Security Threats from Bitcoin Wallet Smartphone Applications: Vulnerabilities, Attacks, and Countermeasures

Yiwen Hu, Sihan Wang, Guan-Hua Tu
Li Xiao, Tian Xie, Xinyu Lei
Department of Computer Science and Engineering
Michigan State University
East Lansing, Michigan, USA
{huyiwen3,wangsih3,ghtu,lxiao,xietian1,leixinyu}@msu.edu

Chi-Yu Li
Department of Computer Science
National Chiao Tung University
Hsinchu City, Taiwan, ROC
chiyuli@cs.nctu.edu.tw

ABSTRACT

Nowadays, Bitcoin is the most popular cryptocurrency. With the proliferation of smartphones and the high-speed mobile Internet, more and more users have started accessing their Bitcoin wallets on their smartphones. Users can download and install a variety of Bitcoin wallet applications (e.g., `Coinbase`, `Luno`, `Bitcoin Wallet`) on their smartphones and access their Bitcoin wallets anytime and anywhere. However, it is still unknown whether these Bitcoin wallet smartphone applications are secure or if they are new attack surfaces for adversaries to attack these application users. In this work, we explored the insecurity of the 10 most popular Bitcoin wallet smartphone applications and discovered three security vulnerabilities. By exploiting them, adversaries can launch various attacks including Bitcoin deanonymization, reflection and amplification spamming, and wallet fraud attacks. To address the identified security vulnerabilities, we developed a phone-side Bitcoin Security Rectifier to secure Bitcoin wallet smartphone application users. The developed rectifier does not require any modifications to current wallet applications and is compliant with Bitcoin standards.

CCS CONCEPTS

• **Security and privacy → Software and application security**; **Mobile and wireless security**.

KEYWORDS

Bitcoin wallets; Blockchain; Security; Mobile networks

ACM Reference Format:
Yiwen Hu, Sihan Wang, Guan-Hua Tu, Li Xiao, Tian Xie, Xinyu Lei and Chi-Yu Li. 2021. Security Threats from Bitcoin Wallet Smartphone Applications: Vulnerabilities, Attacks, and Countermeasures. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy (CODASPY '21), April 26–28, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3422337.3447832

1 INTRODUCTION

Bitcoin is the most popular cryptocurrency throughout the world. Many consumers and businesses (e.g., Microsoft, Newegg, Overstock, Subway, Starbucks) have accepted Bitcoin as a means of payment [1]. With the rapid deployment of the high-speed mobile Internet and the proliferation of smartphones, more users have started accessing their Bitcoin wallets by various Bitcoin wallet smartphone applications. However, previous Bitcoin security works mainly targeted the exploration of the security of Bitcoin protocol, infrastructure, and PC-side Bitcoin wallet clients [26, 33, 35]. This thus raises a natural question: *will these Bitcoin wallet smartphone applications create new attack vectors for adversaries to launch new attacks against mobile users?* Unfortunately, our study yields a positive answer. In this work, we studied the 10 most popular Bitcoin wallet smartphone applications (ranked by download counts) and discovered three security vulnerabilities spanning the implementation of Bitcoin wallet applications and the operation of Bitcoin wallet application services. In regards to the implementation issues, a Bitcoin application library which has been broadly used by various Bitcoin wallet smartphone applications, *BitcoinJ*, leaks Bitcoin wallet user privacy (e.g., all of the users' Bitcoin addresses) (*Vulnerability V1*) and continuously downloads unwanted Bitcoin transactions in the background without providing users any notifications or alerts (*Vulnerability V2*). In terms of operation issues, we found that the principle of the Bitcoin wallet service decentralization can be violated by Bitcoin wallet smartphone applications and the users of these applications are more vulnerable to financial fraud attacks. More threatening is that most users are unaware of this (*Vulnerability V3*).

By exploiting these vulnerabilities, we devise three proof-of-concept attacks against users: (1) Bitcoin Deanonymization attack; (2) Bitcoin Reflection and amplification spamming attack; (3) Mobile Bitcoin wallet fraud attack. The first attack allows adversaries to deanonymize non-RealName Bitcoin addresses used by Bitcoin wallets. This attack not only identifies all Bitcoin addresses of Bitcoin wallet application users with a low error rate (e.g., 10^{-13}%) but also is capable of discovering the real identities of application users while some usage conditions are satisfied. In the second attack, adversaries exploit the intermediate public Bitcoin network elements (e.g., full node clients) to deliver continuously, unsolicited Bitcoin spamming traffic to Bitcoin wallet smartphone application users; the spamming traffic is comprised of all new Bitcoin transactions produced by all global Bitcoin users during the attack. This is a low-cost attack; the attackers only need to send a few small attack

Category	Attack	Victim	Victim Net. Type	Description and Threat	Main Vulnerability
Privacy leakage	Bitcoin deanonymization attack	Users	Wi-Fi	Adversary exploits the vulnerabilities of libraries used by wallet apps to produce the unique wallet fingerprint and associates it with user identity for further user tracking.	V1: Bitcoin Addresses of Wallets can be Leaked (§5.1).
Spamming	Bitcoin reflection & amplification spamming attack	Users and phones	Cellular	Adversary exploits the vulnerabilities from the implementation of applications and leverages the public Bitcoin network elements to produce continuous unsolicited Bitcoin spamming traffic towards victims; this attack causes an increase to the victim's mobile data service bill and the reduction to the victim's smartphone battery life.	V2: No Anti-spam Defense of Downloading Bitcoin Transactions (§5.2).
Financial loss	Mobile Bitcoin wallet fraud attack	Users	Wi-Fi/Cellular	Adversary exploits users' misunderstanding of wallet applications to develop and promote their applications to victims and take advantage of the victims; victims thus suffer from financial loss.	V3: The Decentralization of Bitcoin Wallet Service can be Violated (§5.3).

Table 1: Summary of our main findings on Bitcoin wallet smartphone application vulnerabilities and proof-of-concept attacks.

initiation packets to Bitcoin networks. The amplification factor (the ratio of the size of spamming traffic to the size of attack initiation packets) is nearly 3,666 observed in our experiments. This attack damages victims in two ways: (1) the victims need to pay for the spamming Bitcoin messages while using cellular network services; (2) the victims' phones consume 96% power more than the phones that are not under attack. The third attack shows why adversaries can launch various Bitcoin fraud attacks against Bitcoin wallet application users beyond the limitations of other types of mobile financial attacks. Our findings are summarized in Table 1. We further develop a phone-side Bitcoin Security Rectifier to address the identified vulnerabilities.

In summary, this paper makes three contributions. First, we conducted the first study to explore the security vulnerabilities caused by the inconsistencies between Bitcoin wallet service designs (standards) and the implementations/operations of Bitcoin wallet smartphone applications. Three new security vulnerabilities were discovered and reported to the CVE database (https://cve.mitre.org). Second, we devised three proof-of-concept attacks by exploiting the identified vulnerabilities and assessed their real-world impact. Third, we pointed out the root causes of these vulnerabilities and developed practical solutions. The lessons we learned not only help secure the increasing usage of Bitcoin wallet service but also provide insights for other cryptocurrency platform users (e.g., Ethereum).

The rest of this paper is organized as follows: Sections 2 and 3 present related work and the background of Bitcoin, respectively. Section 4 describes the threat model and the methodology of our paper. We present three discovered security vulnerabilities and sketch three proof-of-concept attacks in Sections 5 and 6, respectively. We propose solutions in Section 7 and conclude this paper in Section 8.

2 RELATED WORK

Studying Bitcoin wallet vulnerabilities mainly falls into two categories: user privacy and wallet security. In the category of user privacy, Gervais *et al.* [34] inferred the privacy of lightweight Bitcoin wallet users using the intercepted Bloom filter. In the best case, the probability that adversaries can correctly discover users' all addresses of that Bitcoin wallet is around 80%. [26] performed network traffic analysis using machine learning techniques to identify the user activities (e.g., sending/receiving Bitcoin) on their smartphone Bitcoin wallet. A lightweight framework was developed by Conti *et al.* [30] to collect and identify Bitcoin addresses managed by the ransomware campaigns. In the category of Bitcoin wallet security, Brengel *et al.* [29] exploited the wrong usage of cryptographic primitives and scanned Bitcoin blockchain for ECDSA nonce reuse to cause Bitcoin wallet private key leakage. Gervais *et al.* [35] claimed that some of Bitcoin wallet services are actually controlled by a small set of entities, which may be targeted and further attacked. Turuani *et al.* [39] applied a formal method to studying the two-factor authentication protocol of `Electrum Bitcoin Wallet` (a Bitcoin wallet application).

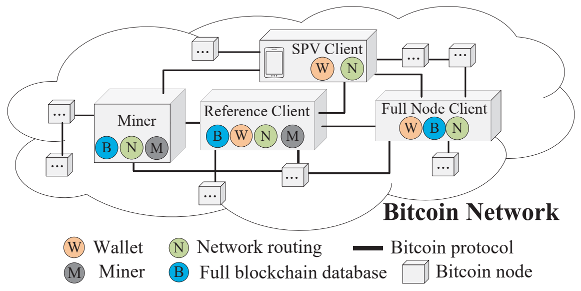


Figure 1: Bitcoin network architecture.

Different from these works, our security study target the implementation and operations of Bitcoin wallet smartphone applications, which have not been fully explored by academia yet.

3 BACKGROUND

Bitcoin Network architecture. Figure 1 illustrates the Bitcoin network architecture. A network node running the Bitcoin protocol may contain four functions: (1) wallet, (2) miner, (3) full blockchain database, and (4) network routing. The wallet function provides the Bitcoin wallet service for users. This function controls access to a user's money (i.e., BTC, the currency of Bitcoin), including managing private/public keys and addresses, tracking the balance, and generating and signing transactions. The miner function validates new transactions and attaches them to the Bitcoin blockchain. The full blockchain database function maintains a complete and up-to-date copy of all Bitcoin blocks. The network routing function allows a node to communicate with other nodes [27].

Based on the functions supported, there are four types of nodes in Bitcoin networks, namely *reference client*, *miner*, *full node client (FNC)*, and *simple payment verification (SPV) client*[1]. These nodes communicate with each other by using Bitcoin protocol over TCP protocol without encryption protection[3, 31]. The *'Reference Client'* implements all four functions. The *'Miner'* supports three of four functions, namely miner, full blockchain database, and network routing. It validates/verifies all new transactions broadcasted in Bitcoin networks and attaches them to the Bitcoin blockchain. The

[1]In this paper, we will use SPV clients and Bitcoin wallet applications interchangeably.

