., phone) should not automatically result in loss or theft of money.
3. *User-Authorized Transactions:* A mobile money system should provide robust mechanisms to minimize the risk of other parties creating or otherwise authorizing transactions on behalf of users.
4. *Transaction Authenticity, or Resistance to Spoofing:* A user's transactions should be authenticated, and messages between users and the mobile money service should not be spoof-able. For example, an adversary should not be able to spoof a message tricking a user into believing a payment has been made when it has not.
5. *Transaction Integrity:* A user's transactions should be protected from modification by other parties.
6. *Robust to Malicious Users:* A mobile money service should be robust to attempts by malicious users to subvert the system or conduct fraudulent transactions.

To achieve widespread adoption, a mobile money system must also support a set of desired functionality. A given system may not provide all of these features, but we extend our list with a set of possible **functionality goals**:

7. *Broad Compatibility:* A mobile money solution that is broadly compatible with a wide range of existing mobile devices will increase its likelihood of adoption.
8. *Support for Branchless Banking Operations:* A useful branchless banking solution must support a variety of operations, including long-distance remittances, in-person transactions, and long-term savings.
9. *Support for Limited Connectivity Types:* When the mobile money system must communicate with remote entities, it should be able to do so over the minimum-available connectivity type (cellular audio).
10. *Funds Recovery:* Mobile money users should be able to recover their funds in the event of device loss or theft.
11. *Usability:* Finally, a mobile money system must be usable to gain widespread adoption, which may depend on the specific implementation. At a minimum, it should be easy for users to execute financial transactions, locate and/or add contacts, etc. Though we do not directly evaluate Braavos's usability in this paper, we are mindful of it in our design and aim to match the usage model of M-Pesa, which is widely deployed.

Traditional mobile money services tend to support this set of functionality. However, our framework is unique in that it allows us to securely provide the following additional (and somewhat surprising) functionality:

12. *Offline Transaction Support:* Since mobile communication networks in developing regions may have limited or intermittent coverage, a desirable feature of mobile money systems would be support for offline transactions, allowing co-located individuals to transfer virtual currency without requiring network connectivity. To the best of our knowledge, no existing mobile money solutions achieve this goal.

Another possible goal is *Transaction Privacy,* or the protection of a user's transactions and/or transaction metadata from eavesdropping or subsequent disclosure. Panjwani notes that transaction metadata privacy is non-trivial under today's typical branchless banking model [39]. Transaction privacy is also non-standard with cryptocurrencies, e.g., while currencies like Zerocash support transaction privacy [7], the most common cryptocurrency (Bitcoin) does not. Hence, Braavos also does not target transaction privacy.

2.3 Limitations of Existing Systems

Unfortunately, we observe that today's branchless banking options do not meet many of the above security goals. These weaknesses have been observed both conceptually (e.g., [39]) and in the wild (e.g., [45]). Notable weaknesses include:

- Implicit trust placed in a (potentially compromised) phone, violating our *Remove Trust from Phones* goal.
- Vulnerability to spoofing and poor authentication of messages, violating our *Transaction Authenticity* goal.
- Non-existent or faulty application layer cryptography and reliance on weak network-layer encryption, which violates our *Transaction Integrity* goal.

In ecosystems rife with outdated phones and SMS-based services, applying security best-practices is non-trivial, and developers often fail to guard against weaknesses such as those above [45]. Furthermore, simply applying best-practices

[2]Note that "Bitcoin" refers to the protocol, while "bitcoin" refers to a unit of the currency.

Branchless Banking Goals	**Braavos**	*Approach*
Remove Trust from Phones	✓	Trusted hardware dongle.
Resilience to Device Theft	✓	Dongle requires authentication.
User-Authorized Transactions	✓	Only trusted hardware dongle holds Bitcoin wallet.
Transaction Authenticity	✓	Signatures on Bitcoin transactions.
Transaction Integrity	✓	OTR for communications, signatures on Bitcoin transactions.
Robust to Malicious Users	✓	Dongle and service help mitigate double spending attempts, and transaction confirmations are authenticated by service.
Transaction Privacy	p	OTR prevents eavesdropping on message content, but transactions ultimately public in blockchain.
Broad Compatibility	✓	Dongle supports any phone with an audio input.
Support for Branchless Banking Operations	✓	Operations described in Section 3.
Support for Limited Connectivity Types	✓	Dongle can communicate over audio channel.
Funds Recovery	✓*	Design described in Section 5.
Offline Transaction Support	✓*	Design described in Section 5.

Figure 1: This table overviews whether and how Braavos meets the goals set in Section 2. Items marked p are partially supported by Braavos; items marked ✓* are included in our design but not prototyped. We omit usability from the table because — though we aim to match M-Pesa's usage model — we did not evaluate it directly.

to existing systems is insufficient to guard against attacks like SMS spoofing (which has cost M-Pesa agents substantial sums of money [37]) and caller ID spoofing [42]. Though the content of SMS messages could be encrypted or cryptographically signed, verifying these signatures and securing cryptographic keys may not be possible under ICTD constraints (e.g., the use of "dumb" phones or minimizing the number of sent SMS messages required) and may also not be backwards compatible.

Additionally, organizations like BitPesa [9] and Kipochi [32] have proposed Bitcoin-based mobile money solutions targeted at developing regions, typically for deployment or cost reasons rather than the security properties we consider here, and thus they often fall short from a security standpoint. In particular, they employ the *hosted wallet* model, meaning they store users' Bitcoin private keys on their servers and construct transactions on behalf of users. This model allows users to initiate transactions via short SMS messages (themselves violating the *Transaction Authenticity, Privacy, and Integrity* goals), but in doing so, it gives services the power to transact on behalf of users without their knowledge or consent, violating our *User-Authorized Transactions* goal.

3. SYSTEM OVERVIEW

Existing mobile money solutions exhibit myriad shortcomings and often fail to adequately protect users. Thus, we ask: given the challenges present in the ICTD context, what techniques can we apply and integrate in novel ways to meet the above goals? We introduce Braavos, our system that explores new directions for protecting mobile money users.

3.1 System Components

Figure 2 gives an overview of Braavos. Braavos is intended to support common mobile money functionality, including creating and sending transactions to other users, maintaining and checking a current balance, as well as depositing money into and withdrawing money from the system. As with many contemporary mobile money systems, the Braavos ecosystem consists of the following components:

- *User*: The user is a person who uses the Braavos system to store, send, or receive funds.
- *Phone*: A mobile phone used by the user, which may be a dumb, feature, or smart phone.
- *Service*: A third party that users interface with to save or transfer funds.
- *Agent*: As in systems like M-Pesa, a human intermediary who facilitates cash deposits into and withdrawals out of the system.

In M-Pesa and similar systems, a user uses his or her phone to directly interface with a service via SMS messages. Unfortunately, SMS messages are easily spoofed (as in real attacks on M-Pesa [37]) and require trusting the user's phone (violating our *Remove Trust from Devices* goal). Our goal is to facilitate similar functionality as that supported by existing mobile money systems, but more securely. However, as exemplified by existing failures, it is challenging to achieve this goal in an ecosystem of untrusted mobile devices and weak channel security. Braavos explores the use of the following additional component:

- *Dongle*: A small, dedicated hardware device with user input and output capabilities that interfaces with the phone's audio and microphone port. Dongles are manufactured and distributed by the service and act as hardware interfaces for conducting financial operations.

