Quantifying the Threat of Sandwiching MEV on Jito: A Measurement of Solana's Leading Validator Client

Nicole Gerzon
Northeastern University
Boston, United States
gerzon.n@northeastern.edu

Ben Weintraub
Northeastern University
Boston, United States
weintraub.b@northeastern.edu

Junbeom In
Northeastern University
Boston, United States
in.ju@northeastern.edu

Alan Mislove
Northeastern University
Boston, United States
amislove@ccs.neu.edu

Cristina Nita-Rotaru
Northeastern University
Boston, United States
c.nitarotaru@northeastern.edu

Abstract

Solana has emerged as a major blockchain platform providing high throughput and low fees. Like other blockchains, Solana can be attacked via so-called "Sandwiching" attacks, where an attacker observes a pending transaction, quickly buys the target cryptocurrency, lets the transaction go through, and then immediately sells it for a profit, skimming that profit from the user who submitted the transaction. While such attacks have been observed by users, they remain underexplored in academic literature on Solana due to technical difficulties studying Solana at scale.

This paper presents a measurement study of Sandwiching attacks using Solana's most adopted validator client, Jito. We develop a methodology to collect Jito data and analyze over four months of data from Jito's use in early 2025, uncovering patterns indicative of both opportunistic and defensive behaviors. Our analysis reveals the ongoing presence of Sandwiching attacks on Jito, finding over 500K instances of Sandwiching attacks resulting in over $7.7M in losses for victims. We also observe users employing defensive behaviors, costing them over $2.4M across our measurement period, that provide little benefit beyond preventing Sandwiching. This demonstrates widespread anticipation of adversarial activity, despite Sandwiching being relatively rare overall. Our findings raise important questions about the perceived versus actual threat of Sandwiching attacks on Solana highlighting the need for more transparent governance around validator-driven extensions.

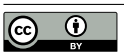
CCS Concepts

• **Networks → Network measurement**; • **General and reference → Measurement**.

Keywords

Blockchain, Solana, Jito, Measurement, MEV, Sandwiching MEV

ACM Reference Format:
Nicole Gerzon, Ben Weintraub, Junbeom In, Alan Mislove, and Cristina Nita-Rotaru. 2025. Quantifying the Threat of Sandwiching MEV on Jito: A Measurement of Solana's Leading Validator Client. In *Proceedings of the 2025 ACM Internet Measurement Conference (IMC '25), October 28–31, 2025, Madison, WI, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3730567.3764493



IMC '25, Madison, WI, USA

ACM ISBN 979-8-4007-1860-1/2025/10
https://doi.org/10.1145/3730567.3764493

1 Introduction

The Solana blockchain is designed to provide higher throughput and lower transaction fees than other blockchains. With 400 millisecond block creation times, an average of 2,500 transactions per second, and a median fee of $0.00064 per transaction [8], Solana has addressed many of the scalability issues of Ethereum and Bitcoin, quickly becoming a major player for online cryptocurrency trading, NFTs, and memecoin markets [3].

However, with the rise in popularity and market cap of blockchain systems, there has been a corresponding rise in manipulation and attacks. One vector for such attacks is manipulation of the transaction ordering in a newly created block: participants may profit by injecting their own transactions or reordering other transactions (if they are the validator) [41]. For example, if there is a proposed transaction to buy a particular cryptocurrency, the validator could conduct a "Sandwiching" attack by inserting transactions to buy that cryptocurrency before the victim and then sell after the victim. This temporarily raises the price for the victim and results in a small, risk-free profit for the attacker [41, 47]. Broadly, such attacks are based on the concept of Maximal Extractable Value (MEV), which refers to the maximum amount of profit that can be earned by manipulating the ordering of transactions within a block on a blockchain. MEV has proven to be profitable: between 2020 and 2024 Ethereum validators accrued approximately $720M[1] [10] through MEV.

Unlike Ethereum, Solana's original design lacks a *public mempool* (i.e., a public list of all pending transactions), making MEV impossible for non-validator users, since only the validators are privy to the information necessary to conduct MEV attacks. This design makes Solana natively resistant to public MEV. However, to enable participants to capture potential MEV revenue and give more profit opportunities to validators, JitoLabs released an alternate validator implementation (Jito) in August 2022, opening up a public mempool, giving all users the ability to capitalize on different MEV opportunities. In tandem, Jito provided reward incentives to

[1]This figure is of September 12, 2025.

Table 1: Example Sandwiching MEV transaction.

Order	Transaction ID	Sender	Action	Token	Amount	Price
1	B	ATTACKER	BUY	TOKEN_A	10,000	\$10 → \$11
2	A	NORMAL	BUY	TOKEN_A	1,000,000	\$11 → \$12
3	C	ATTACKER	SELL	TOKEN_A	10,000	\$12

validators that ran their client (called *Jito tips*). Currently over 97% of Solana validators run a Jito compatible client [14].

In March 2024, JitoLabs changed its policy, suspending the public mempool due to "negative externalities impacting users on Solana" through MEV activity [11]. Despite the fact that Jito closed the highly utilized mempool service, the amount of validator tips earned per day as well as the overall utilization of the Jito network has only increased since then [21]. It has been speculated that this policy change did not, in fact, reduce MEV activity as some validators have re-created the mempool by privately collaborating [29]. In fact, Solana recently banned a number of such validators for "participating in mempools which allow sandwich attacks" [11].