'Full Node Client' supports three functions: wallet, full blockchain database, and network routing. The *'SPV Client'* only supports two functions: wallet and network routing. It is usually deployed on some resource-constrained platforms (e.g., smartphones) which do not have sufficient resources.

How Bitcoin Payment Works. A Bitcoin wallet user first creates a single master key, aka seed phrase, which can be used to derive a collection of key pairs; each key pair comprises a private key and a public key. Each public key in a key pair can generate a Bitcoin address; a Bitcoin address is similar to a traditional bank account number.

To initiate a Bitcoin payment transaction towards single or multiple Bitcoin payees, the payer has to generate a `tx`, the transaction message, and broadcast it to the Bitcoin network. The miners will validate the transaction. If the transaction is valid, the miners will append it to the Bitcoin blockchain.

4 THREAT MODEL AND METHODOLOGY

Threat Model. In this study, adversaries are people or organizations which launch remote attacks against Bitcoin smartphone wallet users. We consider the adversaries with the following capabilities: (1) they can intercept, modify, or inject any messages in the public communication channels; (2) they adhere to all cryptographic assumptions, e.g., adversaries cannot decrypt an encrypted message without the decryption key; and (3) they have no control over the victim's smartphones and Bitcoin network infrastructure.

Responsive Study Methodology. In this work, we studied 10 popular Bitcoin wallet smartphone applications, namely Coinbase, Bitcoin Wallet, Blockchain Wallet, Bitcoin.com Wallet, Luno, Mycelium, Coinomi, BRD, BitPay, and Simple Bitcoin. We bear in mind that some feasibility tests and attack evaluations might be harmful to the victims. We thus conducted this study in a responsible manner through two measures. First, we used only our phones as victims. Second, we did not distribute any malicious Bitcoin applications or libraries to the public. We seek to disclose new security vulnerabilities of Bitcoin wallet smartphone applications and effective attacks, but not to aggravate the damage.

5 SECURITY VULNERABILITIES

In this section, we present three security vulnerabilities of Bitcoin wallet smartphone applications and their service providers.

5.1 V1: Bitcoin Addresses of Wallets can be Leaked

The first security vulnerability is that the Bitcoin addresses of wallet smartphone applications can be leaked to adversaries. In particular, *BitcoinJ* [7], a Bitcoin client library which has been broadly used by a great number of Bitcoin wallet smartphone applications (e.g., `Bitcoin Wallet`, `Mycelium`) to communicate with Bitcoin networks, leaks the Bitcoin wallet user privacy.

According to Bitcoin standards, after an SPV client (i.e., a Bitcoin wallet application) connects with a Bitcoin FNC (Full Node Client), a `filterload` message [3] will be sent to the FNC. This message is used to specify the Bitcoin wallet users' interest in particular transactions. In practice, SPV clients are usually configured to be interested in the transactions in which the Bitcoin wallet users' Bitcoin addresses are involved. The message carries three key parameters, namely *filter*, *nHashFuncs*, and *nTweak*. The *filter* itself is a bit-field Bloom filter so that the FNC can know the address interest of its connected Bitcoin wallet application. The Bloom filter is set based on feeding the data element (e.g., users' Bitcoin addresses) to a set of different *nHashFuncs* MurmurHash3 [2] hash functions. These *nHashFuncs* MurmurHash3 hash functions are initialized via $nHashNum \times 0xFBA4C795 + nTweak$, where *nHashNum* is the index of the hash function (e.g., 1st, 2nd, 3rd) and *nTweak* is a random number selected by SPV clients for a *filter*. To add a data item (e.g., a Bitcoin address) to the *filter*, the data item must be hashed by *nHashFuncs* different hash functions and set the corresponding bits in *filter* by the bitwise OR operation. For example, data items will be added to a clean 6-bit filter (value is 000000) using 2 different hash functions, and we assume that the results of the first and second hash functions are 7 (000111) and 9 (001001), receptively, the filter is set to 001111 (000111 OR 001001).

Our study shows that to prevent adversaries from accurately discovering all Bitcoin addresses ever used by Bitcoin wallet users by intercepting and analyzing the plain-text `filterload` message, two security mechanisms are implemented in *BitcoinJ*. First, some false addresses which do not belong to the user will be filtered out by the *filter* due to the false positive rate of Bloom filter. Second, the `filterload` message is only created and transmitted to the Bitcoin FNCs once (at when the TCP connections with FNCs are successfully established), which means that if adversaries cannot monitor all activities of SPV clients from the very beginning, they cannot intercept the `filterload` messages.

BitcoinJ Code 1 Handling the TCP disconnection with Full Nodes.

```
//Ref: core/src/main/java/org/bitcoinj/core/Peer.java
function HANDLEPEERDEATH(Peer peer)
    ...
    // Compare the current connections with max connections.
    if numPeers < getMaxConnections() then
        // If the wallet loses connections, try opening more until reach the max count.
        // Function triggerConnection() will invoke trigeerConnectionJob()
        triggerConnections();
        ...
    end if
end function
function TRIGGERCONNECTIONJOB(())
    ...
    if ConnectedAndPendingPeers < getMaxConnections() then
        connectTo(); // Function connectTo() will invoke handleNewPeer()
    end if
end function
// When having new connected peers, the wallet will handle these new peers
function HANDLENEWPEER((Peer peer))
    ...
    // Sending a filter that is used to set wallet's interests
    peer.setBloomFilter();
    ...
end function
```

With further analysis, we found that these two security mechanisms are not bullet-proof due to the following reasons. First, the false positive rate (the number of false addresses over the number of all addresses added to filter) of the Bloom filter used in *BitcoinJ* is set to 0.001%. Second, the `filterload` message which includes the calculated Bloom filter is transmitted to an FNC after the TCP connection between the SPV client and the FNC is established. However, our study shows that *BitcoinJ* will automatically discover and connect with a new FNC if the SPV client's existing TCP connection with an FNC is torn down (please see the function HandlePeerDeath

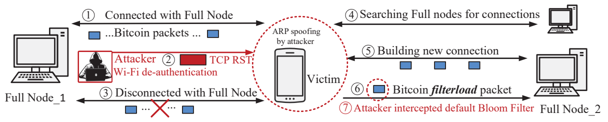
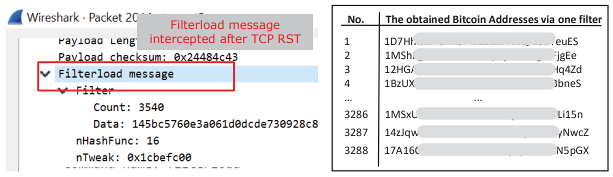

① Connected with Full Node
...Bitcoin packets ...
Attacker ② TCP RST
Wi-Fi de-authentication
③ Disconnected with Full Node
Full Node_1
ARP spoofing by attacker
Victim
④ Searching Full nodes for connections
⑤ Building new connection
⑥ Bitcoin filterload packet
⑦ Attacker intercepted default Bloom Filter
Full Node_2

Figure 2: The initiation of the retransmission of *filterload* messages via TCP reset and Wi-Fi de-authentication attacks.

in the above BitcoinJ Code 1). Therefore, the SPV client may suffer from TCP reset attacks (i.e., intentionally tearing down the victims' TCP connections with old FNCs) and retransmit `filterload` messages to new FNCs. Last but not least, we found that after the network interface used by the SPV client has become unavailable longer than 5 seconds, *BitcoinJ* will use a brand-new random number (i.e., *nTweak*) to generate the `filterload` messages. The root cause is that *BitcoinJ* maintains a `Reset` timer (i.e., 5 seconds) to monitor if there still exists an available network interface. On receipt of Bitcoin messages, the timer will be updated. Once it expires, several variables stored in the memory will be cleaned and reinitialized, including *nTweak* for generating *filter*. Note that the two `filterload` messages generated by two different *nTweaks* can help adversaries to significantly reduce the false positive rate of inferring Bitcoin users' addresses since *nTweak* is used to initialized *nHashFuncs* MurmurHash3 hash functions used in the `filterload` messages. For example, we assume that there are two `filterload` messages generated by two different *nTweaks* and the false positive rate is set to 0.001%. The probability that we incorrectly recognize a false address as a true address is $0.001\% \times 0.001\%$ since the address needs to pass two different *filters*, each only allows 0.001% false addresses to pass.

Validation. We conducted an experiment to validate this vulnerability as follows: First, we connected a tested smartphone (i.e., Samsung Galaxy S6 Edge) with the Internet via a home Wi-Fi network, downloaded and installed a tested Bitcoin wallet application (i.e., an SPV client, *Bitcoin Wallet* [32]) using *BitcoinJ*. Second, we separately deposited 0.0001 BTC into two Bitcoin addresses created by the SPV client. Third, to make the SPV client retransmit `filterload` messages, we launched two attacks against the SPV client: (1) TCP reset attacks using Netwox78 tool [15], which aims to send TCP reset packets on behalf of the SPV client to FNCs and (2) Wi-Fi de-authentication attack using Aircrack-ng [6], which aims to keep the SPV client's Wi-Fi down longer than 5 seconds, as shown in Figure 2. Fourth, we leveraged the *filter* carried in the intercepted `filterload` messages and public Bitcoin transaction database (i.e., *Blockchain.info* [8]) to infer the Bitcoin addresses used by the SPV client.