We discuss possible models for dongle deployment in Section 8, but note here that we are not the first to propose auxiliary hardware devices in an ICTD context (e.g., [15]). Other existing uses of secure auxiliary hardware include banks distributing authentication devices to users [28], second-factor authentication hardware ("security keys") advocated for by the FIDO Alliance [24], and academic work leveraging low-cost smartcards to secure medical applications on untrusted mobile phones [48]. Secure smartcards cost around $5-10 per card for orders of 500 [47], and as prices continue to drop or bulk ordered quantities increase, their viability for low-income users will rise.

We now provide an intuition for how Braavos works. To perform a transaction, a user connects his or her dongle to a phone's audio jack. The user then selects an appropriate operation on the dongle, e.g., to initiate a funds transfer, and then uses his or her phone to call a number for the service. After the user presses an appropriate button on the dongle, the dongle will communicate with the service over the cellular audio channel.

One of our key insights is that, by leveraging the audio channel, Braavos can interface with a wide range of phones with the potential for higher bandwidth than SMS, thereby

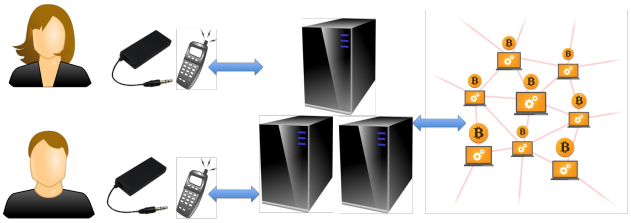

Figure 2: Braavos system overview. From left to right: users plug dongles into their phones via the audio jack, allowing dongles to communicate with the service over the cellular audio channel; the service communicates with the global Bitcoin network.

supporting users with even basic phones that lack Internet connectivity or for whom Internet connectivity is costly. We note, however, that the dongle could also communicate via audio to an Internet-connected smartphone app, similar to the mobile payments platform Square [1]. Beyond removing trust from the phone, the dongle allows a user to maintain a single balance across multiple phones (i.e., by plugging a single dongle into different phones), addressing the unique ICTD challenge of heterogeneous phone usage models.

Although largely transparent to the user, there is one final component in our system:

- *The Bitcoin Network*: Braavos uses the larger Bitcoin network to facilitate storage and transfer of funds.

Recall that part of our goal is to explore the use of cryptocurrencies as an enabler for secure mobile money. We chose to base our design on Bitcoin given its level of maturity, use, and study in both academia and industry. Our design builds on Bitcoin's relevant security properties, e.g., the fact that a user's transactions cannot be forged without knowledge of the associated private key.

3.2 Protocol Overview

In exploring new approaches to improving mobile money security, we must ensure that our solution still provides common mobile money functionality expected by users. We briefly overview the Braavos protocols here, which mirror traditional branchless banking operations. We then dive more deeply into our approach for instantiating them in Section 4, and we discuss user interface considerations in Section 8. A user may be involved in the following core Braavos operations:

- *Enrollment*: A user receives a dongle and enrolls in the Braavos system, first establishing an authentication mechanism with the dongle (e.g., a PIN or fingerprint) that is used to initiate all future operations. The user then calls the service phone number, and the dongle and the service exchange information.
- *Create contact:* A user adds a new contact to his or her dongle, associating a human-memorable identifier with a system identifier.
- *Send funds*: Once enrolled, a user can plug the dongle into any (untrusted) phone to send funds to another user of the system. To do so, the user enters into the dongle (1) the recipient's identifier, and (2) the desired amount of money to transfer. The user then calls the service and presses a button on the dongle, which sends the transaction to the service. Upon receipt of the transaction, the service sends a short confirmation message back to the user to acknowledge the transfer.
- *Receive funds*: Once the service receives a transaction for a particular user, it notifies the user via a voice-mail or SMS. The user then calls the service, plugs the dongle into his or her phone, and selects a "receive funds" option on the dongle. The service subsequently transfers new transactions to the user's dongle.
- *Check balance*: The user's dongle tracks his or her balance and transaction history locally, so the user can check the balance with a button press (i.e., it does not require contacting the service, though the local balance may not account for all incoming transactions).

A user can use the receive and send funds protocols for depositing and withdrawing money, respectively. To deposit funds, the user hands cash to an agent, who creates a transaction to pay the user and sends it to the service from his or her own dongle. The user then gets a callback message and follows the receive funds protocol. To withdraw funds, the user pays an agent using the send funds protocol, and receives cash upon completion. Since the network is untrusted, as we discuss below, it is important for the user and the agent to complete their protocols and conclude business only after receiving confirmations from the service.

3.3 Threat Model

Understanding not only how the components of a mobile money system interact, but the extent to which they trust each other, is critical to move towards a more secure solution. Before diving deeper into the protocols laid out above, we take a step back and consider our threat model, organized according to the components of our system.

User. We assume users will act in their own interests — that is, a user may attempt to exploit the system, e.g., by defrauding agents or conducting fraudulent transactions.

Phone. The user's phone is untrusted by all parties, including the user. We assume that a user's phone — or a malicious application installed on it — can intercept and modify user input as well as incoming and outgoing data.

Service. First, users trust the service to manufacture and distribute dongles (e.g., through in-person distribution). Users also trust the service to process transactions and forward them from dongles to the Bitcoin network, to provide confirmation messages, and to report new incoming transactions. Beyond that, users do not have to trust the service, and in particular, users do not trust the service to store their Bitcoin private keys and construct transactions on their behalves. However, at various design points, we discuss how Braavos can support several trust/convenience tradeoffs.

Though this model allows the service to execute some denial-of-service (DoS) attacks, these attacks are easily audited and detected by users, e.g., when recipients do not receive confirmations of the intended payments. Such attacks are also not unique to Braavos — many third parties, such as ISPs or cellular providers, have the ability to mount DoS attacks. Additionally, since the service does not store users' private keys, it cannot create arbitrary fraudulent transactions, and a single breach of the service would not result in bulk theft of these keys.

Agent. Human agents, common in developing world mobile money systems, cash users into and out of the system. However, users and the service do not necessarily trust agents to report accurate information; thus, receipts for cash-in and cash-out transactions are communicated directly between users and the service over an authenticated channel.

Dongle. The trusted hardware dongle is the key secure component of our system, and we assume that its core logic is tamper-resistant (e.g., similar to a commodity smart card). That is, we assume that even the user cannot extract the Bitcoin private key from the dongle. It is possible that the user or another person could tamper with the dongle's I/O (e.g., to modify or eavesdrop on input). The user would gain nothing from doing so, and as mentioned above, we trust the service to distribute legitimate dongles to users. Nevertheless, the dongle could potentially provide a simple tamper-evident mechanism to alert users to modification.

Given those assumptions, the dongle is trusted by the user to report information (e.g., current local balance) and to process input and transmit data correctly, as well as by the service, which trusts it to conduct only legitimate transactions on the user's behalf. Agents also trust the dongle.

Bitcoin Network. The user and service trust the Bitcoin network and protocol as much as any Bitcoin user trusts them. We discuss additional Bitcoin details in Section 4.

Cellular Network. The users, the service, and the agents do not trust the cellular network. The network may observe, modify, and prevent the delivery of messages.

Non-Goals. To gain unauthorized access to a user's dongle, an adversary would need both physical access to the dongle *and* the ability to authenticate as the user (e.g., by gaining knowledge of the user's PIN). This type of threat is present for many devices, such as password protected computers or smartphones, and thus we do not attempt to guard against such attacks.

Braavos is also not designed to provide complete *transaction meta-data privacy*, just as Bitcoin does not provide transaction meta-data privacy. Although messages between the dongle and the service are encrypted, Braavos makes no effort to hide the existence of these messages, and completed transactions are eventually pushed to the Bitcoin network.

