Attack scenarios and countermeasures

* Using anonymity overlay networks like Tor [1]or mix networks like Loopix[2] is a common mitigation for deanonymization attacks based on network analysis.
* Tor helps to hide the connection between the originating node's IP address and the first node to broadcast the transaction to the peer-to-peer network, but we cluster transactions based on the entry nodes of the first broadcaster rather than the originating node's IP address. Keep in mind that broadcasting transactions through Tor can even lead to further man-in-the-middle vulnerabilities [3].
* Depending on the style of the user's wallet, we separate the scenarios into three categories.

1. Full node with incoming connections (server): A company or an enthusiast eager to volunteer their processing power to benefit the network typically runs a Bitcoin server. In the first scenario, numerous customers of this company may be involved in the transaction relayed through the node, which is damaging to their privacy.
2. Full node without incoming connections: Depending on the collection of entry nodes, transactions coming from a complete node that has no incoming connections may be clustered. The user can restart the software after completing a transaction to avoid this so that each transaction is broadcast across a fresh set of entry nodes.
3. Light wallets: The majority of Bitcoin users employ lite wallets, or minimal payment verification, to outsource validation to another complete node (SPV). From a networking standpoint, the majority of light wallets, particularly those that are portable, do not even connect to a P2P network. Instead, they broadcast transactions to the P2P network by sending them via TLS to the wallet provider's server.

* The final phase of an anonymous transaction is when a wallet processes new transactions, and this is where we found the most applicable and widespread side-channel attacks. The unlikability and anonymity guarantees of the system can be violated by remote adversaries using these attacks.

**Attack Type I: Side-Channels at the Receiving Party**

* Attacks like cache side-channel attacks could be used by an adversary co-located with a user's wallet. Such adversaries, however, are expressly excluded from the threat model taken into account by Monero and Zcash [4].
* Goals of the attacker include figuring out whether two transactions pay the same address and how a known user connects to the P2P network.
* We take into account two distinct assault scenarios: To find out which P2P node (or wallet) the key's owner uses to accept transactions, the adversary knows an anonymous public key and sends a transaction to it. An honest user sends a transaction for which the intended payee's public key and identity are unknown to the adversary. Which P2P node (or wallet) is utilised by the transaction's payee is determined by the opponent.
* Because the attacker can transmit legally designed transactions to a known public key, the latter attack scenario incorporates the first. A break in transaction unlikability results straight from the second case. The attacker only has to know if the payees of two transactions that have been sent into the network share the same P2P node or wallet. Both attack scenarios also compromise user anonymity and can be leveraged for further privacy violations:

1. IP address recovery. If the owner doesn't utilise anonymizing software like Tor, the adversary can connect a public key to the IP address of her P2P node (or her wallet if it connects to a remote node). The victim can be geo-localized or de-anonymized using this information.
2. Diversified address linkability. An attacker can identify whether two public keys are for the same user if they are given two public keys. The attacker attempts to identify the same node or wallet by sending a transaction to each public key. The unlinkability feature of varied addresses is violated by this.
3. Private key recovery. The flaws in several of our assaults also provide opportunities for side-channel timing extraction of a victim's secret "viewing" key. If this key is stolen, the attacker can link all transactions sent to the victim passively (but not take money from the victim).

**Attack strategies.**

Strategy 1: Analyzing wallet-to-node communication traffic. A network adversary or remote node adversary can passively watch changes in the wallet-to-node interaction if a wallet connects to a distant node.

Strategy 2: wallet behaviour can be inferred from the P2P layer. Co-locating the wallet and node prevents a remote adversary from watching how they interact. Yet, information still leaks to the adversary if changes in wallet behaviour affect how the user's P2P node communicates with distant peers.

Both approaches function when a transaction is made, sent into the P2P network, and when it is a part of a block. Wallets then reprocess the transactions to make sure they are genuine (i.e., they did not double spend), and the block and all of its transactions are then shared with each peer.

**Attack Type II: Side-Channels at the Sending Party**

* This section includes more conceptual criticisms. These attacks, however less likely to affect present users, serve as a reminder of the significance of having side-channel-free cryptographic implementations for the long-term and comprehensive security of anonymity-preserving systems.

**Attack strategy.** We take into consideration a cryptographic timing attack that takes use of temporal variations in arithmetic operations based on the values of the operands. Prior to this study, such attacks for many cryptographic primitives have not been taken into account for zk-SNARKs. We take advantage of the correlation between the length of the proof's production and the significance of the prover's witness. We anticipate that the transaction amount, which is contained in the witness, will be tied to the duration of the proof. In the proofs for Zcash, for instance, the transaction amount is divided into bits, and for each non-zero bit, an elliptic curve operation is computed. Hence, the transaction amount's Hamming weight, which is connected with its value, and the proof time are both highly correlated [5][6][7].