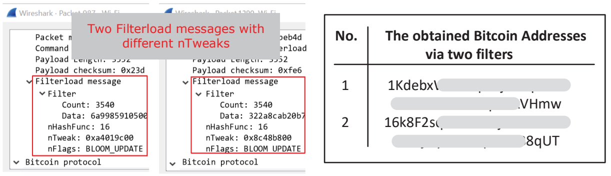
The experimental results are illustrated in Figure 3. By launching a TCP reset attack, adversaries can force the tested SPV client to retransmit a `filterload` message carrying a *filter* with a false positive rate of 0.001%, whereas the Wi-Fi de-authentication attack allows adversaries to obtain two `filterload` messages generated by two different *nTweaks*, which can reduce the false positive rate to 10^{-8}%. We further examined how many Bitcoin addresses recorded in the public Bitcoin transaction database can pass the intercepted *filters*. For the *filter* intercepted by TCP reset attack, there are 3288 Bitcoin addresses passing it. However, only two addresses were used by our Bitcoin wallet application, whereas the remaining



```
Payload Leng
Payload checksum: 0x24484c43
Filterload message
  Filter
    Count: 3540
    Data: 145bc5760e3a061d0dcde730928c8
  nHashFunc: 16
  nTweak: 0x1cbefc00
```

No.	The obtained Bitcoin Addresses via one filter
1	1D7Hh[redacted]euES
2	1MSh[redacted]jgEe
3	12HG[redacted]q4Zd
4	1BzUX[redacted]bneS
...	...
3286	1MSxU[redacted]Li15n
3287	14zJqw[redacted]yNwcZ
3288	17A16[redacted]N5pGX

(a) TCP reset attack


Wireshark · Packet 987 · Wi-Fi
Wireshark · Packet 1290 · Wi-Fi
Two Filterload messages with different nTweaks

```
Packet m
Command
Payload Length: 3552
Payload checksum: 0x23d
Filterload message
  Filter
    Count: 3540
    Data: 6a9985910500
  nHashFunc: 16
  nTweak: 0xa4019c00
  nFlags: BLOOM_UPDATE
Bitcoin protocol
```

```
eb4d
erload
Payload Length: 3552
Payload checksum: 0xfe6
Filterload message
  Filter
    Count: 3540
    Data: 322a8cab20b7
  nHashFunc: 16
  nTweak: 0x8c48b800
  nFlags: BLOOM_UPDATE
Bitcoin protocol
```

No.	The obtained Bitcoin Addresses via two filters
1	1Kdebx[redacted]VHmw
2	16k8F2s[redacted]8qUT

(b) Wi-Fi de-authentication attack

Figure 3: Inferring the Bitcoin addresses ever used by an SPV client.

3286 addresses were not. For two *filters* intercepted by Wi-Fi de-authentication attack, only two Bitcoin addresses passed both two *filters*; both Bitcoin addresses belong to our tested Bitcoin wallet application. Our validation experiment confirmed that *BitcoinJ* indeed leaks the Bitcoin addresses used by Bitcoin wallet applications.

5.2 V2: No Anti-spam Defense of Downloading Bitcoin Transactions

The second security vulnerability is that Bitcoin wallet smartphone applications using *BitcoinJ* will keep downloading the Bitcoin transactions out of the SPV clients' interest from the connected Bitcoin FNCs in the background without raising any alerts or notifications to SPV client users. Specifically, as we described previously, an SPV client will send a `filterload` message specifying its interest in specific Bitcoin addresses and transactions to Bitcoin FNCs. To prevent FNCs or adversaries (man in the middle) from accurately infer Bitcoin user/wallet privacy, the SPV client will add some false data items (i.e., the items are out of SPV client's interest) by configuring the *filter* fields of `filterload` messages (i.e., by setting the false positive rate to 0.001%). If an FNC finds any Bitcoin transactions matching the SPV client's interest, the FNC will prepare an inventory message (i.e., *inv*) carrying the identities of matched transactions and sends the message to the SPV client. SPV client then uses *BitcoinJ* to process the `inv` message.

Our study shows that *BitcoinJ* will conduct an examination on all transactions carried in the `inv` message before downloading them; however, the examination is flawed, as shown BitcoinJ Code 2. Specifically, two examinations are conducted: (1) whether a transaction has not been downloaded before (Line 11) and (2) whether the transaction is not self-generated (Line 14). If the transaction is not in either case, *BitcoinJ* will download the Bitcoin transaction (Line 19). This confirms that *BitcoinJ* does not examine if the Bitcoin transaction being downloaded can pass the prior *filter* sent to the FNC before downloading the transaction. If there are no additional security mechanisms (e.g., disconnecting with the malicious FNC) deployed at *BitcoinJ*, the SPV client will download Bitcoin transactions specified in the received `inv` message no matter whether these transactions are out of its interest or not. The SPV client users

BitcoinJ Code 2 Processing ***tx*** transactions in ***Inv*** Messages.

```
//Ref: core/src/main/java/org/bitcoinj/core/Peer.java
//Processing Inv message
function PROCESSINV(InventoryMessage inv)
    ...
    //Process inventory vectors carried in Inv
    Iterator<InventoryItem> it = transactions.iterator();
    while it.hasNext() do
        InventoryItem item = it.next();
        Tran conf = getConfidenceTable().seen(...);
        //If this transaction vector has been announced before
        if conf.numBroadcastPeers() > 1 then
            it.remove(); //don't download
            //If we created this transaction by ourselves
        else if Tran.getSrc().equals(Tran.Src.SELF) then
            it.remove(); //don't download
        else
            log.debug(": getdata on tx ",..., item.hash);
            ...
            pendingTxDownloads.add(conf);
        end if
    end while
    ...
end function
```

thus suffer from various attacks and are unaware of these attacks (a proof-of-concept attack is elaborated on in Section 6.2).

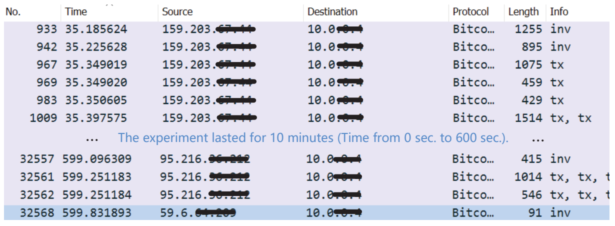
Validation. An experiment was conducted to validate this vulnerability. First, we installed a tested Bitcoin wallet application (i.e., *Bitcoin Wallet* [32]) using *BitcoinJ* on a Samsung Galaxy S6 Edge and connected it with the Internet via a home Wi-Fi network. Second, we intercepted the `filterload` message by using an ARP-spoofing attack [13][2]. Since Bitcoin does not employ message-level encryption and integrity protection on all message fields [31], we thus modified the message fields *Data* and *nFlag* of the `filterload` message to 0xFF...FF and 0, respectively, and sent the modified `filterload` message to the FNC with which the SPV client connects. The above modifications represent that the user is interested in all new Bitcoin transactions produced by all global Bitcoin users. The experiment lasted for 10 minutes.

Figure 4 illustrates that the SPV client received 130 inventory messages and downloaded 2535 Bitcoin transactions from the connected FNC in a 10-min experiment run. We have three observations. First, all downloaded Bitcoin transactions did not involve any Bitcoin addresses ever used by the tested SPV client. Second, the SPV client did not disconnect with the FNC which transmitted a great number of Bitcoin transactions out of the SPV client's interest. Third, the SPV client did not show any alerts or notifications (e.g., suffering from spamming attacks) to the user. These confirmed that both SPV client and *BitcoinJ* did not verify if the Bitcoin transactions are of interest before downloading them and not employ any additional security mechanisms against spamming Bitcoin transactions sent by connected FNCs.

5.3 V3: The Decentralization of Bitcoin Wallet Service can be Violated

By design [38], Bitcoin payment network uses the **P**eer-to-**P**eer (**P2P**) decentralized network architecture, which allows Bitcoin wallet clients to access Bitcoin networks without any intermediate proxies or servers. This approach not only protects the anonymity

[2]Note that this does not mean that adversaries can only intercept Bitcoin wallet users' Bitcoin messages while staying in the same home network of the users since all of their Bitcoin messages will be transmitted over the Internet



```
No.     Time          Source               Destination        Protocol  Length  Info
   933  35.185624     159.203.[redacted]   10.0[redacted]     Bitco…    1255    inv
   942  35.225628     159.203.[redacted]   10.0[redacted]     Bitco…     895    inv
   967  35.349019     159.203.[redacted]   10.0[redacted]     Bitco…    1075    tx
   969  35.349020     159.203.[redacted]   10.0[redacted]     Bitco…     459    tx
   983  35.350605     159.203.[redacted]   10.0[redacted]     Bitco…     429    tx
  1009  35.397575     159.203.[redacted]   10.0[redacted]     Bitco…    1514    tx, tx
        ...  The experiment lasted for 10 minutes (Time from 0 sec. to 600 sec.).  ...
 32557  599.096309    95.216.[redacted]    10.0[redacted]     Bitco…     415    inv
 32561  599.251183    95.216.[redacted]    10.0[redacted]     Bitco…    1014    tx, tx, t
 32562  599.251184    95.216.[redacted]    10.0[redacted]     Bitco…     546    tx, tx, t
 32568  599.831893    59.6.[redacted]      10.0[redacted]     Bitco…      91    inv
```

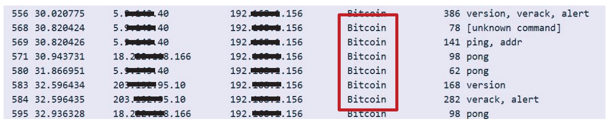
Figure 4: The received inventory messages and downloaded Bitcoin transactions from the connected FNC in a 10-min experiment run.

of Bitcoin wallet users (e.g., no server-side user registration is required and there are no checkpoints monitoring all Bitcoin user activities) but also prevents the Bitcoin wallet services from being compromised by a few nodes in Bitcoin networks. Adversaries need to control the majority of Bitcoin miners and full nodes; otherwise, the adversaries are unable to tamper Bitcoin payment transactions [27]. To keep Bitcoin decentralized, the Bitcoin official site, https://bitcoin.org/ [18] also releases the reference implementation of Bitcoin nodes, **Bitcoin Core** [19] supporting all Bitcoin functions (e.g., wallet, network routing, miner).

However, `Bitcoin.org` only developed and released the Bitcoin Core for PC users (Windows, Mac OS, and Linux) rather than smartphone users, which means that all Bitcoin wallet smartphone applications on the market are developed by are other parties. Even worse is that to the best of our knowledge, Bitcoin does not stipulate any censorship mechanisms examining and enforcing the compliance of the Bitcoin wallet smartphone applications. Therefore, this raises two questions: *Will these Bitcoin wallet smartphone applications still keep Bitcoin decentralized? If not, are users aware of the violation?* A security study and a user study were conducted on 10 popular Bitcoin wallet smartphone applications and their users, respectively. Unfortunately, our results show that the answers to both the above questions are no. We have three findings. First, the desirable Bitcoin decentralization is not offered in some wallet applications. These wallet applications do not allow users to directly access the Bitcoin networks; all Bitcoin transactions must be transmitted to the intermediate servers deployed by the application developers through secure channels (e.g., TLS). We call these applications violating Bitcoin decentralization **non-P2P** wallet applications thereafter. Second, we find that the users of some non-P2P wallet applications cannot fully control their Bitcoin wallet private keys. Therefore, the BTCs deposited by the users in their Bitcoin addresses can be transferred by other parties to other Bitcoin addresses without prior user consent. Last but not least, most wallet application users are unaware of the violation of Bitcoin decentralization. Therefore, these users may suffer from various Bitcoin wallet fraud attacks.