4. CORE PROTOCOLS

Before diving deeper into the ways Braavos integrates existing primitives in non-traditional ways to confer stronger security properties on core mobile money operations, we discuss the Bitcoin and encryption protocols that it leverages.

Bitcoin. Recall that a user's Bitcoin *wallet* contains a private key, which is used to derive an *address*, an identifier for the wallet. A Bitcoin transaction takes as input one or more addresses for recipients, the value to pay to each address, and a set of previous unspent transaction outputs owned by the user. That is, Bitcoin transactions are *chained* together — unlike in ordinary banking, a user does not have a balance of funds in the traditional sense; rather, he or she has unique transaction outputs available to spend. To spend those outputs, the user must possess the associated private key. Transactions are verified in a peer-to-peer manner by nodes in the network called *miners*, which perform a *proof-of-work* to record them in the *blockchain*. Bitcoin miners take small transaction fees in exchange for this service.

Bitcoin allows us to reason concretely about the properties of cryptocurrencies that mobile money systems may benefit from. Specifically, Bitcoin exhibits several desirable security properties of interest for Braavos:

Unforgeable transactions. Without knowledge of the private key associated with a Bitcoin wallet, fraudulent transactions originating from that wallet cannot be created. In Braavos, we leverage this by limiting knowledge of the user's Bitcoin private key to the trusted hardware dongle, removing the untrusted phone and the semi-trusted service's (see Section 3.3) ability to conduct transactions on the user's behalf.

Authenticated transactions. Bitcoin transactions are signed, thereby authenticating the corresponding wallet's owner.

Unique transactions. Bitcoin transactions are cryptographically unique, providing users an inherent defense against replay attacks: even if a transaction is sent or received multiple times, it will only be credited to a user's wallet and documented in the blockchain once. Although we could build additional replay defenses, Bitcoin allows us to leverage existing properties of the underlying currency scheme.

Resilience to double spending attacks. In a double spending attack on a digital currency, malicious users attempt to spend the same virtual "coins" in multiple transactions. Bitcoin is designed to reduce the likelihood of such attacks: as long as greater than some fraction of the network is honest, only one of the spending attempts will be confirmed in the blockchain. However, it can take upwards of 60 minutes for a transaction to be reliably confirmed, which is not ideal for in-person transactions with merchants. We discuss below how Braavos can support fast transaction confirmations, but we also note here that prior work has proposed techniques for quickly detecting double spending attempts to better facilitate fast payments [31].

Secure Channels. Although Bitcoin transactions are signed, not all messages in Braavos are transactions (e.g., confirmations from the service, or information requests from dongles). Therefore, Braavos must be able to authenticate non-Bitcoin messages and ensure they have not been modified in transit. To provide a secure communication channel, Braavos employs the *Off the Record Messaging*, or OTR, encryption protocol [12]. We chose OTR because it has been reviewed by the cryptography community and provides a relatively lightweight mechanism for authenticating messages. OTR is also ideal for intermittent, short-lived connections, such as those in a mobile money system, when compared to alternatives like SSL/TLS. Along with authentication, OTR provides additional benefits such as confidentiality and perfect forward secrecy in the face of network attackers, preventing eavesdropping of Braavos-related messages in transit. (Note, however, that Bitcoin transactions are eventually made public in the blockchain, so Braavos provides transactions confidentiality only in transit to the service.)

4.1 Protocol Operations

While the above mechanisms provide useful security properties, it is not necessarily intuitive how to incorporate them into a mobile money system to actually reap these benefits. We thus describe below how Braavos integrates these mechanisms to securely support the protocols highlighted in Section 3.2. These operations are summarized in Figure 3.

We intentionally describe these protocols in a way that is agnostic to the underlying communication channel. While our implementation uses audio signaling over a cellular channel to maximize deployment potential, the core protocols are not tied to the underlying transport. For example, if the user had a smartphone with Internet connectivity, then an untrusted app could ferry the encrypted and authenticated

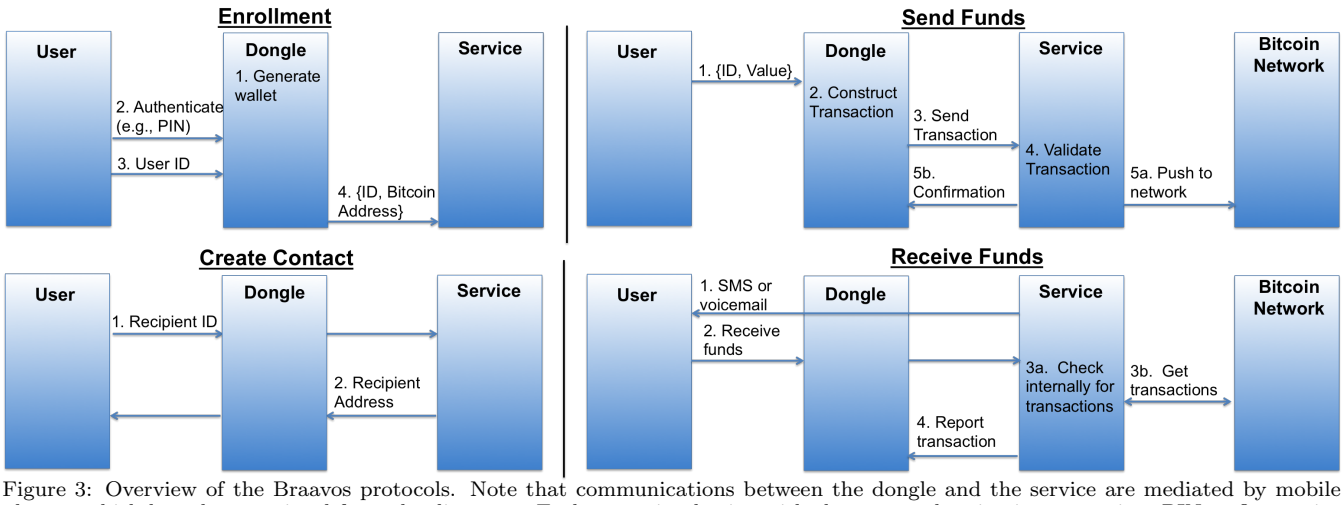

Figure 3: Overview of the Braavos protocols. Note that communications between the dongle and the service are mediated by mobile phones, which have been omitted from the diagrams. Each operation begins with the user authenticating, e.g., via a PIN or fingerprint scan, and communications are encrypted with OTR.

data between the dongle and the service over IP.

Dongle Setup. We assume that the dongle is manufactured in a trusted fashion by a trusted party, similar to how smartcards used in the banking industry are fabricated today. Before a user receives his or her dongle, that dongle is set up as follows. First, to prevent counterfeit dongles, the dongle comes pre-configured with the *service's public key and a public-key certificate signed by the service*, which it uses to authenticate itself to the service and to other dongles. If the service's key is compromised, Braavos must revoke those credentials on dongles. In our design, each dongle comes pre-configured with multiple service keys. If one key gets compromised, the service can switch to the next one and use it to supply additional keys. We assume that the service follows best practices in protecting its private keys.

The dongle next performs an OTR *authenticated key exchange* with the service to establish a secure channel. Unless otherwise noted, subsequent communications between the dongle and service are encrypted via OTR.

Enroll. The primary objective of enrollment is for the user to establish an account with the service.

1. The dongle generates a Bitcoin wallet, including the private key needed to conduct transactions. This step may happen during dongle fabrication time. (Note that we revisit this design point in Section 5.) The Bitcoin keys generated in this stage are distinct from the OTR-related keys maintained by the dongle.
2. The user receives the dongle and establishes an authentication mechanism (e.g., PIN)
3. The user enters an ID (e.g., a username) on the dongle, which is sent to the service. If the ID is already in use, the service informs the user to select another.
4. The user's {ID, Bitcoin Address} is sent to the service.