**THE TRANSACTION FLOODING ATTACK**

* This attack analyzes the ring signature mechanism of Monero, which hides the genuine input keys by combining them with various output keys (used as decoy keys) produced by earlier transactions. The transaction flooding attack's fundamental concept is straightforward. A large knowledge base, or set of output keys, must be accumulated by the attacker in order for the system to choose keys to be utilised as mixins in upcoming transactions.
* In order to establish an input ring of size 11 in Monero, each input must have 10 mixins in addition to the real spend key for each new transaction. The system adds the mixins to the transaction's input after selecting them from the output keys of earlier transactions using a decoy picker that makes use of a gamma distribution [8].

**The Attacker Model**

* The attacker can obtain blockchain data since Monero's blockchain data is open to the public and available to anybody. In order to be able to track transaction inputs, we presume that the attacker is prepared to pay transaction costs.
* In order to flood the network, we also assume that the attacker has access to at least two separate Monero addresses. The quantity of XMR required to cover the fees paid for the attack transactions must be present at one of those addresses.
* The other addresses will be used to store output keys and accept transactions. Keep in mind that setting up a new Monero wallet is simple and free.
* Lastly, we suppose that the attacker can initiate as many transactions as he likes, so long as he pays the transaction costs, at any given time. There is no timing guarantee, thus it is up to the miners to choose and validate the transactions. If the network's transaction pool contains transactions that are awaiting confirmation that pay higher fees, the miners will choose those first.

YASHWANTH’S PART

RESULTS FOR LITECOIN, STEP 1

1,737 data points covering the price of Litecoin were gathered between August 24th, 2016, and May 26th, 2021. These data values are transformed into input format for window sizes of one day, three days, seven days, and thirty days. The training data size is set at 1,200 data points for each of these various window sizes, and the remaining 1,737 data points are used to evaluate the performance of the model. The prediction of testing data points—which vary and are also connected to specific window sizes—is used to compute the MSE loss. For the purpose of 1-day price forecast, for instance, 1,200 input pairs are used for training and 536 are used for testing.[1]

The standard LSTM and GRU models provide MSE losses of 0.02085 and 0.02113, respectively, for 1-day window sizes, compared to the suggested model's MSE loss of 0.02038. FIGURE 3a depicts the loss contrast for the 1-day prediction window.

Litecoin's 1-day price forecast by various models is shown on a time series graph in FIGURE 3b along with current prices.

The testing input pairs total 534, and the suggested model's MSE loss is 0.02103, for a window size of three days. Both the standard GRU and the LSTM have MSE losses of 0.02210 and 0.02131, respectively. FIGURES 4A and 4B depict the time series data for Litecoin and the MSE loss comparison of the suggested model, LSTM, and GRU, respectively.

For a window size of 7 days, there are 530 input combinations that are being tested. In comparison to the MSE losses of the LSTM and GRU, which were 0.02545 and 0.02409, respectively, the suggested model's MSE loss is 0.02337. FIGURE 5a shows a bar graphic comparing the losses of three different models, and FIGURE 5b shows a time series comparison.

The MSE loss for 30 days is 0.026375 and the testing input pairs are 507, respectively. The suggested model's losses are contrasted with the losses of the LSTM and GRU, which were 0.02800 and 0.02716 respectively, in FIGURE 6a.

A time-series comparison of all three models using real data from April 27 to May 26 is also shown in FIGURE 6b.[2]

(2) ZCASH RESULTS

A total of 1,671 data points covering the price of Zcash were gathered between October 29th, 2016, and May 26th, 2021. These data values are transformed into input format for window sizes of one day, three days, seven days, and thirty days. The training data size is set at 1,200 data points for each of these various window sizes, and the remaining 1,671 data points are used to evaluate the performance of the model. The prediction of testing data points—which vary and are also connected to specific window sizes—is used to compute the MSE loss. [3]

Establishing connections between nodes. When a Node joins the Zcash network it needs to connect to existing nodes on the network. To establish these connections the Node initiates a TCP handshake with these nodes. To achieve this, the only information the Node needs is the IP address of the network nodes. At the end of this process the Node will have established incoming and outgoing connections to Peer1 . . . Peerk. Receiving and sending transactions. Nodes on the Zcash p2p network follow a three-step protocol to propagate transactions. To send a transaction to a peer, the Node first sends just the transaction hash to Peer1, . . . , Peerk and will follow up with the entire transaction only if it is requested. In more detail propagating a transaction across the Zcash p2p network involves the following steps: Inventory step: In this step the Node announces the knowledge of a tx to its peers Peeri . The Node sends an INV message which contains Htx to its peers. If this hash was observed before, Peeri simply ignores the INV message, else it proceeds to the get data step. Get data step : Peeri sends a command GETDATA, Htx to Node to request the transaction tx. Note that if a Peeri has requested GETDATA for a particular INV message from Node then it will ignore INV messages for the same transaction from other peers for a specific amount of time (2 min in Zcash3 ) and simply add those INV messages to a queue This time-out serves as a window for the Node to respond to the GETDATA message, and will be relevant to our attack. Send tx : In this step Node responds with the tx for the corresponding INV it sent in the first step. The Node then adds this transaction to its buffer. The following stages are required to propagate a transaction across the Zcash peer-to-peer network in more detail:

Taking an inventory The Node informs its Peeri in this stage that it is aware of a tx. The Node notifies its neighbours via an INV message that includes Htx. Peeri merely ignores the INV message if this hash has already been noticed; otherwise, it moves on to the get data step. obtain info step Peeri requests the transaction tx from Node by sending the instruction GETDATA, Htx.