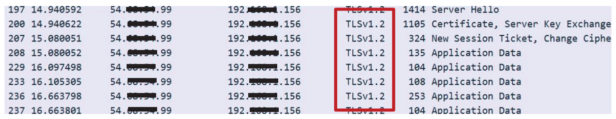
Validation. One experiment was conducted to validate this vulnerability as follows: First, we installed a tested Bitcoin wallet application on a Samsung Galaxy S6 Edge, connected it with the Internet via a home Wi-Fi router, and disabled all background data services on S6. Second, we used a packet capture tool, `tcpdump`, to capture all packets sent by and destined to S6 on the home Wi-Fi router. Third, we started the tested Bitcoin wallet application.

Figure 5(a) and 5(b) illustrate the packets that we obtained from `Bitcoin Wallet` and `Coinbase` wallet applications, respectively.



```
556 30.020775   5.[redacted].40      192.[redacted].156   Bitcoin    386 version, verack, alert
568 30.820424   5.[redacted]40       192.[redacted].156   Bitcoin     78 [unknown command]
569 30.820426   5.[redacted]40       192.[redacted].156   Bitcoin    141 ping, addr
571 30.943731   18.2[redacted]8.166  192.[redacted].156   Bitcoin     98 pong
580 31.866951   5.[redacted]40       192.[redacted].156   Bitcoin     62 pong
583 32.596434   203[redacted]95.10   192.[redacted].156   Bitcoin    168 version
584 32.596435   203.[redacted]5.10   192.[redacted].156   Bitcoin    282 verack, alert
595 32.936328   18.2[redacted]8.166  192.[redacted].156   Bitcoin     98 pong
```

(a) Packets captured on **Bitcoin Wallet**.



```
197 14.940592   54.[redacted].99   192.[redacted].156   TLSv1.2   1414 Server Hello
200 14.940622   54.[redacted].99   192.[redacted].156   TLSv1.2   1105 Certificate, Server Key Exchange
207 15.080051   54.[redacted].99   192.[redacted].156   TLSv1.2    324 New Session Ticket, Change Ciphe
208 15.080052   54.[redacted].99   192.[redacted].156   TLSv1.2    135 Application Data
229 16.097498   54.[redacted].99   192.[redacted].156   TLSv1.2    104 Application Data
233 16.105305   54.[redacted].99   192.[redacted].156   TLSv1.2    108 Application Data
236 16.663798   54.[redacted].99   192.[redacted].156   TLSv1.2    253 Application Data
237 16.663801   54.[redacted].99   192.[redacted].156   TLSv1.2    104 Application Data
```

(b) Packets captured on **Coinbase**.

Figure 5: Packet analysis of two Bitcoin wallet applications (`Bitcoin Wallet` (P2P wallet client) and `Coinbase` (non-P2P wallet client)).

We found that `Bitcoin Wallet` transmitted and received several Bitcoin messages, whereas we did not observe that any Bitcoin messages are transmitted/received from `Coinbase` but a number of TLSv1.2 packets. In particular, `Coinbase` needs to establish a TLSv1.2 connection with an intermediate server, whose IP address is 54.xxx.xxx.99. Our results show that `Coinbase`, `Luno`, `Blockchain Wallet`, `Bitcoin.com Wallet`, `Mycelium`, `Coinomi`, and `BitPay` do not allow their users to directly access Bitcoin networks without intermediate services (non-P2P wallets), whereas `Bitcoin Wallet`, `BRD`, and `Simple Bitcoin Wallet` can do (P2P wallets).

User Study. We conducted a small-scale user study (50 participants) to study if Bitcoin smartphone wallet users have noticed that there are two types of Bitcoin smartphone wallets (i.e., P2P Bitcoin wallet, and non-P2P Bitcoin wallet) in the current market and the later applications violate the Bitcoin decentralization. In this study[3], we interviewed fifty participants. The result shows that only one participant knows that he has to access Bitcoin payment networks through the intermediate servers deployed by his Bitcoin wallet application developer, whereas other participants are unaware of that. We further examined the root causes of this phenomenon. There are two main reasons. First, about one third of participants said that they only downloaded one Bitcoin wallet application based on the downloads and reviews of the application, so they did not notice the existence of two different types of Bitcoin wallet applications. Second, about two thirds of participants said they did use more than one Bitcoin wallet applications. However, these wallet applications provide users with similar user interfaces; Figure 6 illustrates the user interfaces of `Bitcoin Core` (P2P-based client for PCs), `Bitcoin Wallet` (P2P-based client for smartphones), and `Coinbase` (non-P2P-based client for smartphones) while sending and receiving BTCs. Therefore, they did not know that there are two types of Bitcoin wallet applications.

Negative Impact. Another experiment was conducted to study the possible negative impact of the violation of the desirable Bitcoin decentralization. We aim to explore whether the BTCs that non-P2P Bitcoin wallet application users deposited will be transferred to other Bitcoin addresses without prior user consent. Note that the service that we target is Bitcoin wallet services rather than Bitcoin exchange services (e.g., selling BTCs).

The experiment was conducted as follows: First, we deposited 0.0001 BTCs into the Bitcoin address provided by the tested Bitcoin wallet application. Second, we monitored if there are any new transactions associated with Bitcoin address by querying a public online Bitcoin transaction database, Blockchain.Info [8]. The monitoring process lasted for a week. Our results showed that the BTCs deposited by us were transferred to other Bitcoin addresses after about 27 hours without our permission. Similar results were also observed on another non-P2P-based wallet application, `Luno`. Our results show that not all Bitcoin wallet smartphone application users have the full control of their Bitcoin wallet private keys.

Moreover, one thing worth discussing is that seemingly, the violation of the desirable Bitcoin decentralization is not necessarily a vulnerability since the non-P2P wallet applications may be benign and the P2P-based wallet applications supporting Bitcoin decentralization can be malicious. However, on the second thought, it is not the case due to the following reasons. First, no matter whether the non-P2P wallet applications are benign or not, their proxies/servers may be compromised by adversaries and all Bitcoin wallet private keys of their users are thus leaked to adversaries. In contrast, the operations of P2P Bitcoin wallet applications do not rely on any intermediate proxies/servers but decentralized Bitcoin networks. Second, it is challenging for a malicious P2P-based wallet application to bypass conventional Deep Packet Inspection (DPI) security mechanisms since all messages transmitted by P2P-based applications are plain-text Bitcoin messages rather than anything else, the message-by-message investigation can be thus easily conducted to identify suspicious activities. Last but not least, the violation Bitcoin decentralization violation provides an effective means for non-P2P wallet application providers to ***spy on user information*** (e.g., locations, smartphone models, IP addresses, serving WiFi/cellular networks), which threatens Bitcoin user anonymity.

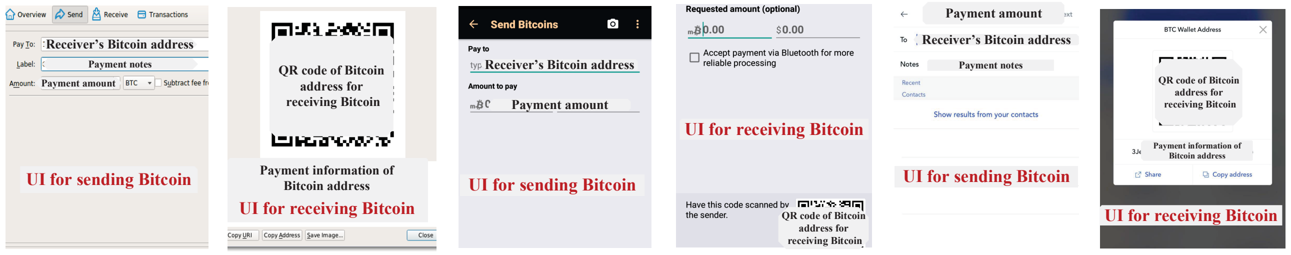
6 PROOF-OF-CONCEPT ATTACKS

This section presents three proof-of-concept attacks.

6.1 Bitcoin Deanonymization Attack

This attack aims to deanonymize anonymous Bitcoin addresses and transactions of a Bitcoin wallet smartphone application user. In recent years, Bitcoin deanonymization is a popular research topic. Gervais et al. [34] have demonstrated that if an SPV client (a Bitcoin wallet app) possesses a small number of Bitcoin addresses (e.g., < 20), the probability that adversaries can correctly guess all those Bitcoin addresses using a single *filter* from the intercepted `filterload` message is 80%, while the *filter* uses a targeted false positive rate of 0.05%. However, the prior art [34] has three key limitations. First, adversaries may not be able to intercept the filter transmitted by victims if adversaries cannot continuously monitor the victims for a very long time. Second, the accuracy of guessing all of the victim's Bitcoin addresses largely varies with the number of Bitcoin addresses that the victim uses. For example, when an SPV client owns 50 Bitcoin addresses (the targeted false positive of 0.05%

[3]We obtained the IRB approval for this study comprising four questions: (**Q1**): Have you ever used any Bitcoin wallet smartphone applications? (**Q2**): Did you know that there are two types of Bitcoin smartphone wallet apps (i.e., P2P/Non-P2P Bitcoin wallet)? Note that P2P Bitcoin wallet apps allow users to directly access Bitcoin networks. Non-P2P Bitcoin wallet apps have to access Bitcoin network through third-party intermediate servers. (**Q3**): If you answer YES in Q2, can we know how you notice these two types? (**Q4**): If you answer NO in Q2, can we know the possible reason (e.g., only using one app and there is no comparison, only a few differences between applications)?