In this paper, we aim to better understand MEV Sandwiching attacks in Solana by examining data from Jito. Similar to other high throughput blockchains, Solana is difficult to study at scale, requiring unique measurement approaches. We collect over four months of data and examine both the prevalence and impacts of Sandwiching MEV attacks, and defensive behaviors by Solana users. Thus, this paper makes the following contributions:

(1) *Measure activity on Jito.* We develop techniques to collect data on Jito bundles, providing visibility into an otherwise opaque aspect of the Solana ecosystem. We use this methodology to collect over four months of data on Jito bundles from early 2025.
(2) *Analysis of Jito data.* We then analyze this sample, finding 521,903 instances of Sandwiching MEV attacks, costing Solana users over \$7.7M.[2]
(3) *Detecting defensive behaviors.* Finally, we examine defensive behaviors by users via *defensive bundling*, a method of MEV protection advertised by Jito. We find that over 86% of Jito bundles containing a single transaction and have insufficient tips to result in priority placement, suggesting that the bundling was done only to avoid MEV attacks.

The remainder of this paper is organized as follows: Section 2 provides background and related work, while Section 3 details our measurement methodology. Section 4 overviews our results and Section 5 provides a concluding discussion.

2 Background and Related Work

Here we provide background on Solana and Sandwiching attacks and overview related work.

2.1 Solana

Solana is a blockchain that can be used to both record and verify transactions, as well as store the terms for executable programs called smart contracts [40]. Solana has its own cryptocurrency—SOL—currently valued at roughly \$242,[3] each of which is further divisible into one billion *lamports*. Solana uses a Proof-of-Stake (PoS) consensus mechanism where users *stake* (i.e., delegate) their SOL with validators to increase the degree of trust in the validator within the network [15]. Unlike other blockchains, Solana's transaction fees are relatively low, starting at 5,000 lamports (0.000005 SOL) for a base fee as well as an optional priority fee paid to the validator for faster transaction acceptance.

Solana's defining features are its high throughput, low latency, and relatively low transaction fees: while Bitcoin has been observed to process roughly 15K transactions per hour, Solana has been observed to process over 4.3M [9]. When compared to other popular blockchains such as Ethereum, whose full ledger is currently 1.4TB in size[3] [43], Solana's full ledger was estimated to be 400TB in April 2025, expected to grow by several terabytes *monthly* [17]. This presents challenge when measuring Solana data at scale: existing APIs impose strict rate limits [6, 32, 34, 37] and running a full archival node incurs a steep cost, including a \$40,000 initial investment, as well as \$3,000 monthly cost [16].

While there is a body of work discussing Solana's contract vulnerabilities and applications [33, 38, 40] as well as comparing Solana to other blockchain mechanisms [27, 36], the only available measurement study on Solana—or any blockchain near Solana's throughput abilities—that we are aware of which dates back to 2022, before multiple Solana updates and the emergence of Jito [30]. That study investigated 12M transactions over 500K blocks in the span of two months [30]; however, since its publication Solana has grown significantly, now producing over 200K blocks with over 80M non-voting transactions *per day* [35].

2.2 Sandwiching MEV

MEV refers to the maximum amount of profit a validator can get from a block on a blockchain based on the ordering of transactions. MEV is made possible through two mechanisms: a DEcentralized eXchange (DEX) that allows users to trade without a central intermediary resulting in volatile currency price changes with every trade, and a publicly visible memory pool (*mempool*) of queuing transactions waiting to be put into a new block [18]. A savvy cryptocurrency trader or bot can look through this publicly visible mempool, see if there are any queuing transactions that will result in the price of a currency changing, and pay a crafted priority fee to push their own transaction into the right slot to benefit from this price shift. While there are versions of MEV that provide overall benefits to a blockchain ecosystem [18], in this paper we focus on the canonical malicious example of MEV: Sandwiching. Sandwiching MEV is when an attacker takes advantage of the dynamic nature of DEX rates to *front-run* the original transaction with a trade that changes the price of the target cryptocurrency and then *back-run* that same original transaction to sell the cryptocurrency immediately after, effectively skimming profit, as can be seen in Table 1.

Unlike Ethereum and Bitcoin, Solana is natively resistant to public MEV as its design lacks a publicly visible mempool [5]. While Sandwiching MEV has never been formally studied on Solana,

[2]This figure uses a SOL to USD conversion rate as of September 12, 2025.

[3]This figure is as of September 12, 2025.

there is a rich body of prior work around identifying and quantifying [31, 45, 46] Sandwiching MEV on other Blockchains. Multiple previous studies have investigated the prevalence and effects of MEV within the Ethereum ecosystem [2, 7, 12, 39, 41, 47]. MEV activities on Ethereum have resulted in over $720M[4] in gains for MEV participants since Sept. 2022 [10]. Sandwiching attacks, in particular, resulted in user losses estimated at $87.7 million in the first half of 2022 alone [20].