Upon completion of enrollment, the service has learned the {ID, Bitcoin Address} pair of a particular user.

Create Contact / Request Address. Before sending funds, the user must learn the recipient's Bitcoin address. Under our threat model, the user can instruct the dongle to query the service for the address corresponding to a recipient ID. This approach maximizes usability at the expense of additional trust in the service. Under a stronger threat model, the user could instead enter a recipient's Bitcoin address by hand and bypass the service for this operation.

Send Funds. In the send funds protocol:

1. The user selects a recipient ID from a list of contacts on his or her dongle and enters a value to pay.
2. The dongle verifies that the user has enough funds by checking previously retrieved Bitcoin transaction outputs, loads the corresponding Bitcoin address for the target recipient, and constructs a Bitcoin transaction.
3. The dongle sends the transaction to the service and logs it locally. If a transaction is dropped in transit, the user can simply send the same one again without creating a new one. Since transactions are unique (as discussed previously), there is no danger of accidentally losing money by sending the same one again.
4. The service validates the integrity of the transaction.
5. The service publishes the transaction on the Bitcoin network and sends a confirmation message back to the user to acknowledge receipt of the transaction.

For the last step, the service sends the confirmation to the user immediately after pushing the transaction to the Bitcoin network. It may, however, take an hour or more for the transaction to be reliably confirmed in the blockchain. The recommended usage model for Bitcoin requires the user to wait for multiple blockchain confirmations, since there is a risk of double spending attacks in the meantime. Our design mitigates such attacks by leveraging the user's semi-trust in the service and the trustworthiness of the dongle (Section 3.3). Since we assume that Bitcoin private keys cannot be removed from trusted dongles, even a malicious user cannot use the same private key to double spend. Hence, double spending is not feasible under our threat model and our system design described so far.

Receive Funds. The receive funds operation allows a user to load new Bitcoin transaction outputs from the service onto his or her dongle. This involves the following:

1. The service sends the user an unauthenticated voicemail or SMS notification of new funds.
2. The user connects to the service (e.g., via a phone call or smartphone app) and selects a "receive funds" option on his or her dongle, prompting the dongle to send a request for new funds to the service.

3. A user may receive funds from another Braavos user or an external Bitcoin user. That is, anyone with knowledge of a Braavos-user's Bitcoin address can send bitcoins to that address. The service checks if the user has unspent outputs internally, as well as in the blockchain, that have not yet been reported to the dongle.
4. The service sends back the appropriate information. This could be `<tx hash, output index, value>` tuples, or the full, raw transactions. The former option places some trust in the service to report the correct information; the latter removes trust but requires communicating and storing larger amounts of data.

While the initial unauthenticated callback is a convenience, it is not strictly necessary. The user could poll the service to check for new transactions, particularly if he or she is concerned that the callback was blocked or compromised.

Having presented the ways in which Braavos supports traditional mobile money operations, we next consider auxiliary protocols from a conceptual standpoint. We then discuss UI considerations in Section 8.

5. RECOVERY & OFFLINE

As described above, Braavos — like existing mobile money systems — allows users to transfer money only when they have cellular connectivity. Additionally, because Braavos does not reveal a user's Bitcoin private key to either the service or the user, it would seem to bear the same risks as physical wallets or stored value cards (like subway cards): if the dongle (wallet) is lost or stolen, then the funds are gone.

We thus consider extensions to Braavos's core protocols that overcome the above limitations by securely providing functionality not traditionally achievable in today's systems. In particular, we discuss how to support *recovery*, i.e., the (trusted) reconstitution of a user's private key onto another dongle if the original dongle goes missing. We refer to this process as **wallet recovery**, involving participation from both the user and the service to ward against certain threats (defined below).

Additionally, we explore an emergent capability of Braavos — namely, that Braavos can support temporary disconnected operations through an **offline transaction** mechanism. Since dongles use secure hardware and are trusted to track transactions honestly, and since transactions are cryptographic and can be uploaded by any party (including the sender or receiver of a transaction), Braavos allows users to directly conduct dongle-to-dongle transactions, using their existing Bitcoin wallets. These transactions can be uploaded later when the users regain network coverage.

We must ensure that attackers cannot exploit these new operations to double-spend funds, e.g., by maliciously reconstituting a key onto a new dongle and using it for offline transactions while the original dongle is still in use. We first introduce these operations individually, and then discuss how they can coexist without sacrificing security.

Wallet Recovery. Recall that we do not trust the service to keep the user's private key, and that we assume that private keys are never available outside of trusted dongles. Though these design choices prevent the service from conducting fraudulent transactions and the user from conducting double-spending attacks, they complicate wallet recovery. If neither the service nor the user have a copy of the private key, then how can the key be reconstituted on another dongle after the original dongle is lost?

We first propose a solution and then discuss its pros and cons. Our proposed solution involves *both* the service and the user in key (re)generation. Braavos generates private keys collaboratively by combining both user- and service-held secrets, so that neither party can reconstitute a key on its own. Private keys can be generated deterministically by applying a known hashing procedure on those combined secrets. The user's secret could be in the form of a passphrase. The service will *only* send its secret to a trusted dongle (by first verifying that it is communicating with a real dongle), thereby preserving the property that Bitcoin private keys are only ever stored or (re)generated inside the trusted dongles.

The user must remember his or her portion of the secret, which may raise usability concerns. However, Braavos supports a gradient of trust, allowing users to reflect a *spectrum* of security preferences, unlike traditional branchless banking systems that require complete trust. Novice users willing to trust the service may not mind having weak user-side passphrases, whereas security conscious users can select stronger passphrases. A user who prefers for Braavos to more closely emulate a real wallet could, of course, choose not to enable the wallet recovery capability; such a user's keys would never be exposed or reconstituted outside the original dongle, absent a failure of Bitcoin's underlying cryptography or the pseudorandom number generation processes.

Offline Transactions. We assume that Braavos users typically have cellular connectivity, but may occasionally lose that connectivity or find themselves temporarily outside of a coverage area. Therefore, a desirable feature not found in traditional mobile money systems is the ability to conduct transactions offline. Somewhat surprisingly, we find that this ability stems naturally from the unique approach Braavos takes to securing mobile money operations. To support in-person offline transactions, we allow users to conduct dongle-to-dongle transactions by plugging two dongles into each other. Recall that a dongle is built on secure hardware and that we trust the dongle to keep track of the user's balance and create only valid transactions (Section 3.3). Thus, offline transactions are ordinary Bitcoin transactions generated by the dongle and passed from user to user. For example, when Alice conducts an offline transaction with Bob, Alice's dongle generates a valid Bitcoin transaction from her wallet and shares it with Bob's dongle. When Alice *or* Bob return online, this transaction is uploaded to the service and ultimately to the Bitcoin blockchain.

Note that every participant in an offline transaction must store and upload the *entire chain* of offline transactions up to the current transaction. Consider the following example, depicted in Figure 4. Alice pays Bob with an offline transaction, Bob then pays Charlie offline, and finally Charlie attempts to withdraw his funds. Since Alice and Bob might not yet (or ever) go online, Charlie must upload the entire chain of offline transactions from Alice to Bob to himself for the final transaction to be confirmed as valid by the Bitcoin network. If this chain becomes long, the dongle may run out of storage space or the upload may take a long time; to mitigate this, Braavos can limit the length of an offline transaction chain before a user must go online.