Receiver's Bitcoin address
Payment notes
Payment amount
UI for sending Bitcoin
QR code of Bitcoin address for receiving Bitcoin
Payment information of Bitcoin address
UI for receiving Bitcoin
Send Bitcoins
Pay to
Amount to pay
Requested amount (optional)
Accept payment via Bluetooth for more reliable processing
Have this code scanned by the sender.
Show results from your contacts
BTC Wallet Address
Share
Copy address

(a) Sending BTCs (b) Receiving BTCs (c) Sending BTCs (d) Receiving BTCs (e) Sending BTCs (f) Receiving BTCs

Figure 6: Comparison of the user interfaces of `Bitcoin Core` (the official P2P client on computers released by https://bitcoin.org, the leftmost two figures), `Bitcoin Wallet` (a P2P SPV client on smartphones, two figures in the middle), `Coinbase` (a non-P2P SPV client on smartphones, the rightmost two figures).

will be achieved), the probability that adversaries correctly guess all those Bitcoin addresses is significantly reduced to $6.67 \times 10^{-212}\%$[4] while considering 714.9 million Bitcoin addresses have been added to the Bitcoin Blockchain until 09/01/2020 [20]. Third, the prior art did not identify the real identities (e.g., names) of Bitcoin wallet application users and the real-world damage is thus limited.

Our Bitcoin deanonymization attack aims to address the above issues. By exploiting the vulnerability V1, adversaries can force victims' SPV clients to actively transmit multiple distinct `filterload` messages; each message has a filter created with a default false positive rate of 0.001% which is 50 times smaller than [34]. By leveraging the multiple filters, adversaries can reduce the error rate of identified Bitcoin addresses to an acceptable rate (e.g., three distinct filters lead to the error rate of $10^{-13}\%$). Specifically, in this attack, the number of falsely recognized Bitcoin addresses is 0.00000072 while considering all Bitcoin addresses used (i.e., 714.9 million). With the low error rate, the identified Bitcoin addresses thus can be developed as a new unique wallet fingerprint; it is extremely unlikely for two users to generate the same Bitcoin address [16]. Moreover, by exploiting the vulnerabilities of the popular Wi-Fi calling service (a.k.a Voice over WiFi, VoWiFi)[5] on smartphones, adversaries can discover the Bitcoin wallet application user's identity. Note that the Wi-Fi calling service, which has been broadly supported by many cellular network operators in 52 countries until February 2019 [14].

In the following, we first introduce adversary assumptions and then present our attack design, implementation, and evaluation. Finally, we discuss attack implications.

Adversary assumptions. We assume that the victims' smartphones support Wi-Fi calling services and adversaries can deploy Wi-Fi networks and surveillance cameras near victims in public areas (e.g., Starbucks, Walmart, and Mcdonalds). In practice, it is not very difficult for adversaries to achieve that. For example, the adversary can deploy a rogue AP (e.g., impersonating Starbucks' Wi-Fi AP) by enabling mobile hotspot services on his/her smartphone while using the smartphone as a surveillance camera facing towards the

[4]The number of false Bitcoin addresses is 357,450 (= $714.9 \times 10^6 \times 0.05\%$) and the probability that the adversaries correctly guess user's all Bitcoin addresses is thus 6.67 $\times\, 10^{-212}\%$ (= $C^{50}_{50}/C^{50}_{357450}$ = 1/ (1.5×10^{213})).

[5]The VoWiFi allows mobile users to access cellular network voice/text services through public/private Wi-Fi networks.

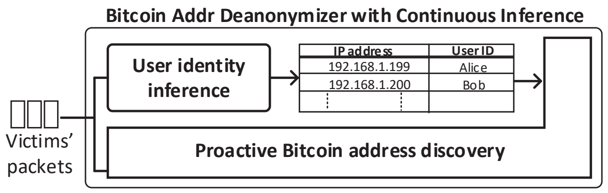


Figure 7: The overview of Bitcoin deanonymization attack.

victims. More discussions on the applicability of this attack will be provided at the end of this section.

Attack design. Figure 7 illustrates the flow chart of this attack. First, as we assume in §4, adversaries can intercept all packets of victims. Second, to analyze these packets, we develop a *Bitcoin Address Deanonymizer* which consists of two modules: (1) user identity inference module and (2) Bitcoin address discovery module. We next elaborate on the designs of the two modules.

1) User identity inference: We leveraged the newly deployed cellular network service, Wi-Fi calling service, and visual techniques to infer the user identity of the owner of a Bitcoin wallet on a smartphone. This module extended [40] to infer the user identity by leveraging unique human call motions (i.e., moving the phone to right/left ear while taking in a call and putting the phone down when a call ends) and visual detection techniques.

This module consists of two major functions. The first function is to infer the IP addresses of the users' smartphones, their Wi-Fi calling events, and event timestamps by analyzing encrypted Wi-Fi calling IPSec packets. [40] has shown that adversaries can identify a Wi-Fi calling user's call event (e.g., dialing a call, answering a call, or talking) by analyzing Wi-Fi calling's IPSec packets. Specifically, by analyzing Wi-Fi users' IPSec packets, adversaries are able to accurately know the time that the owner of a particular IP address is talking over the Wi-Fi calling service (the time that a call starts and the time a call ends). The second function is to use surveillance cameras to record the motions of users, discover the time that a Wi-Fi calling call starts and ends, and identify their user identities by face recognition techniques and online reverse image search engines (i.e., finding people's names by providing photos)[22, 24]. By associating the time that a Wi-Fi calling call starts and ends provided by the first function, we are able to infer the user identity of an IP address (e.g., 192.168.1.75 $\leftrightarrow$ Alice). The output result ⟨*IP address, User Identity*⟩ will be sent to the Bitcoin address discovery module.

2) Bitcoin address discovery: On receipt of ⟨*IP address, User Identity*⟩ sent by the user identity inference module, the Bitcoin address discovery module will first verify if the IP address belongs to an SPV client and then discover all Bitcoin addresses used by the SPV client. By analyzing intercepted packets, it is easy to tell if an IP address belongs to an SPV client since all Bitcoin messages are not ciphered. We further exploited the vulnerability V1 to obtain the victim's multiple filters with low default false positive rates. Therefore, by a given IP address of an SPV client, this module will output a quadruplet ⟨*Bitcoin addresses*, *User identity*, *IP address last used*, *timestamp*⟩. The result will be further updated to a database.

Attack implementation. *1) User identity inference:* We improved Xie *et al.*'s approach [40] to accurately intercept Wi-Fi calling packets of victims and identify their timestamped call events and IP addresses of the victim devices. For example, we could identify that a Wi-Fi calling user uses the IP address of 192.168.1.5 to dial a call at 10:50:10 am and end the outgoing call at 10:50:16 am. Moreover, we applied several computer vision technologies to discovering user identities of users by face recognition and their timestamped calling motions. This comprised two steps. First, an SVM (Support Vector Machine) model was trained to detect users' dialing/talking motions. We provided the SVM with false and positive training videos. For each training video, we assumed that the person could be identified in a person bounding box, and its corresponding HOG (Histogram of Oriented Gradient) descriptor was extracted. We used VLFeat[6] to extract HOG descriptors and train our SVM. Second, once the SVM detects the users' dialing/talking motions, we would recognize the user identity in video frames. We used MatConvNet and Tensorflow as the deep learning libraries for the tiny face detector (i.e., discovering small faces in a video frame) and DR-GAN modules (Disentangled Representation learning-Generative Adversarial Network). *Note that in practice, adversaries can also leverage some online reverse image search engines (e.g., socialcatfish [24]) to discover the identity of a face.*

By correlating the timestamps of Wi-Fi-inferred and Video-inferred calling events, the user identity inference module can associate a user identity with an IP address. The result <*user identity*, *IP address*> is sent to the Bitcoin address discovery module.

2) Bitcoin address discovery: We launched a Wi-Fi de-authentication attack against the victim device and obtained multiple filters generated from distinct *nTweaks*. First, we deployed the Aircrack-ng on a Linux computer and changed our wireless network adaptor card to the monitor mode which allowed the card to discover nearby Wi-Fi routers with their MAC address and SSID (service set identifier of Wi-Fi router, e.g., "Starbucks WiFi"). Second, we performed the command "`airplay-ng` –deauth 0 -c XX:XX:XX:EC:3B:30 -a XX:XX:XX:36:92:10 `wlp3s0mon`" where EC:3B:30 and 36:92:10 were the last three bytes of MAC addresses for the user device and Wi-Fi router, and `wlp3s0mon` was our wireless network adaptor card. Third, after a 5-second attack period, we stopped the attack. We then observed that the disconnected victim device would reconnect to the Wi-Fi router in less than 10 seconds and transmitted

[6]VLFeat is an Open and Portable Library of Computer Vision Algorithms specializing in image understanding and local features extraction and matching. http://www.vlfeat.org/

Performance Matrix	U1		U2		U3		U4	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Call start time error (sec)	0.9	0.01	1.4	0.02	1.0	0.02	1.0	0.02
Call end time error (sec)	0.8	0.01	0.8	0.04	0.8	0.02	1.1	0.02
Identity discovered?	√		√		√		√	

Table 2: The performance of user identity inference module on the 4-user group with users U1, U2, U3, and U4.

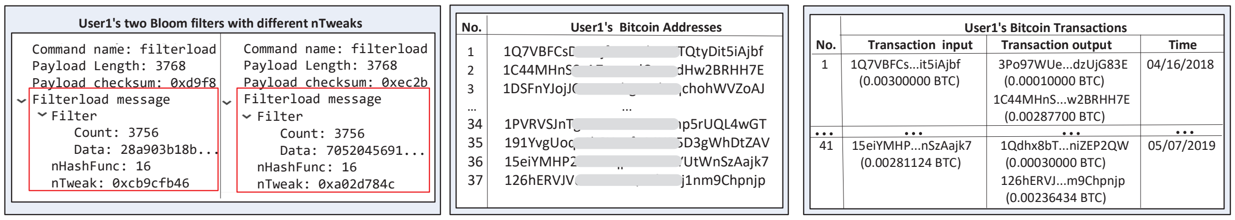
a `filterload` message generated by a new *nTweak*. By repeating Steps 2 and 3, adversaries could obtain multiple `filterload` messages and identify Bitcoin addresses used by the SPV client.

Attack evaluation. We evaluated the performance of the *Bitcoin Address Deanonymizer* as below. First, we invited four students (U1, U2, U3, U4) to participate in this evaluation experiment. All participants provided us with their frontal portraits (as training data) and their user identities (e.g., name). Second, the participants entered a room where a survivance camera and a tested Wi-Fi router was installed. Then they randomly selected one phone from a box containing four tested smartphones (all tested phones were of the same model and the same color). On the tested phones, we activated the Wi-Fi calling service and pre-install the `Bitcoin Wallet`, a Bitcoin wallet application. Third, all participants were asked to dial at least a Wi-Fi calling call and made a Bitcoin transaction that transferred a random amount of BTCs to one of our Bitcoin addresses within 10 minutes. We conducted the experiment for five runs.