While numerous defenses and mitigations have been proposed [28, 42, 44], none are completely effective without sacrificing important usability properties or modifying a blockchain's operational logic. However, users have employed a number of strategies for reducing the risk and impact of Sandwiching attacks, including splitting up larger trades into smaller transactions [46], and properly setting slippage tolerance on trades [19]. Slippage tolerance is a user-set metric for a transaction that describes the maximum price for which to carry out that trade, in case the cryptocurrency's price changed between sending the transaction and its actual execution [19]. Prior work on Ethereum has shown that when slippage tolerance is properly set, it acts as a cap on how much an attacker can extract from the Sandwiching MEV, but cannot fully prevent the attack from occurring [46].

2.3 Jito-Solana

Jito-Solana (Jito) is a third-party extension of the Solana validator client founded by JitoLabs, originally advertised as the 'MEV-powered' extension to Solana [11]. As of Sept. 2025, 97% of the top 500 validator nodes run a Jito-compatible client, including every validator node in the highest staked subset of validators, Solana's current *super-minority* [14]. Jito incentivizes mass adoption by allowing validators to take extra tips (called Jito tips) in exchange for including user-bundled transactions within blocks [22].

Jito allows users, referred to as "searchers", to bundle up to five transactions per request. If this bundle is accepted by the validator, the transactions within the bundle are guaranteed to execute atomically together within the block in the order submitted [22]. This capability allows for the execution of multiple different Solana transactions in a specific order, as is necessary for MEV (recall Section 2.2). Importantly, information on which transactions were originally submitted as a bundle through Jito is not available on Solana's final ledger. Along with existing identifiers for transaction on Solana, (`transactionIds`), Jito bundles are assigned their own ids, dubbed `bundleIds`.

In addition to transaction ordering, until March 2024, Jito provided searchers access to a public mempool of queuing transactions that was contributed to by all validators running the Jito client [11]. Recall from Section 2.2 that Solana's original design lacks a public mempool making it natively MEV-resistant, Jito's public mempool removed this technical barrier to MEV. This public mempool opened up MEV opportunities for users without access to their own validator node or private mempool source. Despite the economic benefits for both searchers and participating validators, JitoLabs discontinued the mempool service citing the negative effects of Sandwiching attacks as the primary reason [11].

Since the closure of this public mempool, online users claim that harmful MEV on Solana remains a pervasive issue and has simply moved to private validator-controlled mempools [25]. The Solana Foundation has taken steps to put validators it detects participating in mempools that allow Sandwiching MEV attacks on a blocklist [29]. In addition to these observations, Jito offers a 'MEV protection' option for transactions, advertising this option as a major benefit to using Jito [23]. Multiple popular Solana trading apps such as Jupiter and BONKBots offer MEV protection settings through Jito [4, 24] along with regular Solana settings for minimizing slippage tolerance. It remains unclear whether these changes indeed removed the impact of Sandwiching MEV on Solana, or if there are active private mempools that still enable such attacks.

In this paper we measure the prevalence of Sandwiching MEV attacks on Jito. These attacks are executed when a victim transaction that was originally submitted to be processed on Solana is instead included in a Jito bundle surrounded by an attacker's transactions that result in a Sandwiching MEV attack on the victim transaction.

3 Methodology

This section describes the methodology behind pulling Jito bundle data and detailed information on bundled transactions, the process of analyzing the bundles for instances of Sandwiching, as well as the criteria for determining defensive behavior on Jito.

3.1 Collecting Jito Data

Despite the fact that Jito is an extension of the Solana validator client, information on which transactions were initially bundled and submitted through Jito is not available on the final Solana ledger. Jito does not provide a publicly documented API to pull any historical data. However, the Jito Explorer website displays a myriad of different historical data representations, relying on an undocumented API to return this information. We reverse engineer the undocumented API calls on this public-facing website by isolating the endpoint responsible for returning the most recent 200 bundles, and change it to to return the most recent 50,000 bundles instead. We use this API call to request the most recent bundles data roughly every two minutes. Our rate of collection was chosen to have reasonable load on Jito's servers while trying to collect as much of the full set of `bundleIds` as possible. We collected data from February 9, 2025 until June 9, 2025. There were a number of periods of time where our data collection was down due to instability or changes to the Jito interface, bugs in our code, or other transient errors (these are noted in the appropriate graphs shaded grey).

Because each request returns data on up to 50,000 bundles, there may be periods of time when "spikes" in the usage result in us missing certain bundles. To determine whether we have indeed collected all of the bundles, we determine if there is any overlap for the bundles returned in successive calls; if any bundles appear in both, we know we have not missed any. We found that, on average, 95% of successive pairs of requests to the Jito API indeed had overlap in the bundles returned; this gives us high confidence that our measurement collected the vast majority of the Jito bundles.