It should be noted that if a Peeri requests GETDATA from a Node for a specific INV message, the Node will disregard all other peers' INV messages for that transaction for a set period of time (2 minutes in Zcash3) instead of responding to them. Instead, the other peers' INV messages will simply be added to a queue. This time-out, which will be crucial to our attack, gives the Node a chance to reply to the GETDATA message. to text: As a response to the corresponding INV it sent in the previous phase, Node sends the tx in this step. This operation is subsequently added to the node's buffer.[5]

MONERO

During system setup, the system manager assigns n tracing authorities and sets a threshold of k tracing votes as the absolute minimum. For tracing, a private key (ski,TPKE) is given to each specified tracing authority (AUi). We will assume for the sake of explanation that system manager is in charge of the master key mkR2CT = skTC, skTA, and skTPKE. The tracing authorities AUi, however, genuinely have control over skTPKE. Users can mine, spend, and validate a Monero coin in a manner identical to the original RingCT2.0 protocol by generating their keypair via R 2CT-KeyGen[10].

• For the sake of generality, we record mining using BlockGen and BlockVer. It substitutes a collision-resistant chameleon hash CH for the original hash function (such as SHA-256) used in blockchain.

The target block should be found first if the system management wants to reverse the history of remote payments on a chain. After that, move from coarse-grained (transaction and block) to fine-grained (accumulator, commitment) redaction.

To ensure consistency, we wrote redaction into a smart contract and allowed a global redaction policy to determine the redaction's execution order and acceptability standards.

• If the recipient is changed during redaction, the corresponding commitment and address should also be modified; otherwise, the redaction cannot be traced or verified. All subsequent payments from this account should be cancelled if we cancel the payment and prohibit the payer from making more. To do this, we can call for a public account to receive the incoming funds.[11]

IV. ANONYMITY PROTOCOL COMPARISON

Different methods are employed in a complicated blockchain system to achieve anonymity. Only identity anonymity is taken into account for Bitcoin and Ethereum Layer1, the two most well-known cryptocurrency initiatives in blockchain.

Both Bitcoin and Ethereum have suggested mixing services to achieve identity anonymity. Bitcoin has some well-known projects in the mixing business, such as Mixcoin [14] and CoinJoin [15], and Ethereum has Tornado Cash. Because of its retail emphasis, Tornado Cash is more easily compared to other coin mixers for Bitcoin, according to Foxley's analysis [7].

The mixing service is still open to attack, though, as common users' casual use of Tornado Cash or Coinjoin could expose private details about their wallets.

Discussion

While we have demonstrated that our model is more accurate than other models that have been proposed in the literature in predicting the block verification time of Zcash, we also note that it works significantly better on contemporary systems with faster disc access, more RAM, and more CPU cores. Unpredictable access periods result from the Zcash client flushing state to disc more frequently as a result of low RAM. The effect of multi-core, high(er) clock rate CPUs is also apparent. When compared to the HDD benchmark (i = 246.312 s), delays caused by a single verification process, such as verifying a single input, are significantly reduced in the SSD benchmark (i = 61.411 s) [9]

CONLUSION

Because social and psychological factors have an impact on bitcoin pricing, academics have found it challenging to predict future prices. The names of VARIMA, ARIMA, and GARCH

types of time series that are frequently employed to forecast and study the financial sector. Nevertheless, these methods have a number of flaws, such as practicality, which causes accuracy to decline with non-uniform data. Researchers have employed a variety of ML techniques to forecast cryptocurrency values, including SVM, random forest, and KNN. Recently, DL algorithms have produced accurate results in the forecasting of a number of financial markets. The entire scenario has advanced thanks to neural networks.[1]

Secondly, we explore and contrast the various mechanisms used by various cryptocurrencies to achieve anonymity .They are incredibly difficult to ensure transaction anonymity in account-based structures. Yet, due to non-standard usage, identity anonymity is not a guarantee either. Thankfully, Bitcoin and Ethereum layer1 implemented mixing services to improve identity anonymity, however they still required standard usage.

As this is going on, Layer2 has been proposed by Ethereum, which combines zk-SNARK zero knowledge proof with zk-Rollup to address both identity and transaction anonymity. We compared the proof size, prover time, verification time, and transaction confirmation time for zk-SNARK and zkRollup because they both adopted zk-SNARK for the purpose of transaction anonymity [2]