1) User identity inference: Our results show that the Wi-Fi calling based user identity inference module can recognize the user identities of four participants in five experiment runs (as shown in Table 2). However, we also observe that the average errors of the call start/end times are about 1.5 seconds (i.e., the error between the Video-based estimated time and the Wi-Fi calling inferred time). This implies that if multiple users dial and end their calls within a 1.5-second interval, the current prototype may not accurately recognize user identity. Nevertheless, this issue can be solved by providing the user identity inference module with more call statistic information, e.g., how long a user keeps silent in a call. This improvement is one of our future works.

2) Bitcoin address discovery: On receipt of the inferred user identity and the user's IP address, this module obtains two filters via the aforementioned Wi-Fi de-authentication attack, applies the two filters to discover all Bitcoin addresses used by the SPV client (Bitcoin wallet application), and retrieves all related Bitcoin transactions. Figure 8 illustrates how this module deanonymizes all Bitcoin addresses and Bitcoin transactions related to a Bitcoin wallet used by the participant U1 in one experiment run. Figure 8(left), Figure 8(middle), and Figure 8(right) show (1) two `filterload` messages generated with different *nTweaks*, (2) Bitcoin addresses which successfully pass both filters, and (3) all Bitcoin transactions which are related to the discovered Bitcoin addresses, respectively. The last Bitcoin transaction (No. 41) was made by the participant U1 to transfer 0.0003 BTC to one of our Bitcoin addresses during this experiment run; the participant U1 confirmed this result.

Implication. The key implication of this attack is multidimensional user privacy leakage. It not only deanonymizes all Bitcoin addresses and transactions of a Bitcoin wallet smartphone user but also provides adversaries with a reliable user tracking mechanism based on the victims' unique application-layer Bitcoin-wallet-based



User1's two Bloom filters with different nTweaks

```
  Command name: filterload          Command name: filterload
  Payload Length: 3768              Payload Length: 3768
  Payload checksum: 0xd9f8          Payload checksum: 0xec2b
⌄ Filterload message              ⌄ Filterload message
  ⌄ Filter                          ⌄ Filter
      Count: 3756                       Count: 3756
      Data: 28a903b18b...               Data: 7052045691...
    nHashFunc: 16                     nHashFunc: 16
    nTweak: 0xcb9cfb46                nTweak: 0xa02d784c
```

No.	User1's Bitcoin Addresses
1	1Q7VBFCsD[redacted]TQtyDit5iAjbf
2	1C44MHnS[redacted]dHw2BRHH7E
3	1DSFnYJojJ[redacted]chohWVZoAJ
...	...
34	1PVRVSJnT[redacted]np5rUQL4wGT
35	191YvgUoq[redacted]5D3gWhDtZAV
36	15eiYMHP2[redacted]YUtWnSzAajk7
37	126hERVJV[redacted]j1nm9Chpnjp

	User1's Bitcoin Transactions		
No.	Transaction input	Transaction output	Time
1	1Q7VBFCs...it5iAjbf (0.00300000 BTC)	3Po97WUe...dzUjG83E (0.00010000 BTC) 1C44MHnS...w2BRHH7E (0.00287700 BTC)	04/16/2018
...	...	...	...
41	15eiYMHP...nSzAajk7 (0.00281124 BTC)	1Qdhx8bT...niZEP2QW (0.00030000 BTC) 126hERVJ...m9Chpnjp (0.00236434 BTC)	05/07/2019

Figure 8: Deanonymize a Bitcoin wallet user - the participant No.1 (Left: two `filterload` messages with different *nTweaks*, Middle: all Bitcoin addresses used by this wallet, Right: all Bitcoin transactions produced by this wallet).

fingerprints (i.e., Bitcoin addresses that the victim's Bitcoin wallet application used). The Bitcoin addresses used by an SPV client can be considered as its unique wallet fingerprint for two reasons. First, by the analysis of [16], a Bitcoin address collision (i.e., a Bitcoin address is used by different wallets) only occurs in the next millennium. Second, our attack can accurately discover all Bitcoin addresses used by an SPV client since the false positive rate (the probability that the adversary wrongly identify a Bitcoin address) in our attack can be freely reduced by adversaries to 10^{-5K}, where K is the number of the launched Wi-Fi de-authentication attacks. Note that in practice, users may increase the default false positive rates due to privacy concerns. However, this attack still accommodates the high default positive rates. For example, if the rate is set to 50%, the adversaries can significantly reduce the rate to 0.097% by launching the attacks 10 times.

Impractical Attacks? Seemingly, this attack may not be practical since it requires adversaries to deploy surveillance cameras and Wi-Fi networks to monitor smartphone users' activities. Although it is not very challenging for adversaries to launch this attack in some public places and monitor mobile users for a certain time period (e.g., a few hours). However, if the monitored users do not access any Wi-Fi calling services in the attack time, adversaries cannot deanonymize the identities of the owners of Bitcoin wallet applications and associate the identities with Bitcoin wallet fingerprints (i.e., the used Bitcoin addresses). Therefore, adversaries have no incentives to launch this attack on a large-scale and the real-world impact of this attack is thus limited.

However, on the second thought, it is not the case due to three reasons. First, according to recent reports/studies [36, 37], Bitcoin is one of the most popular cryptocurrencies that criminals use to bypass the supervision of monetary and law enforcement authorities to launder dirty money, scam victims, defraud users, receive ransom, to name a few. To effectively prevent criminals from disrupting the financial order and thwart cyber attacks, we believe that the monetary and law enforcement authorities have incentives to discover the real identities of Bitcoin wallet users. Second, nowadays, for the sake of public safety, it is not rare to observe surveillance cameras in our daily life. For example, there are about 627,000 and 1,150,000 CCTV cameras deployed in London (UK) and Beijing (China), respectively [25]. Moreover, in some countries, the governments have applied face recognition techniques and the deployed surveillance cameras to trace criminals, e.g., China [9], U.S.A (Chicago, Detroit) [12]. Third, in the era of smart everything, several cities (e.g., New York, Moscow, Hong Kong, Seoul, Paris) and countries (e.g., UK, Switzerland, Singapore, Denmark) have deployed public Wi-Fi hotspots for their residents [5]. A recent industry report [21] forecasts that the number of global public Wi-Fi hotspots will be increased to 628 million in 2023 from 169 million hotspots in 2018, which is a fourfold increase, and around 71% of mobile communication flows over Wi-Fi, which leads to the rising demands in the Wi-Fi calling market [17].

Therefore, we believe that to defend against illegal financial transactions/trades, the monetary and law enforcement authorities have incentives to launch the Bitcoin deanonymization attack. However, this does not mean that we advocate that the privacy of Bitcoin wallet application users can be compromised regardless of what the reason is.

6.2 Bitcoin Reflection and Amplification Spamming Attack

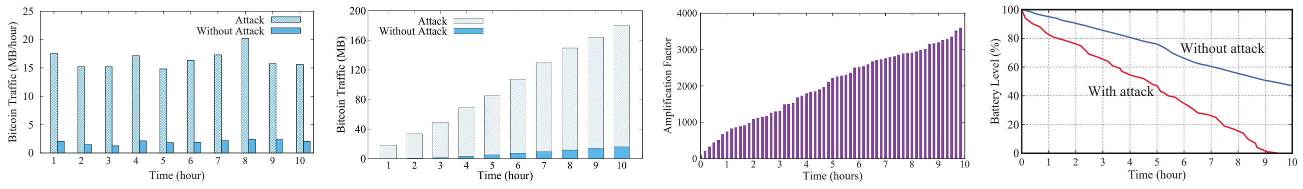
This attack aims to leverage benign Bitcoin full node clients to introduce continuous, unwanted Bitcoin traffic (e.g., 14-20 MB/hour) to victims, which causes an increased cellular network data bill and 96% more power consumption. By exploiting V2, adversaries thus can enforce the Bitcoin wallet applications to continuously download the unwanted Bitcoin transactions from the connected Bitcoin FNCs. In the following, we first briefly present the design of this attack, discuss its implementation, and finally mainly evaluate its negative impact.

Adversary assumptions. The adversary can intercept and modify the Bitcoin messages transmitted between victims and Bitcoin FNCs with which the victims connected.

Attack design. This attack works as follows. First, adversaries intercept `filterload` messages sent by Bitcoin wallet applications (e.g., `Bitcoin Wallet`) to the connected Bitcoin FNCs. Second, adversaries can modify the intercepted `filterload` messages by changing the fields *Data* and *nFlags* to 0xFF..FF and 0, respectively, since this message does not support the encryption and integrity protection. The modifications are used to tell the Bitcoin full node clients that the user is interested in all Bitcoin transactions. The adversaries further send this spoofed message to the Bitcoin FNCs.

Attack implementation. We implemented this attack by developing a *BitCoinTrudy* server on top of *Trudy* [4]. The *BitCoinTrudy* server was deployed between the victims and Bitcoin FNCs, which helped us to identify and intercept `filterload` messages, modify the *Data* and *nFlag* fields, update the checksum of Bitcoin message and TCP checksum accordingly, and deliver the spoofed `filterload` messages to the connected Bitcoin FNC.

Attack evaluation. We evaluated the damages of the spamming attack with two metrics: (1) spamming traffic volume and (2) power consumption of the victim phones. The evaluation experiment was



(a) Bitcoin traffic per hour. (b) Accumulated Bitcoin traffic. (c) Amplification factor observed. (d) Power consumption.

Figure 9: The volume of Bitcoin traffic and power consumption for phones with/without Bitcoin spamming attack.

conducted as follows. First, we installed a tested Bitcoin wallet application (i.e., `Bitcoin Wallet`) on two Google Pixel XL phones using Android 8.1.0. Second, we created two Bitcoin accounts through `Bitcoin Wallet`. Third, we started the `Bitcoin Wallet` and a battery consumption monitor tool (i.e., Batterystats [11]) which continuously monitored battery usage on two tested smartphones. We recorded the volume of Bitcoin traffic and power consumption of two tested smartphones. Fourth, we used the `BitCoinTrudy` that we developed to launch the devised Bitcoin spamming attack against one tested smartphone. The experiment stopped at the time that the victim's battery power was exhausted.