Jito's API endpoint only provides the `bundleIds`, the corresponding `transactionIds` within that bundle, as well as the associated Jito tip; it does not provide the full content of included transactions. With an average of 14.8M bundles and 26M transactions per day

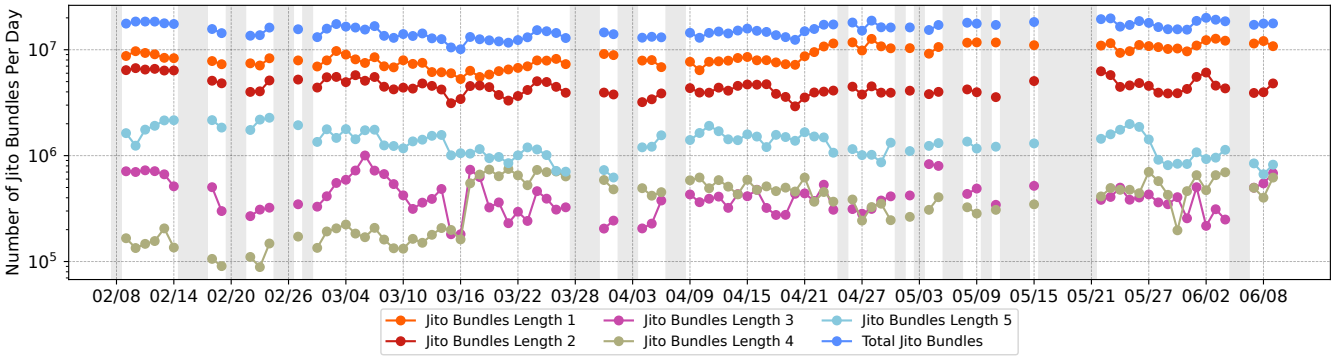


Figure 1: Number of Jito bundles per day, broken down by number of transactions bundled together. Shaded gaps indicate days when there were errors with data collection.

(recall each bundle can contain up to five transactions), collecting transaction data is challenging. There is no public API that would allow us to pull that much transaction data from Solana or Jito, existing RPC providers (Helius, QuickNode, Bitquery, ChainStack, etc.) place restrictions on API calls and "compute units" far below what is necessary for pulling this type of bulk transaction data, even at their highest paid subscription level.

Fortunately for us, another API endpoint on Jito website's is able to return this data for hundreds or thousands of transactions at a time. To limit the load, we request the detailed transaction information only for bundles of length three, which captures the canonical example of Sandwiching behavior with a victim transaction in the middle. These bundles of length three average 2.77% of the total bundles in a day, significantly lowering the number and size of requests when pulling this data. When executing these queries, we only request 10,000 transactions at a time, and space out the requests at least two minutes apart.

Figure 1 shows the breakdown of Jito bundles categorized by number of transactions throughout the measurement period. We can observe the periods of measurement downtime (highlighted in grey). We can also observe that the majority of Jito bundles have length one, containing a single transaction.

3.2 Finding and Analyzing Sandwiching

After collecting the `bundleIds` and detailed transaction data for bundles of length three, we identify bundles which represent Sandwiching attacks. We use the following set of criteria, based on similar heuristics in prior work for Ethereum [31]:

(1) The first and third transaction in the bundle are signed by the same account A, the second transaction is signed by a different account B.
(2) The same set of minted coins is being traded in all three transactions.
(3) The first trade by account A causes the exchange rate to increase for account B.
(4) When looking at the net change in currencies as a result of all transaction within the bundle, account A net gains currency with no payment (that is the profit from the MEV) or ends with net profit when looking at quantity of coin sold.
(5) We exclude bundles where the final transaction is only tipping a Jito validator.[4]

We note that some instances of Sandwiching MEV may include instructions to disguise their intent, such as adding on a fourth unrelated transaction, an unrelated currency trade, or doing multiple sandwiches in one bundle. While we expect these to be infrequent, the amount of Sandwiching reported by our methodology should be treated as the lower bound as our methodology would miss such attacks.

It is important to note that the market value of different cryptocurrencies and memecoins present within these trades often fluctuate second-to-second. There is unfortunately no existing way to find the value of a non-widely popularized coin at the time of transaction execution (aside from executing the transaction yourself and seeing its effects). For this reason, when quantifying the financial harm from Sandwiching MEV, we focus on transactions that trade to or from SOL, a more stable cryptocurrency that can be more reliably translated to an approximate USD amount. Therefore, in cases where all cryptocurrency traded is something other than SOL, we exclude them when calculating the financial impact of an attacker's behavior; this approach also leads to our results that report financial impacts being a lower bound.

3.3 Classifying Defensive Bundling

Users who wish to avoid having their transactions Sandwiched can potentially do so by employing *defensive bundling*. In our described attack scenario, victim transactions are Sandwiched by non-consensual inclusion in an attacker's Jito bundle. This bundling process allows attackers to ensure that the transactions within that bundle execute atomically in the order they defined in the bundle. Bundling not only ensures execution in the right order, as is required for Sandwiching, but also nullifies any potential financial risk for the attacker: if the victim's transaction fails within the bundle, the attacker's transactions within that bundle do not execute.

[4]Not only are these not instances of Sandwiching, but this behavior on-chain is indicative of users utilizing a trading app or smart contract that implements Jito in the backend and simply adds on a final transaction to a bundle orignally length 2 to tip out the Jito validator.