To prevent counterfeit dongles from conducting fraudulent offline transactions, transactions sent from one dongle to another are signed with a dongle-specific key; the sending dongle presents to the receiving dongle a public-key certifi-

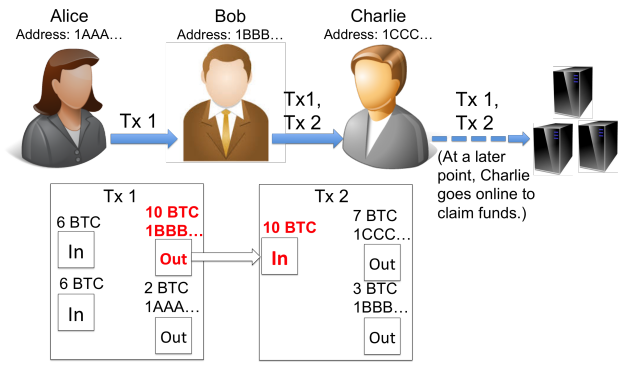

Figure 4: An offline transaction chain, ending with Bob sending Charlie 7 bitcoins. (We use "Tx" as shorthand for "transaction".) Note that (in Bitcoin terms) the input of Transaction 2 corresponds to a particular output of Transaction 1 — thus, for Transaction 2 to be considered valid by the service and by the Bitcoin network, both must know about Transaction 1. Hence, Charlie must upload the entire transaction chain to claim his funds.

cate signed by a trusted service private key.

Supporting *Both* Offline and Recovery. The text above describes how to individually support either offline transactions or recovery. However, these protocols inherently lie in tension with each other, so supporting both without exposing Braavos to double spending attacks is challenging. For example, a user (attacker) could pretend to have lost his or her dongle and reconstitute the private key onto a new dongle, resulting in two active dongles. The user can now use the new dongle in online transactions, while using the old dongle in offline transactions. The Bitcoin network would not observe this double-spend attack until the offline transactions are later uploaded to the Bitcoin network when the recipient goes online. Detection at this point is likely too late, however, if the offline transaction has already resulted in the exchange of goods or services.

We therefore desire a mechanism to support both offline transactions and recovery, while preventing such double spending attacks. Our proposal is to ensure that *only one dongle per user is active at a time*, by ensuring that a dongle is "fresh." For purposes of exposition, we define the freshness period to be one week. When a dongle connects to the service, the service will give the dongle a certified (signed by the service private key) timestamp. The service also takes care to ensure that it never sends the certified timestamp to anything but a trusted dongle. Moreover, we must ensure that only the service can load timestamps onto dongles. One way to enforce this is to establish long-lived OTR keys on dongles at fabrication time.

Each dongle is uniquely identifiable to the service, with a serial number (communicated over the long-lived OTR connection) and the aforementioned dongle-specific certificate. When recovery is initiated, the service will *stop* giving certified timestamps to the original dongle and will, henceforth, only give certified timestamps to the new dongle. However, since transactions sent directly between dongles are not encrypted by OTR, we must take care to prevent reuse of timestamps by participating parties. Our use of dongle-specific keys, described above, facilitates this. For additional defense, the service can include a user's Bitcoin public key in the timestamps it issues to that user's dongle, and the dongle can include this timestamp in the metadata of outgoing Bitcoin transactions. This leverages Bitcoin's inherent integrity protections to ensure that the party presenting a timestamp with a transaction really owns that timestamp.

To ensure that only one of a user's dongles is active at a time, recipients of offline transactions must verify that a sending dongle has *two* timestamps: a recent one that is less than a week old, and an older one (e.g., the first timestamp the dongle ever received) that is more than one week old. This process enforces a one-week *waiting period* before a recovered dongle can be used offline; the service can also prevent the use of recovered dongles online until one week after the recovery process has completed. (Note that completely new dongles can be certified as such by the service and can be used online and offline immediately, since there could be no previously pending transactions from that wallet.) Thus, as long as offline transactions are uploaded within a week, their recipients can be assured that the sender could not have created and used a new dongle to double spend those funds in the meantime.

We note that there is a tradeoff with the choice of waiting period: a longer waiting period allows for longer periods of safe disconnected operation for other users, but it also increases the time that the user who lost his or her dongle must wait before being able to use Braavos again.

6. IMPLEMENTATION

We ground our inquiry in a concrete prototype implementation. Our prototype aided in iteratively surfacing and reasoning concretely about conceptual challenges throughout our design process. We implemented a simple proof-of-concept dongle and service in python on Raspberry Pis and desktop Linux machines, and we structured it in a modular manner (Figure 5) similar to the TCP/IP protocol stack.

Application Layer. Our application layer implements the core Braavos protocols using a python-based OTR library for encryption [44], as well as a python Bitcoin library [43] that enables transaction manipulation and creation. In particular, we implement the enroll, send funds, receive funds, and create contact operations as described in Section 4.1. At the service end, we utilize the popular blockchain.info API to interface with the Bitcoin network. Our application layer implementation comprises 1131 lines of code.

Transport Layer. To provide guaranteed data delivery over audio, we implemented a simple reliable transport layer similar to a stripped-down version of TCP. Our transport layer supports variable-length packets (up to 70 bits including a 7-bit packet delimiter) and exposes blocking, socket-like *sendAll* and *receiveData* APIs responsible for sending and receiving an arbitrary number of bytes, respectively.

Physical Layer. We built a simple modem (759 lines of code) for Braavos's physical layer. Many techniques exist for encoding data in audio signals; however, cellular voice channels present unique constraints. The cellular codecs are optimized for human voice-like signals and employ techniques such as Voice Activity Detection (VAD) and Automatic Gain Control (AGC) that distort signals. Prior efforts demonstrated the ability to send data over cellular voice channels [3, 16, 19]; however, due to the proprietary nature of industry efforts, we leverage ideas from the research community. In particular, Dhananjay et al. propose a protocol, *Hermes* [19], for modulating data over cellular voice channels. Techniques employed by Hermes include periodic amplitude variation to mimic the pulsing of a voice and struc-

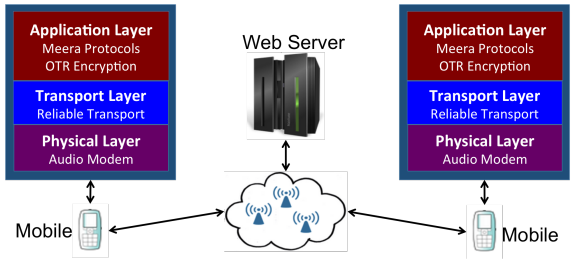

Figure 5: Our modular Braavos prototype implementation, similar to the TCP/IP protocol stack.

turing a signal to achieve a fixed fundamental frequency.

We leverage the above-mentioned ideas in our implementation; however, since our goal is to demonstrate the feasibility of the data-over-audio channel in Braavos, we do not focus on optimizing the underlying modulation scheme.

End-to-End Transaction Test. Upon implementing Braavos, we successfully used it — including the audio channel — to send and receive bitcoins between Braavos and a Coinbase [18] account.

7. EVALUATION

Having described how Braavos can support and extend traditional branchless banking operations with stronger security guarantees, we next analyze it conceptually based on our goals and evaluate its feasibility based on our prototype.

7.1 Security Analysis

Figure 1 overviews the security and functionality goals met by Braavos and summarizes the primary techniques used to achieve each. We now examine how Braavos behaves when faced with a variety of threats.

Threats from the Cellular Network. According to our threat model (Section 3.3), we assume that the cellular network may attempt eavesdropping, man-in-the-middle, or other network-level attacks on communications between the dongle and service. Braavos protects the confidentiality and integrity of these communications using OTR for end-to-end encryption. Denial-of-service attacks are discussed below.