1) Spamming traffic volume: Figures 9(a), 9(b), and 9(c) plot the Bitcoin spamming traffic per hour, the accumulated Bitcoin spamming traffic, and amplification factor observed on the victim, respectively. We have three observations. First, the phone which is under the attack will receive 14-20 MB Bitcoin traffic per hour, whereas the phone which is not under the attack merely receives 1-2 MB Bitcoin traffic every hour. Second, the victim's phone receives around 164 MB in the 10-hour experiment, whereas the phone which is not under the attack only receives 16 MB. This shows that adversaries can produce continuous Bitcoin spamming traffic towards the victim. If victims are using cellular network data services to access Bitcoin networks, they have to pay for the unwanted spamming traffic. Third, at the end of the experiment, the attack amplification factor is increased to around 3666. Note that the unwanted spamming traffic comes from new Bitcoin transactions produced by all global Bitcoin users during the attack. Thus, the factor may be increased while more users use Bitcoin services.

2) Power consumption: The results of the power consumption for phones which are under and not under the 10-hour spamming attack are plotted in Figure 9(d). Our results show that the phone under the attack consumed **96**% more power than the phone which is not under the attack.

Implication. The major implication of this attack is to provide adversaries with a new attack vector to launch continuous spamming traffic against mobile users. The attack cost is low, 4.5 KB Bitcoin message per hour. However, the attack damage is larger (14-20 MB/per hour) than the cost. The victims suffer from increased mobile service bills and shorter smartphone battery lifetime.

6.3 Mobile Bitcoin Wallet Fraud Attack

According to V3, we developed a non-P2P Bitcoin wallet on mobile devices to launch a Bitcoin fraud attack that stealthily transfers BTCs from one address to another one without user consent. The non-P2P wallet was used to break Bitcoin's decentralized design principle that can secure user wallet. With a centralized service model, the wallet service provider can easily manipulate its users' Bitcoin accounts without their approval. In particular, a non-P2P Bitcoin wallet smartphone application, *MyBitcoin*, including a client and a server, was developed. The client is a simple Android application providing a simple user interface for users to send and receive BTCs. The server supports Bitcoin wallet and network routing functions. The *MyBitcoin* client first connects with the server through HTTPS and the server further connects with the Bitcoin networks via the Bitcoin protocol. Our evaluation shows that the *MyBitcoin* users can successfully transfer BTCs to other Bitcoin addresses without any issues. This confirms that no additional security mechanisms are adopted by Bitcoin networks to prevent the Bitcoin decentralization violation in practice.

This malicious wallet application can be actively promoted to users since, in practice, users usually choose the application with many downloads or a high review score [28]. We analyzed the downloads of a popular Bitcoin wallet application, `Mycelium`. We found that once the number of downloads is larger than 1,000, the growth rate of downloads is significantly increased. Similar observations were also made on other Bitcoin applications. In other words, if adversaries can increase the number of downloads to more than one thousand, more victims are projected to download the applications. In practice, this is not technically challenging for adversaries to achieve this goal. Many companies (e.g., MobiRink [23]) provide promotion services that increase the number of application downloads and positive reviews of particular applications. According to our discussion with these promotion companies, it costs only less than $100 for adding 500 downloads and 100 positive comments to particular applications. Once victims install malicious applications, they will suffer from various Bitcoin wallet fraud attacks. However, unfortunately, due to the restrictions of our IRB, we were unable to develop a real malicious Bitcoin wallet application and assess its real-world negative impact. Therefore, we provide a detailed analysis of this Bitcoin wallet fraud attack, which compares it with other types of fraud attacks as follows:

Comparison with other fraud attacks. This attack does not require sophisticated attacking techniques; however, the negative impact of this attack is far-reaching compared with other types of financial fraud attacks: (1) mobile banking fraud attack and (2) mobile payment fraud attack against smartphone users. The former attack aims to obtain victims' usernames and password and further steals victims' deposits by deploying fake mobile banking applications (e.g., `Chsae` is the fake version of the Chase application). The latter attack aims to steal victims' deposits from their bank accounts or use victims' credit cards without their approval by deploying malicious online payment applications similar to Paypal, Venmo, CashApp. We compare the devised Bitcoin fraud attack with these

Attacks		Mobile Banking Fraud ★	Mobile Payment Fraud ◇	Mobile Bitcoin Wallet Fraud
Deployment Difficulty		High	Medium	**Low**
Credit Transactions	Deposit accounts	Users' banking accounts	Users' banking accounts	**App providers' accounts**
Debit Transactions	Send Money without user approval	Yes	Yes	**Yes**
	Send Money Limits	Pre-defined payees: \$50,000-\$250,000 per day Non-pre-defined payees: \$1,500-\$5,000 per day	\$5,000-\$25,000 per transaction	**No**
	Prevent users from receiving fraud transaction alerts?	No	No	**Yes**
Fraud Protection	Can victims get refund from fraud transactions?	Yes*	Yes*	**No**

★: This attack aims to obtain victims' usernames and password and further steals victims' deposits by deploying the fake mobile banking applications of Chase, Bank of America, Wells Fargo, Citi, and U.S.bank.
◇: This attack aims to steal victims' deposits from their bank accounts or use victims' credit cards without their approval by deploying malicious online payment applications. The limitations are based on the study of five most popular mobile online payment applications, such as Paypal, Google Pay, Apple Pay, Venmo, and CashApp.
*: It is likely for victims to recover the loss by filing disputes, however, it varies with the countries of victims.

Table 3: Comparison of fraud attacks between Bitcoin wallet smartphone application and other types of mobile wallets.

two attacks from four aspects: (1) deployment difficulty, (2) credit transactions, (3) debit transactions, and (4) fraud protection.

The comparison results are summarized in Table 3. We have four observations. First, the deployment difficulty of mobile Bitcoin wallet fraud attack is lower than other attacks. For mobile banking fraud attacks, the fake mobile banking applications (e.g, Chsae) need to pass the examination conducted by mobile application stores (e.g., Google Play and Apple Store). For mobile payment fraud attacks, since the money transfer is usually limited to the users using the same mobile payment applications, it is not easy for a new mobile payment application to have a great number of customers in a short time. Second, the money of the users of mobile banking/payment fraud applications is deposited in the users' banking accounts, whereas the money of the users of mobile Bitcoin wallet fraud applications is deposited in adversaries' Bitcoin addresses. Third, all of these attacks are capable of transferring victims' money to other accounts without users' approval. However, the mobile banking and mobile payment attacks have limitations of the amount of money that can be transferred, whereas the Bitcoin wallet fraud attack does not. Moreover, the Bitcoin wallet fraud attack can prevent the victims from receiving any alerts about the fraud transactions, nevertheless, the other two attacks cannot since the alerts are delivered by the financial institutes (e.g., Chase, Citi) of the victims, which are out of adversaries' control. Fourth, the victims of the mobile Bitcoin wallet fraud attacks are unlikely to get the money (i.e., BTCs) stolen by adversaries back since Bitcoin is not a government-insured asset in many countries, whereas the victims of the other two attacks may still have chances to recover their loss by filing disputes to their banks. For example, for credit card users in the U.S., they usually do not need to pay for the abnormal transactions by filing disputes.

7 SOLUTION: BITCOIN SECURITY RECTIFIER

We developed a phone-based Bitcoin Security Rectifier, a smartphone application, to address/mitigate the identified vulnerabilities; this approach does not require any modifications to the existing Bitcoin protocol standards, Bitcoin wallet applications, libraries, and wallet service operations. This approach is motivated by two factors. First, the revise of Bitcoin protocol standards is time-consuming, which is unlikely to be archived and updated to the whole Bitcoin ecosystem in a short time. Second, Bitcoin wallet application service providers may not be willing to change their current operations/designs due to business concerns.

Design. The Bitcoin Security Rectifier will examine all incoming and outgoing Bitcoin messages and take the following actions.

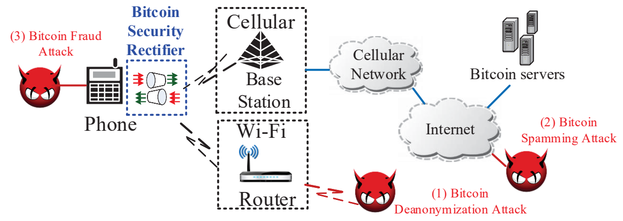
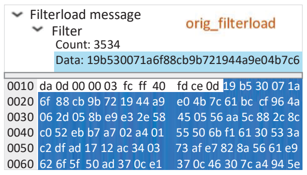


Figure 10: The Bitcoin security rectifier evaluation testbed setup.

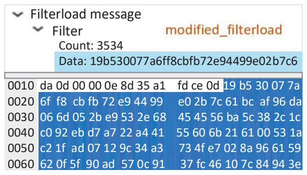
For *V1*, if the Bitcoin message being examined is an outgoing `fliterload` message, it first saves the message into memory, referred to as *orig-filterload*, then creates a copy and adds 10*N Bitcoin addresses which do not belong to the user to the Bloom filter, where N is the number of Bitcoin addresses that have been added by the user to the Bloom filter of *orig-filterload*. The modified `filterload` message is referred to as *modified-filterload*. Finally, the *modified-filterload* will be sent to the Bitcoin network. Using this approach, adversaries cannot obtain *low-false positive-rate* Bloom filters from victims (against the privacy leakage attack). For *V2*, the rectifier checks if Bitcoin transactions carrying `Inv` messages are of the SPV clients' interests by leveraging the cached *modified-filterload* and calculates and monitors the real false positive rate. If the real false positive rate associated with an FNC is higher than the previously configured false positive rate, the rectifier will force the SPV clients to disconnect with the FNC since the FNC may have been abused by adversaries to launch spamming attacks. For *V3*, if the Bitcoin wallet application has been started and there are no Bitcoin messages observed in a pre-defined time period (e.g., 5 mins), a warning message will be sent to the user.

Implementation. The proposed Bitcoin Security Rectifier was written in Java and developed on Android smartphones. The rectifier consists of two components, namely Bitcoin message capturer and Bitcoin security rectification. The Bitcoin message capturer is a light-weight Bitcoin-specific capture service that was implemented on top of NetWorkPacktCapture VPN [10] on smartphones to only intercept Bitcoin messages; for other types of data packets, they will be directly routed to the destinations. The Bitcoin security rectification examines the intercepted Bitcoin messages and modifies Bitcoin messages if needed. Note that the developed rectifier does not require the root privilege but the permission of BIND_VPN_SERVICE.