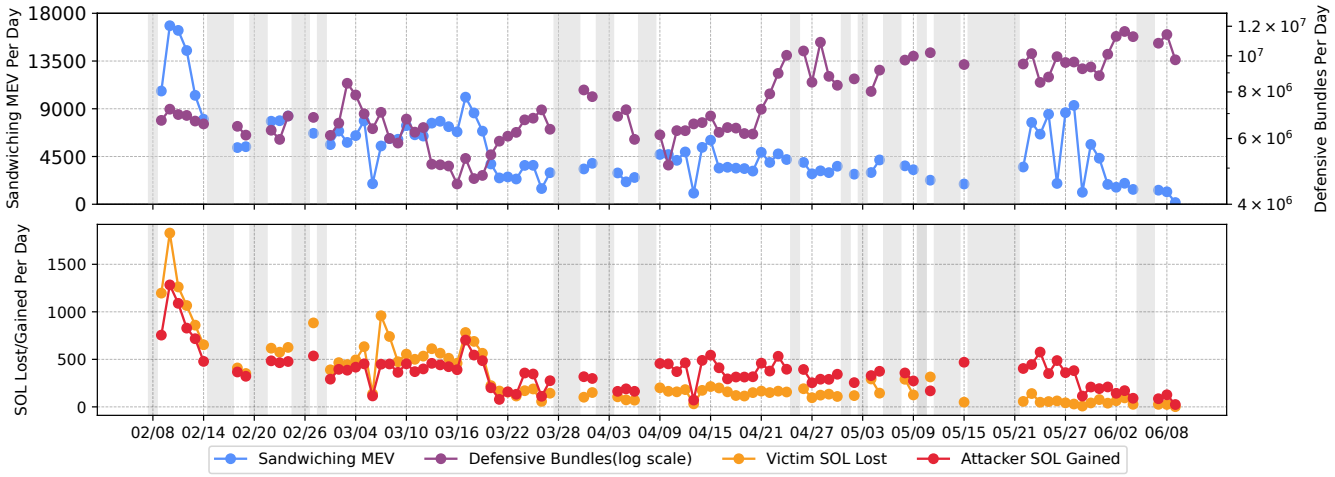


Figure 2: The number of Sandwiching attacks observed each day and the Defensive Bundles per day behavior(top), as well as the victim losses and attacker gains from Sandwiching per day in SOL (bottom).

One option for users to prevent their transactions from Sandwiching is for the user to preemptively bundle their own transaction, ensuring that it cannot be included in an attacker's bundle as bundles cannot be nested on Jito. In other words, instead of submitting the transaction natively in Solana, they can instead put the single transaction inside of a Jito bundle, which prevents other users from including it in a (Sandwiched) bundle but costs a small amount in Jito tips.[5] In fact, Jupiter—Solana's largest and most popular aggregator [13]—offers a "MEV protection" option and, through experimentation with our own transactions, we found that this resulted in the transaction being issued in a Jito bundle of length one.

However, there is another reason why a user might issue a Jito bundle containing a single transaction, which is to increase the priority: Jito bundles include validator tips, which can result in faster transaction processing. The minimum Jito tip that can be spent when bundling is 1000 lamports (0.000001 SOL). Through experimentation with Jupiter we find that depending on market activity the lowest users are allowed to submit is anywhere between 100,0000 and 1,000 lamports. Recent work suggests that even higher Jito tips on length one bundles have a negligible effect on the time-to-confirmation of the bundled transaction, where a high tip is anything above 50% of the 95th percentile tip within a block [1]. Relying on Jito's own dashboard for tracking tipping percentiles within bundles, we find the average 95th percentile tip amount to be ±0.02 SOL (2,000,000 lamports) [26].

Therefore, we distinguish bundles submitted for priority from bundles submitted for MEV protection based on the Jito tip amount, using the minimum amount observed on Jupiter to provide a conservative estimate of this activity. We consider bundles of length one that have a Jito tip at or below 100,000 lamports to be utilizing a version of MEV protection, since there would otherwise be no economic benefit towards paying the Jito tip to bundle the transaction.

[5]It is important to note that this cannot fully prevent a transaction from being frontrun or even Sandwiched, as other techniques could be employed to modify the final transaction ordering, but this technique of defensive bundling would make the particular execution of a Sandwiching attack that is described and studied in this work impossible.

4 Analysis

We now present our analysis of Sandwiching MEV and defensive behaviors by Solana users.

4.1 Sandwiching MEV on Jito

Applying the methodology for identifying Sandwiching MEV attacks described in Section 3.2, we find 521,903 such attacks over the course of our measurement period across all observed bundles. Figure 2 (top) shows the number of attacks detected per day over the measurement period. We can observe a general decreasing trend: while we detected over 15,000 such attacks each day near the start of the observation period, we detect roughly 1,000 per day towards the end of the period.

For instances of identified Sandwiching MEV where the victim and attacker are trading between SOL and a different cryptocurrency, we are also able to quantify the profit lost by the victim of the Sandwiching as well as the amount gained by the attacker. This is done by comparing the conversion rate (between SOL and the traded cryptocurrency) at which the attacker purchases/sells in their first transaction to the rate at which the victim is able to purchase/sell in their following transaction. By multiplying the attacker's rate by the amount purchased/sold by the victim we can see the price the victim would have paid had they not been Sandwiched. Out of the 521,903 instances of Sandwiching, 143,348 (28%) did not include SOL as one of the traded mints, meaning our estimate should be considered a lower bound on the amount of profit/loss.