Threats from the User's Phone. To mitigate threats due to the user's untrusted phone, Braavos leverages a secure hardware dongle and tunnels encrypted audio data through the phone. This design prevents the phone from violating the confidentiality or integrity of messages between the dongle and the service, e.g., by modifying a user's intended transactions. By leveraging Bitcoin, we also inherit the unforgeability of Bitcoin transactions without knowledge of the corresponding private key, which is held only by the dongle.

Threats from the Service. To prevent the service from forging or modifying a user's transactions, the Bitcoin private key required to create valid transactions for a user is stored only on that user's trusted dongle. Importantly, the service significantly differs from a centralized or "hosted wallet" model (e.g., as in [2, 9, 32]) in that it does not have direct control over a user's money. In the common case, the user trusts the service to provide correct information about incoming transactions and about the mapping between recipient identifiers and Bitcoin addresses — by providing false information here, in the worst case the service can trick the user into sending an intended transaction amount to an unintended recipient. However, as discussed in Section 3.3, these attacks are limited in scope, can be detected, and represent a tradeoff with usability (a more concerned user can choose to directly enter a recipient's Bitcoin address).

Threats from Malicious Users. We assume the dongle is trustworthy and tamper-resistant, preventing the user from extracting the private key. This prevents users from conducting double spending attacks. In Section 5, we further described how Braavos can support both offline and recovery protocols while still preventing double spending attacks. Additionally, we adopt Bitcoin's resilience to malicious Bitcoin users even external to Braavos: it is difficult for those users to mount online double-spending attacks against Braavos users (e.g., when sending remittances into Braavos) as long as the majority of Bitcoin miners are honest.

Threats from Malicious Agents. Braavos places limited trust in agents. For cash-in and cash-out transactions, users should rely on authenticated messages from the service for transaction confirmations, rather than relying on the word of human agents. This reliance on authenticated messages from the service means that a user may need to stay colocated with the agent for some period of time; many other branchless banking systems also have this property.

Threats from Fraudulent Dongles. To prevent agents or users from introducing fraudulent dongles into the ecosystem, each dongle comes preconfigured with a public-key certificate signed by the service, which can be validated even during offline transactions by other dongles. There are numerous places in the protocol in which the service or another dongle first verifies that a dongle is authentic before performing some action with that dongle.

Denial of Service Attacks. Similar to other systems, Braavos does not prevent denial-of-service attacks. For example, the service can refuse to upload a valid transaction to the blockchain, the cellular network can refuse to deliver messages, the user's phone can refuse to make calls, and an agent can refuse to conduct a cash-in or cash-out transaction. However, this class of attacks is detectable by users and cannot result in fraudulent transactions.

7.2 Prototype Evaluation

Our prototype implementation served primarily to inform and drive our conceptual design, and as an aid to identify potential design oversights. To demonstrate the feasibility of our model, we evaluate it next.

Experimental Setup. To evaluate our transport and physical layers, we conducted microbenchmark and end-to-end experiments with two Linux machines communicating between two AT&T phones connected via 3.5mm audio cables to the headphone and microphone ports of CMedia USB soundcards. We assume that a dedicated dongle would be optimized for audio processing capabilities, and hence we chose to emulate such capabilities with desktop machines.

We additionally profiled the Braavos application layer on both a desktop machine (Dell Optiplex 9020: Intel Core i7-4790 quad core 3.6GHz processor, 16GB RAM) and a Raspberry Pi (Model B+: ARM1176JZF-S 700MHz processor, 512MB RAM). Our goal was to study how existing low-end, unoptimized hardware (the Raspberry Pi) performs on application layer operations, like Bitcoin transaction creation and OTR encryption, when compared to a high-end desktop.

Full System Evaluation. To validate our full system im-

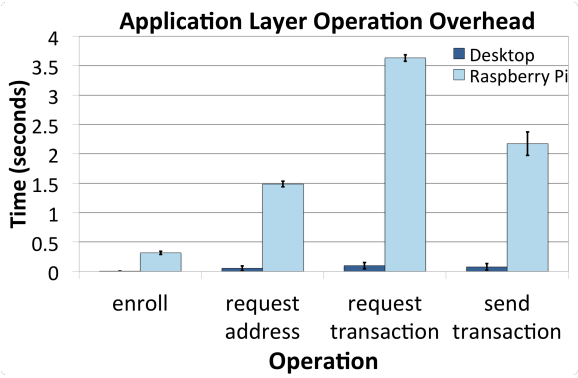

Figure 6: This graph compares the application layer runtime (i.e., excluding communication overhead) of our Braavos prototype running on a desktop and a Raspberry Pi. The latter is lower-end hardware more representative of a possible dongle; even unoptimized, the incurred overhead is acceptable.

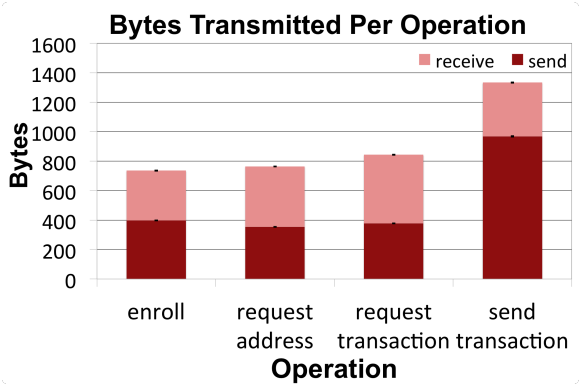

Figure 7: The size of each protocol message in our prototype. We discuss the time needed to send these messages in the text.

plementation, we profiled the operations supported by Braavos. Figure 6 shows the average overhead incurred by our application layer (i.e., not including communication but including Bitcoin and OTR operations) over five trials on both a desktop machine and a Raspberry Pi. While the Raspberry Pi is clearly slower, the application layer only incurs an overhead on the order of seconds on the unoptimized, low-end hardware. We believe these overheads are appropriate for our application domain. These results also suggest that an optimized hardware implementation would fare well.

Next, we examine how much data is transmitted for each Braavos operation. Figure 7 shows the average number of bytes (after OTR encryption) transmitted for each operation over five trials. To induce variations between trials, we created syntactically valid, but randomly generated, requests (e.g., randomly generated Bitcoin transactions with 1 input and 2 outputs). As previously noted, we separately verified that Braavos works in practice by sending a real Bitcoin transaction over the audio channel to a Coinbase account.

To send a transaction and receive confirmation (the largest operation at 1.3 KB total data), our unoptimized implementation took an average of 83.9 minutes (standard deviation 22.4 minutes) across five trials. Returning to our earlier citations of other in-band modems over cellular audio channels, we note that there do exist commercial in-band modems with significantly higher throughput than our unoptimized implementation. For example, other researchers reimplemented the proprietary Airbiquity modem and found a raw data rate of 400 bps over cellular audio, with an effective throughput of 21 bytes per second [16]. At this rate, the 1.3 KB operation to send a transaction and receive confirmation would take about one minute. Furthermore, the authors of [19] proposed a protocol for achieving 1.2 kpbs throughput with low error rates. We emphasize that our focus was on demonstrating the *feasibility* of Braavos's model and its constituent components with a proof-of-concept, not on optimizing the underlying communication channel.

Microbenchmarks. We use a set of microbenchmarks to profile the physical and transport layers of Braavos. All measurements use fixed-size 70 bit packets with 27 bit payloads.