Evaluation. The evaluation experiment setup is illustrated in Figure 10. First, the Bitcoin security rectifier was installed on a tested smartphone. Second, we deployed a server to launch the Bitcoin deanonymization attack on the Wi-Fi network serving the tested

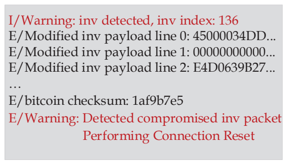


```
Filterload message
  Filter
    Count: 3534
    Data: 19b530071a6f88cb9b721944a9e04b7c6
0010 da 0d 00 00 03 fc ff 40   fd ce 0d 19 b5 30 07 1a
0020 6f 88 cb 9b 72 19 44 a9   e0 4b 7c 61 bc cf 96 4a
0030 06 2d 05 8b e9 e3 2e 58   45 05 56 aa 5c 88 2c 8c
0040 c0 52 eb b7 a7 02 a4 01   55 50 6b f1 61 30 53 3a
0050 c2 df ad 17 12 ac 34 03   73 af e7 82 8a 56 61 e9
0060 62 6f 5f 50 ad 37 0c e1   37 0c 46 30 7c a4 94 5e
```



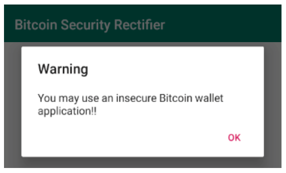
```
Filterload message
  Filter
    Count: 3534
    Data: 19b530077a6ff8cbfb72e94499e02b7c6
0010 da 0d 00 00 0e 8d 35 a1   fd ce 0d 19 b5 30 07 7a
0020 6f f8 cb fb 72 e9 44 99   e0 2b 7c 61 bc af 96 da
0030 06 6d 05 2b e9 53 2e 68   45 45 56 ba 5c 38 2c 1c
0040 c0 92 eb d7 a7 22 a4 41   55 60 6b 21 61 00 53 1a
0050 c2 1f ad 07 12 9c 34 a3   73 4f e7 02 8a 96 61 59
0060 62 0f 5f 90 ad 57 0c 91   37 fc 46 10 7c 84 94 3e
```

(a) Against Deanonymization attack.



```
I/Warning: inv detected, inv index: 136
E/Modified inv payload line 0: 45000034DD...
E/Modified inv payload line 1: 00000000000...
E/Modified inv payload line 2: E4D0639B27...
...
E/bitcoin checksum: 1af9b7e5
E/Warning: Detected compromised inv packet
                    Performing Connection Reset
```

(b) Against Spamming attack.



You may use an insecure Bitcoin wallet application!!

(c) Against Fraud attack.

Figure 11: The evaluation of Bitcoin Security Rectifier.

smartphone. Third, we deployed a server between the tested smartphone and the Bitcoin full node clients with which the phone connected in order to launch Bitcoin spamming attack. Fourth, on the tested phone, we installed the application *MyBitcoin* to emulate the Bitcoin fraud attack. We evaluated if our Bitcoin security rectifier can defend against these three attacks.

The evaluation results are illustrated in Figure 11. For the Bitcoin deanonymization attack, the rectifier can intercept the Bloom filter with a low default false positive rate and protect it by adding Bitcoin addresses that do not belong to the victim (see Figure 11(a)). Therefore, the adversary cannot accurately infer all Bitcoin addresses owned by the victim. For the Bitcoin spamming attack, Figure 11(b) shows that the rectifier can intercept the received malicious Bitcoin inventory message, examine it, and disconnect from malicious Bitcoin FNCs. For the Bitcoin fraud attack, Figure 11(c) confirms that the rectifier successfully detects if the user uses a non-P2P Bitcoin wallet application and provides a security warning for the user.

8 CONCLUSION

Bitcoin wallet smartphone applications, such as `Coinbase`, `Luno`, `Bitcoin Wallet` are increasingly popular nowadays. In this work, we studied the security implications of the 10 most popular Bitcoin wallet smartphone applications. We uncovered three security vulnerabilities among these studied applications. By exploiting them, we devised three proof-of-concept attacks which allow adversaries to (1) deanonymize users' real identities, Bitcoin addresses, and transactions, (2) introduce continuous unwanted Bitcoin spamming traffic towards victims, and (3) launch Bitcoin fraud attacks to take advantage of Bitcoin wallet users. Our analysis shows that the root causes of these security vulnerabilities stem from the improper implementation of Bitcoin wallet applications and the operational slips of wallet service providers. We further developed a phone-side Bitcoin Security Rectifier to help users defend against the identified security threats, without the need of modifying current Bitcoin protocols, infrastructures, and wallet applications and services.

9 ACKNOWLEDGEMENT

This work is supported in part by the National Science Foundation under Grants No. CNS-1815636, CNS-1814551, and CNS-1617412. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors only and do not necessarily reflect those of the National Science Foundation.

REFERENCES

[1] From dell to rakuten: 10 most popular vendors that accept bitcoin. https://cointelegraph.com/news/from-dell-to-rakuten-10-most-popular-vendors-that-accept-bitcoin, 2016.
[2] Murmurhash3 hash function. https://sites.google.com/site/murmurhash/, 2017.
[3] Bitcoin protocol. https://en.bitcoin.it/wiki/Protocol_documentation, 2018.
[4] Trudy. https://github.com/praetorian-inc/trudy, 2018.
[5] 9 cities offer largest free wifi networks. https://interestingengineering.com/these-9-cities-offer-the-largest-free-wifi-networks, 2019.
[6] Aircrack-ng. https://www.aircrack-ng.org/, 2019.
[7] Bitcoinj library. https://github.com/bitcoinj/bitcoinj, 2019.
[8] Blockchain.info. https://www.blockchain.com/, 2019.
[9] Do you know what level of domestic face recognition monitoring is achieved. https://zhuanlan.zhihu.com/p/39868461, 2019.
[10] Networkpacketcapture. https://rb.gy/hntp8w, 2019.
[11] Profile battery usage with batterystats and battery historian. https://developer.android.com/studio/profile/battery-historian, 2019.
[12] Real time facial surveillance. https://rb.gy/ozuzbw, 2019.
[13] A simple arp spoofer for windows. https://github.com/alandau/arpspoof, 2019.
[14] Status of 4g/lte and lte-a networks globally. http://www.haddentelecoms.com/sites/default/files/2019-02/Status-of-LTE-networks-globally-02-2019.pdf., 2019.
[15] Tool 78: Reset every tcp packet. http://www.cis.syr.edu/~wedu/Teaching/cis758/netw522/netwox-doc_html/tools/78.html, 2019.
[16] Version 1 bitcoin addresses. https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses#Collisions_.28lack_thereof.29, 2019.
[17] Voice over wifi (vowifi) market. https://rb.gy/zktie2, 2019.
[18] Bitcoin. https://en.wikipedia.org/wiki/Bitcoin, 2020.
[19] Bitcoin core. https://bitcoincore.org/en/about/, 2020.
[20] Bitcoin explorer. https://www.blockchain.com/explorer, 2020.
[21] Cisco annual internet report. https://rb.gy/thfl1p, 2020.
[22] Face detection and recognition homepage. https://facedetection.com/, 2020.
[23] Mobirink-start promoting your app. https://mobirink.com/, 2020.
[24] Social catfish. https://socialcatfish.com/reverse-image-search/, 2020.
[25] Surveillance camera statistics. https://rb.gy/anniu4, 2020.
[26] Aiolli, F., Conti, M., Gangwal, A., and Polato, M. Mind your wallet's privacy: identifying bitcoin wallet apps and user's actions through network traffic analysis. In *Proceedings of the 34th ACM/SIGAPP SAC* (2019), pp. 1484–1491.
[27] Antonopoulos, A. M. *Mastering Bitcoin: unlocking digital cryptocurrencies.* " O'Reilly Media, Inc.", 2014.
[28] Askalidis, G., and Malthouse, E. C. The value of online customer reviews. In *Proceedings of ACM Conference on Recommender Systems* (2016), pp. 155–158.
[29] Brengel, M., and Rossow, C. Identifying key leakage of bitcoin users. In *International Symposium on Research in Attacks, Intrusions, and Defenses* (2018), Springer, pp. 623–643.
[30] Conti, M., Gangwal, A., and Ruj, S. On the economic significance of ransomware campaigns: A bitcoin transactions perspective. *Computers & Security 79* (2018), 162–189.
[31] Conti, M., Kumar, E. S., Lal, C., and Ruj, S. A survey on security and privacy issues of bitcoin. *IEEE Communications Surveys & Tutorials 20*, 4 (2018), 3416–3452.
[32] developers, B. W. Bitcoin wallet. https://play.google.com/store/apps/details?id=de.schildbach.wallet&hl=en_US, 2019.
[33] Gangwal, A., and Conti, M. Cryptomining cannot change its spots: Detecting covert cryptomining using magnetic side-channel. *IEEE Transactions on Information Forensics and Security 15* (2019), 1630–1639.
[34] Gervais, A., Capkun, S., Karame, G. O., and Gruber, D. On the privacy provisions of bloom filters in lightweight bitcoin clients. In *Proceedings of the 30th Annual Computer Security Applications Conference* (2014), pp. 326–335.
[35] Gervais, A., Karame, G. O., Capkun, V., and Capkun, S. Is bitcoin a decentralized currency? *IEEE security & privacy 12*, 3 (2014), 54–60.
[36] Lee, S., Yoon, C., Kang, H., Kim, Y., Kim, Y., Han, D., Son, S., and Shin, S. Cybercriminal minds: an investigative study of cryptocurrency abuses in the dark web. In *NDSS* (2019), Internet Society, pp. 1–15.
[37] Liao, K., Zhao, Z., Doupé, A., and Ahn, G.-J. Behind closed doors: measurement and analysis of cryptolocker ransoms in bitcoin. In *APWG Symposium on Electronic Crime Research (eCrime)* (2016), pp. 1–13.
[38] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. Tech. rep., Manubot, 2019.
[39] Turuani, M., Voegtlin, T., and Rusinowitch, M. Automated verification of electrum wallet. In *International Conference on Financial Cryptography and Data Security* (2016), Springer, pp. 27–42.
[40] Xie, T., Tu, G.-H., Li, C.-Y., Peng, C., Li, J., and Zhang, M. The dark side of operational wi-fi calling services. In *IEEE CNS* (2018), pp. 1–1.