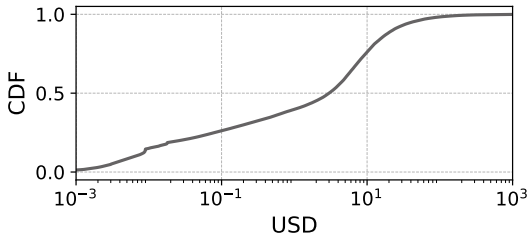


Figure 3: Cumulative distribution of USD lost by users whose transactions were sandwiched in Solana.

Using this analysis we are able to find that Solana users, in aggregate, missed out on at least $7,712,138[6] of revenue due to the effects of these MEV attacks, while attackers gained $9,678,466.[6] [7]

Figure 2 (bottom) shows aggregate attacker gains and victim losses per day for transactions that include SOL trades. Similar to the number of Sandwiching, we see that total losses per day decreased over the measurement period as well.

Next, we take a closer look at victim losses per-transaction. Figure 3 presents the cumulative distribution of amount lost by each Sandwiched transaction. We observe a wide variation in loss among victims, with the median transaction having lost roughly $5,[6] while some transactions lost over $100.[6] This result suggests that the losses incurred by victims were not trivial, and raises the question of whether users took steps to avoid being Sandwiched.

4.2 Defensive Bundling on Jito

Recall from Section 3.3 that we identify users who aim to prevent their transactions from being Sandwiched by looking for Jito bundles of length one that also have Jito tips of less than 100,000 lamports. Figure 4 presents the cumulative distribution of Jito tips for three groups of Jito bundles: all bundles of length one, all bundles of length three, and all bundles we identify as Sandwiching attacks.

We make a number of observations. First, we see that over 86% (864,889,302) of the bundles of length one have Jito tips that are too small to serve as a meaningful incentive for prioritization, suggesting that the purpose of creating a Jito bundle was likely to avoid MEV attacks. We find that throughout our measurement period, users cumulatively spent $2,421,868[6] on this defensive bundling activity. We can observe in Figure 2 that the number of defensive bundles increased throughout our measurement period, which aligns with our previous observation that the number of Sandwiching attacks decreased during the same period.

Second, we see that the bundles where we detect Sandwiching have a very different distribution of tip values compared to all other Jito bundles of length three. While the median length-three bundle has a Jito tip of 1,000 lamports, the median Sandwiching bundle has a Jito tip of over 2,000,000 lamports, a difference of over three orders of magnitude. This dramatically higher tip suggests that attackers

[6]This figure uses a SOL to USD conversion rate as of September 12, 2025.
[7]There are instances when the attacker sells more in the last transaction of the Sandwich than what they bought in the first transaction, this is likely due to constraints set by the victim's slippage

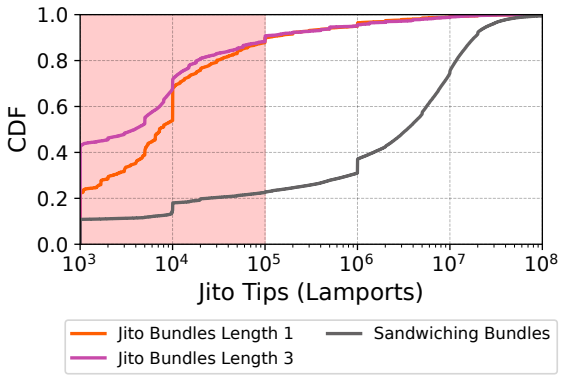


Figure 4: Cumulative distribution of Jito tip amount for bundles of length one and three, as well as for bundles identified as Sandwiching attacks.

are using Jito tips to prioritize their attack bundles, potentially to outbid others attacking the same victim transaction.

5 Concluding Discussion

Despite Solana's native MEV resistance, MEV Sandwiching attacks occur on the platform through at least one underexplored avenue: bundles submitted through the Jito validator client. Our study collects four months of Jito bundle data to identify and quantify both offensive MEV Sandwiching activity and defensive bundling, a feature promoted by Jito to protect against malicious MEV. We find substantial evidence of continued MEV Sandwiching attacks on Solana through the Jito validator client: 521,903 instances of Sandwiching MEV attacks, representing over $7.7M[6] in losses for victims in our measurement period. We observe that Sandwiching attacks appear to get less frequent throughout our measurement period, which may be partially explained by a corresponding increase in defensive bundling activity (Figure 2), but may also be due in part to fluctuations in the price of SOL or Solana's platform popularity throughout the measurement period.

Despite only 0.038% of all measured Jito bundles represent Sandwiching attacks, we observe that users cumulatively spent over $2.4M[6] on defensive bundling activity—in the form of Jito tips that would not be necessary to pay if the transaction was sent through Solana itself—during our measurement period. While this amount is not proportional to the prevalence of Sandwiching MEV observed, the average amount spent on Jito tips per defensive bundle was only $0.0028[6], while the probability of higher losses from successful Sandwich attacks (Figure 3) was much greater. Therefore, despite the low likelihood of being attacked by Sandwiching MEV, the threat of significant loss is sufficient to encourage high use of Jito for protection against MEV.

These findings demonstrate widespread anticipation of adversarial activity among Jito-Solana users and raise important questions about the perceived versus actual threat of Sandwiching MEV attacks on Solana. Our study highlights the need for more transparent governance around validator-driven extensions that can alter native blockchain properties — such as MEV resistance — and potentially introduce new vectors for malicious behavior.