First, we measured the round trip time (RTT) for packets to understand how efficient our physical and transport layers are. We measured an average RTT of 4.91 seconds (standard deviation of 0.02 seconds) over 100 packets. Additionally, we quantified the reliability of the physical layer by observing the packet loss rate, which we define according to Equation 1. For a sequence of 1000 packets, we observed a packet loss rate of 0.375.

$$\text{Packet Loss Rate} = 1 - \left(\frac{\#\text{of Received Packets}}{\#\text{of Sent Packets}}\right) \quad (1)$$

To understand the overhead incurred by our transport layer due to packetization and transcoding, we measured the time to send a fixed number of packets (i.e., one call to our *sendAll* API). The average time over five trials to send 100 packets, including retransmissions and receipt of all ACKs, was 14.9 minutes, with a standard deviation of 1.6 minutes.

These results are relatively consistent with our full system measurements, though we observed significant variability in the time it took to conduct the Braavos protocols, likely due to varying packet loss rates resulting from variations in the cellular channel over time.

8. DISCUSSION

Having presented Braavos, which enables secure mobile money operations in the face of unique ICTD constraints, we reflect on usability and deployment considerations, as well as limitations of our preliminary prototype.

Usability and Form Factor. We have focused on the design and prototype implementation of Braavos rather than directly on its usability; however, usability is clearly important for its adoption. At a high level, we have designed Braavos to mirror the functionality and protocols of existing mobile money systems like M-Pesa, matching their usability [34]. We have also aimed to hide the underlying complexities of Bitcoin from the user as much as possible while supporting a gradient of trust — for example, giving users the option to request ID to Bitcoin address mappings from the service, or allowing for passphrases with varying degrees of complexity.

Our prototype is not implemented on custom hardware and uses a command-line interface for demonstration purposes. In a full system, we expect the dongle to include standard hardware I/O capabilities, including an LCD screen, a phone-style keypad (to enter PINs, transaction amounts, and recipient identifiers), and a button per Braavos protocol or a set of arrow keys to select the protocol from a list displayed on the screen. We envision a form factor similar to the second factor devices distributed by some banks [28]. We also note that not all phone audio jacks are equivalent,

but that the dongle can come with a set of adapters.

Deployment and Economic Considerations. There are different possible models for dongle distribution and funding. For example, like M-Pesa, the service can charge transaction fees for certain transactions and use these fees in part to fund the dongles. (Taking small fees would also allow the service to provide some level of fraud protection, e.g., reimburse users in the case of fraudulent transactions resulting from stolen PINs or physical manipulation of the dongle's I/O pathway.) A nonprofit organization aimed at improving access to financial services in developing regions (e.g., [8]) could also help subsidize dongles. Prior work [10] that involved deploying a mobile money system in Afghanistan considered seeding participants with phones and money.

Another possible cost for users is phone service fees for calls made to the service. In an optimized implementation of Braavos's audio channel, e.g., [19], we anticipate these calls to be short. With sufficient transaction fees, the service may also wish to provide a toll-free number to its users to incentivize adoption. We also observe that M-Pesa is owned by Safaricom, a mobile network provider, suggesting that both parties may benefit and collaborate in such a system.

A final deployment consideration is Bitcoin. While Braavos leverages many of Bitcoin's inherent security properties, Bitcoin does present some possible disadvantages, including the (current) volatility of the currency's value, the public nature of the blockchain, and lengthy transaction confirmation times. We hope that our work, including the manifestation of Bitcoin's advantages and disadvantages within it, will lay a foundation for further efforts aimed at integrating cryptocurrencies into mobile money systems, whether built on Bitcoin or an alternate cryptocurrency.

Prototype Limitations. Finally, we consider the limitations of our current prototype implementation. First, our prototype dongle does not use custom-built hardware; second, our audio communication protocol was not optimized for performance. However, our goal in this work is not to create a final product, but rather to explore the feasibility and implications of a cryptocurrency-based mobile money system along with additional mechanisms for bolstering security (such as a trusted hardware dongle and the use of the audio channel). Indeed, prior work suggests that sufficiently high data rates over audio are possible (e.g., [16, 19]) and previous work has considered auxiliary hardware in the developing world context [15]. Our prototype implementation allowed us to iteratively design and evaluate our system, and our experience suggests that Braavos is a promising approach for secure branchless banking in developing regions.

9. ADDITIONAL RELATED WORK

Secure Branchless Banking. Others have considered security and privacy issues in branchless banking in the developing world. For example, Panjwani identified security concerns and requirements in this context [39], which we extend in our work. Existing branchless banking systems do not meet these requirements, and academic work has not closed this gap. Additionally, Panjwani proposed a protocol for improving transaction receipt authentication [40], but we are not aware of much other work in this space.

The authors of [17] consider a SIM-based approach to improving mobile money security that adds web server functionality to the SIM environment, and TagPay [49] uses near sound data transfer to authenticate a user's phone to a physically co-located, networked point-of-sale device. Compared to these prior efforts, Braavos explicitly strives to meet a broader set of security goals (e.g., reducing trust in the service). Moreover, while [17] and TagPay tightly couple mobile money functionality to phone ownership, Braavos supports heterogeneous phone usage models, and unlike [15], it supports diverse communication channels ranging from voice to IP. Braavos additionally does not require parties to be physically co-located like TagPay, making it more suitable for long-distance remittances. Braavos also explores emergent functionalities like offline transactions that arise from integrating new techniques into mobile money systems.

Recent work has identified systemic vulnerabilities in many Android mobile money applications used in the developing world [45]. In [26], the authors raise related concerns from a legal/policy perspective, and highlight the importance of security to the widespread adoption of mobile money in developing regions. These findings, combined with real attacks [37] and easily exploitable vulnerabilities [42] in systems like M-Pesa motivate our current work.

Others have studied security in the ICTD context more generally (e.g., [6]) or studied the use of mobile money in the developing world without a security focus (e.g., [10]). We build on both directions in this work.

Extending Mobile Phones. Prior efforts have proposed extending mobile device capabilities with additional hardware. For example, FoneAstra [15] extends a phone with additional sensing capabilities through a serial connection, and AudioDaq [51] and others [33] use the phone's audio jack to power and communicate with external peripherals. Though we did not focus on the custom hardware implementation for our proposed dongle, these prior works suggest that such a hardware implementation is feasible. Unlike these prior works, our dongle aims to provide security properties while minimizing trust in the phone itself. As discussed previously, Plug-n-Trust [48] also uses trusted external hardware to secure medical applications on an untrusted mobile phone.

Bitcoin. The computer security community has studied various aspects of Bitcoin in recent years, including security and anonymity analyses of the existing protocol and ecosystem [22, 23, 35] along with the development of alternate versions of Bitcoin (e.g., [5, 7]). A summary of many of these issues appears in [11]. We leverage Bitcoin and some of its security properties in our work, though our overall design is more general and could be combined with other Bitcoin variants or digital currency schemes. Interest in Bitcoin is increasing in developing regions as well (e.g., [13, 25]). In [27], Hileman presents a detailed index for reasoning about the potential utility and plausibility of adoption spanning over 150 countries, incorporating factors such as the prevalence of international remittances and financial repression.

Related to our work, there exist dedicated Bitcoin hardware wallets like Trezor [50]. Like Braavos, Trezor stores the user's Bitcoin private key on the device and supports recovery. Unlike Braavos, however, Trezor requires an Internet-connected computer rather than a phone, making it unsuitable for the developing world context we target in this work.