References

[1] [n. d.]. Exploring Solana's updates—stake-weighted QoS, priority fees, and Jito MEV—and their impact on transaction prioritization and landing latency. https://chorus.one/articles/transaction-latency-on-solana-do-swqos-priority-fees-and-jito-tips-make-your-transactions-land-faster [Online; accessed 2024-12-11].

[2] Zeinab Alipanahloo, Abdelhakim Senhaji Hafid, and Kaiwen Zhang. 2024. Maximum Extractable Value (MEV) mitigation approaches in ethereum and layer-2 chains: A comprehensive survey. *IEEE Access* (2024).

[3] Forbes Digital Assets. 2025. *What Is Solana (SOL)? How It Works And What To Know.* https://www.forbes.com/sites/digital-assets/article/what-is-solana-sol-how-it-works-and-what-to-know/ Accessed: 2025-05-15.

[4] BONKbot. 2025. *MEV Protection.* https://docs.bonkbot.io/settings/mev-protection Accessed: 2025-05-15.

[5] Bartek @ Caishen. 2024. *Exploring Sandwich Attacks on Solana.* https://medium.com/coinmonks/exploring-sandwich-attacks-on-solana-442034afc80e Accessed: 2025-05-15.

[6] Chainstack. 2025. Throughput Guidelines. https://docs.chainstack.com/docs/limits. Accessed: 2025-09-12.

[7] Tianyang Chi, Ningyu He, Xiaohui Hu, and Haoyu Wang. 2024. Remeasuring the arbitrage and sandwich attacks of maximal extractable value in Ethereum. *arXiv preprint arXiv:2405.17944* (2024).

[8] CoinLedger. 2025. Solana vs Ethereum. https://coinledger.io/tools/solana-vs-ethereum Accessed: 2025-03-24.

[9] Solana Compass. 2025. *Live Network Performance.* https://solanacompass.com/ Accessed: 2025-05-15.

[10] Cryptic Woods Research. 2024. libMEV: Powerful MEV Explorer Designed for Searchers by Searchers. https://libmev.com/ Accessed: 2025-09-16.

[11] Cryptorank. 2024. *Solana removes several validators for sharing mempool data.* https://cryptorank.io/news/feed/24ec3-solana-removes-validators-mempool-data Accessed: 2025-05-15.

[12] Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. 2020. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *2020 IEEE symposium on security and privacy (SP)*. IEEE, 910–927.

[13] DeFiLlama. 2025. DeFiLlama Trading Report. https://defillama.com/aggregators Accessed: 2025-05-13.

[14] Jito Foundation. 2023. StakeNet Validator History. https://www.jito.network/stakenet/history/ Accessed: 2025-09-12.

[15] Solana Foundation. 2025. Staking and Inflation FAQ. https://solana.com/tr/staking#what-is-staking Accessed: 2025-05-13.

[16] GetBlock.io. 2025. Solana Archive Node Guidelines. https://getblock.io/blog/solana-archive-node-guidelines/. Accessed: 2025-09-12.

[17] GetBlock.io. 2025. Solana Full Node: Complete Guide. https://getblock.io/blog/solana-full-node-complete-guide/. Accessed: 2025-09-12.

[18] Vincent Gramlich, Dennis Jelito, and Johannes Sedlmeir. 2024. Maximal extractable value: Current understanding, categorization, and open research questions. *Electronic Markets* 34, 1 (December 2024), 1–21. https://doi.org/10.1007/s12525-024-00727-x

[19] Lioba Heimbach and Roger Wattenhofer. 2022. Eliminating Sandwich Attacks with the Help of Game Theory. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '22)*. ACM, 153–167. https://doi.org/10.1145/3488932.3517390

[20] Dae Sik Hong. [n. d.]. MEV on Ethereum: A Policy Analysis - International Center for Law & Economics. https://laweconcenter.org/resources/mev-on-ethereum-a-policy-analysis/ [Online; accessed 2024-12-11].

[21] Jito Explorer. [n. d.]. Jito Explorer | Bundles Overview. https://explorer.jito.wtf/. Accessed: 2025-05-14.

[22] Jito Foundation. [n. d.]. Jito Foundation: Solana Liquid Stake Pool. https://www.jito.network/. Accessed: 2025-05-14.

[23] JitoLabs Documentation. [n. d.]. Low Latency Transaction Send. https://docs.jito.wtf/lowlatencytxnsend/. Accessed: 2025-05-14.

[24] JupiterExchange. 2025. *Jupiter Exchange Status Update.* https://x.com/JupiterExchange/status/1831698316557701612?lang=en Accessed: 2025-05-15.

[25] Jack Kubinec. 2024. *Solana should reintroduce a public mempool, researcher says.* https://blockworks.co/news/researcher-encourages-solana-public-mempool Accessed: 2025-05-15.

[26] Jito Labs. 2025. Jito Labs Public Dashboard. https://jito-labs.metabaseapp.com/public/dashboard/016d4d60-e168-4a8f-93c7-4cd5ec6c7c8d Accessed: 2025-05-15.

[27] Xiangyu Li, Xinyu Wang, Tingli Kong, Junhao Zheng, and Min Luo. 2021. From bitcoin to solana–innovating blockchain towards enterprise applications. In *International Conference on Blockchain*. Springer, 74–100.