10. CONCLUSION

Mobile money systems have created unprecedented opportunities for people in the developing world to access financial

services. Unfortunately, despite their growing popularity, existing systems exhibit systemic security vulnerabilities. In this work, we explore novel methods of designing secure mobile money solutions, and we delineate a set of security and functionality goals to guide the design of such systems in the face of unique ICTD challenges. To meet these goals, we introduce Braavos, a secure mobile money framework that integrates non-traditional techniques — secure trusted hardware, cryptocurrencies (specifically Bitcoin), and data-over-audio — and in doing so, lays a foundation for future efforts to explore new directions for branchless banking security. Among other benefits, Braavos removes trust from the user's potentially compromised phone, ensures the authenticity and integrity of transactions, and enables offline transactions between co-located users without cellular connectivity. An end-to-end evaluation of our prototype implementation and a set of microbenchmarks demonstrate the feasibility of the Braavos model for improving the security of mobile money for individuals in the developing world.

Acknowledgements

We thank Gaetano Borriello, Waylon Brunette, and Joshua Blumenstock for their insights regarding branchless banking in ICTD contexts. We also thank Eric Whitmire and Elliot Saba for many discussions on signal processing, and we thank Jeff Haley for lending his Bitcoin expertise.

References

[1] Square. `https://squareup.com/`.

[2] 37coins. `https://www.37coins.com/en/`.

[3] Airbiquity. `http://www.airbiquity.com/`.

[4] J. C. Aker and I. M. Mbiti. Mobile phones and economic development in Africa. *Center for Global Development Working Paper*, (211), 2010.

[5] U. Andrew Miller, U. Elaine Shi, A. Juels, B. Parno, and J. K. UMD. Permacoin: Repurposing Bitcoin Work for Data Preservation. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2014.

[6] Y. Ben-David, S. Hasan, J. Pal, M. Vallentin, S. Panjwani, P. Gutheim, J. Chen, and E. A. Brewer. Computing security in the developing world: A case for multidisciplinary research. In *5th ACM Workshop on Networked Systems for Developing Regions*, pages 39–44. ACM, 2011.

[7] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Practical Decentralized Anonymous E-Cash from Bitcoin. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2014.

[8] Bill & Melinda Gates Foundation. Financial services for the poor. `http://www.gatesfoundation.org/What-We-Do/Global-Development/Financial-Services-for-the-Poor`.

[9] Bitpesa. `https://www.bitpesa.co/`.

[10] J. E. Blumenstock, M. Callen, T. Ghani, and L. Koepke. Promises and Pitfalls of Mobile Money in Afghanistan: Evidence from a Randomized Control Trial. In *International Conference on ICTD*, 2015.

[11] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *IEEE Security and Privacy (Oakland)*. IEEE, 2015.

[12] N. Borisov, I. Goldberg, and E. Brewer. Off-the-record communication, or, why not to use PGP. In *ACM workshop on Privacy in the Electronic Society*, 2004.

[13] D. Cawrey. Banks, corruption and crypto: Can bitcoin change india? `http://www.coindesk.com/banks-corruption-crypto-can-bitcoin-change-india/`, 2014.

[14] Mobile phone acess reaches three quarters of planet's population. `http://www.worldbank.org/en/news/press-release/2012/07/17/mobile-phone-access-reaches-three-quarters-planets-population`.

[15] R. Chaudhri, G. Borriello, and W. Thies. Foneastra: making mobile phones smarter. In *4th ACM Workshop on Networked Systems for Developing Regions*, page 3. ACM, 2010.

[16] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*, 2011.

[17] S. Cobourne, K. Mayes, and K. Markantonakis. Using the smart card web server in secure branchless banking. In *Network and System Security*, pages 250–263. Springer, 2013.

[18] `https://www.coinbase.com`.

[19] A. Dhananjay, A. Sharma, M. Paik, J. Chen, T. K. Kuppusamy, J. Li, and L. Subramanian. Hermes: Data transmission over unknown voice channels. In *10th ACM International Conf. on Mobile Computing and Networking*, 2010.

[20] N. Eagle. Don't let developing countries lag behind in the smartphone revolution. `http://www.theguardian.com/global-development-professionals-network/2014/dec/18/developing-countries-smartphone-revolution-internet-access`, Dec. 2014.

[21] Eko. `http://eko.co.in/`.

[22] I. Eyal. The Miner's Dilemma. In *36th IEEE Symposium on Security and Privacy, S&P 2015*, 2015.

[23] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography & Data Security*, 2014.

[24] `https://fidoalliance.org/`.

[25] G. Gupta. Venezuelans turn to bitcoins to bypass socialist currency controls. `http://www.reuters.com/article/2014/10/08/us-venezuela-bitcoin-idUSKCN0HX11O20141008`, October 2014.

[26] A. Harris, S. Goodman, and P. Traynor. Privacy and Security Concerns Associated with Mobile Money Applications in Africa. *Washington Journal of Law, Tech. & Arts*, 2013.

[27] G. Hileman. The Bitcoin Market Potential Index. In *Financial Cryptography*, 2015.

[28] `www.hsbc.co.uk/1/2/contact-and-support/security-centre/secure-key`.

[29] `https://internet.org/`.

[30] W. Jack and T. Suri. Mobile money: the economics of M-PESA. 2011.

[31] G. O. Karame, E. Androulaki, and S. Capkun. Double-spending fast payments in bitcoin. In *ACM Conference on Computer and Communications Security*, 2012.

[32] Kipochi. https://www.kipochi.com.

[33] Y.-S. Kuo, S. Verma, T. Schmid, and P. Dutta. Hijacking power and bandwidth from the mobile phone's audio interface. In *ACM Symp. on Computing for Development*, 2010.

[34] I. Mas and O. Morawczynski. Designing mobile money services: Lessons from M-PESA. *Innovations*, 4(2):77–91, 2009.

[35] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. In *Internet Measurement Conference (IMC)*, 2013.

[36] M-pesa. http://www.safaricom.co.ke/personal/m-pesa.

[37] Security Breach at M-PESA: Telco 2.0 Crash Investigation. http://www.telco2.net/blog/2010/02/security_breach_at_mpesa_telco.html.

[38] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[39] S. Panjwani. Towards end-to-end security in branchless banking. In *12th ACM Workshop on Mobile Computing Systems and Applications*, 2011.

[40] S. Panjwani. Practical receipt authentication for branchless banking. In *3rd ACM Symposium on Computing for Development*, 2013.

[41] Pew Research Center. Emerging Nations Embrace Internet, Mobile Technology. http://www.pewglobal.org/2014/02/13/emerging-nations-embrace-internet-mobile-technology, February 2014.

[42] Phone hacking latest trend in mobile money frauds. http://www.humanipo.com/news/3562/phone-hacking-latest-trend-in-mobile-money-frauds/.

[43] https://github.com/petertodd/python-bitcoinlib.

[44] https://github.com/python-otr/pure-python-otr.

[45] B. Reaves, N. Scaife, A. Bates, P. Traynor, and K. Butler. Mo(bile) Money, Mo(bile) Problems: Analysis of Branchless Banking Applications in the Developing World. In *Proceedings of 24th USENIX Security Symposium*, 2015.

[46] D. Richardson, R. Ramirez, and M. Haq. Grameen Telecom's village phone programme in rural Bangladesh: A multi-media case study final report. *CIDA*, 2000.

[47] http://www.smartcardfocus.com/shop/ilp/id~521/smartcafe-expert-3-2-72k/p/index.shtml.

[48] J. M. Sorber, M. Shin, R. Peterson, and D. Kotz. Plug-n-trust: practical trusted sensing for mhealth. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 309–322. ACM, 2012.

[49] http://en.tagpay.fr/tagpay-platform#nsdt.

[50] https://www.bitcointrezor.com/.

[51] S. Verma, A. Robinson, and P. Dutta. Audiodaq: turning the mobile phone's ubiquitous headset port into a universal data acquisition interface. In *10th ACM Conference on Embedded Network Sensor Systems*, 2012.