[28] Dahlia Malkhi and Pawel Szalachowski. 2022. Maximal Extractable Value (MEV) Protection on a DAG. arXiv:2208.00940 [cs.CR] https://arxiv.org/abs/2208.00940

[29] Danny Nelson. 2024. Solana Heavyweights Wage War Against Private Mempool Operators. *CoinDesk* (10 June 2024). https://www.coindesk.com/business/2024/06/10/solana-heavyweights-wage-war-against-private-mempool-operators Accessed: 2025-05-13.

[30] Giuseppe Antonio Pierro and Roberto Tonelli. 2022. Can Solana be the Solution to the Blockchain Scalability Problem?. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 1219–1226. https://doi.org/10.1109/SANER53432.2022.00144

[31] Kaihua Qin, Liyi Zhou, and Arthur Gervais. 2022. Quantifying Blockchain Extractable Value: How dark is the forest?. In *2022 IEEE Symposium on Security and Privacy (SP)*. 198–214. https://doi.org/10.1109/SP46214.2022.9833734

[32] QuickNode. 2025. Predictable Pricing, Designed to Scale. https://www.quicknode.com/pricing. Accessed: 2025-09-12.

[33] Sven Smolka, Jens-Rene Giesen, Pascal Winkler, Oussama Draissi, Lucas Davi, Ghassan Karame, and Klaus Pohl. 2023. Fuzz on the beach: Fuzzing Solana smart contracts. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1197–1211.

[34] Solana Foundation. 2025. Clusters and Public RPC Endpoints. https://solana.com/docs/references/clusters. Accessed: 2025-09-12.

[35] Solscan. 2025. Solana Analytics. https://solscan.io/analytics Accessed: 2025-05-13.

[36] Han Song, Yihao Wei, Zhongche Qu, and Weihan Wang. 2024. Unveiling decentralization: A comprehensive review of technologies, comparison, challenges in bitcoin, ethereum, and solana blockchain. In *2024 IEEE 6th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Vol. 6. IEEE, 1896–1901.

[37] Triton One. 2025. Archival Data Access. https://docs.triton.one/chains/solana/old-faithful-historical-archive-1. Accessed: 2025-09-12.

[38] Jianshu Wang. 2023. Exploring digital timestamping using smart contract on the Solana blockchain. In *Second International Conference on Green Communication, Network, and Internet of Things (CNIoT 2022)*, Vol. 12586. SPIE, 184–190.

[39] Ben Weintraub, Christof Ferreira Torres, Cristina Nita-Rotaru, and Radu State. 2022. A flash(bot) in the pan: measuring maximal extractable value in private pools. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3517745.3561448

[40] Xiangfan Wu, Ju Xing, and Xiaoqi Li. 2025. Exploring Vulnerabilities and Concerns in Solana Smart Contracts. *arXiv preprint arXiv:2504.07419* (2025).

[41] Sebastian Wunderlich. 2023. Current State of MEV in the Ethereum Ecosystem. In *Konferenzband zum Scientific Track der Blockchain Autumn School 2023*. Hochschule Mittweida, 78–84.

[42] Sen Yang, Fan Zhang, Ken Huang, Xi Chen, Youwei Yang, and Feng Zhu. 2023. SoK: MEV Countermeasures: Theory and Practice. arXiv:2212.05111 [cs.CR] https://arxiv.org/abs/2212.05111

[43] YCharts. 2025. Ethereum Chain Full Sync Data Size (Daily) - Historical Data. https://ycharts.com/indicators/ethereum_chain_full_sync_data_size. Accessed: 2025-09-12.

[44] Liyi Zhou, Kaihua Qin, and Arthur Gervais. 2021. A2MM: Mitigating Frontrunning, Transaction Reordering and Consensus Instability in Decentralized Exchanges. arXiv:2106.07371 [cs.CR] https://arxiv.org/abs/2106.07371

[45] Liyi Zhou, Kaihua Qin, Christof Ferreira Torres, Duc V Le, and Arthur Gervais. 2020. High-Frequency Trading on Decentralized On-Chain Exchanges. arXiv:2009.14021 [cs.CR] https://arxiv.org/abs/2009.14021

[46] Patrick Züst, Tejaswi Nadahalli, Ye Wang Prof, and Dr. Roger Wattenhofer. 2021. Analyzing and Preventing Sandwich Attacks in Ethereum. https://api.semanticscholar.org/CorpusID:236193518

[47] Patrick Züst, Tejaswi Nadahalli, and Ye Wang Roger Wattenhofer. 2021. Analyzing and preventing sandwich attacks in ethereum. *ETH Zürich* (2021), 1–29.

A Ethics

During our study, we were conscious of the load our measurements put on Jito's servers, and took steps to rate limit our collection script. As mentioned in Section 3, we pull data from Jito's web API endpoint only once every two minutes for continuous data collection, and keep that same rate when pulling select detailed transaction data. Both the Jito bundle data and the detailed Solana transaction data that we collect is publicly accessible and contains no PII.

B Acknowledgments

We would like to thank our anonymous reviewers and our shepherd for their valuable comments and feedback. This work was partially supported by the National Science Foundation grants 2247307 and 2318290.