# Modeling Large-Scale Manipulation in Open Stock Markets

**Carter Yagemann, Pak Ho Chung, Erkam Uzun, Sai Ragam, Brendan Saltaformaggio, and Wenke Lee** | Georgia Institute of Technology

**This article studies the feasibility of using a botnet to automate stock market manipulation, incorporating data from U.S. Securities and Exchange Commission case files, security surveys of online retail brokerage accounts, and dark web marketplace listings.**

Open markets are susceptible to manipulations that can drive the price of securities (e.g., stocks) up and down, irrespective of their underlying fundamental value. Such behavior creates price volatility and trader uncertainty that can enrich the perpetrator of the manipulation to the detriment of everyone else. In response, most modern securities markets have outlawed well-known techniques for manipulating prices, and criminals have been rooted out by the combined efforts of industry and government entities.

However, in our review of the past decade's U.S. Securities and Exchange Commission's (SEC's) prosecutions, we make the startling observation that most of the detected manipulations were carried out by a lone individual, making it easier to isolate their manipulative orders from background trading activity. When the number of perpetrators increases to two or three, the investigators' success usually hinges on whether links to human relationships (e.g., coworkers and family) can be uncovered. This raises serious security questions about how prepared stakeholders are to handle market manipulations in our increasingly decentralized and automated world, where manipulative orders can originate from thousands of accounts operating in coordination to sow distrust and enrich criminals.

To better understand this threat, we envision an army of retail brokerage accounts, controlled without authorization by a criminal botmaster via devices infected with botnet malware. Such a scenario would be analogous to banking trojan campaigns like Torpig, which was controlling 180,000 infected devices when it was seized by authorities in 2009.[1]

Based on SEC case files, the current security practices of major online retail brokerages, and other data sources, we design a model for this proposed adversary and construct an agent-based simulation capable of evaluating attacks under varying conditions. The results validate our concerns, providing evidence that coordinating a stock market manipulating botnet is trivial and can generate millions of dollars per year in illicit profits—consistent with previous manual manipulations—while being robust to technical factors, like network latency. We also observe that sufficient quantities of stolen brokerage accounts are being traded in online criminal marketplaces to make the attack feasible.

*Digital Object Identifier 10.1109/MSEC.2021.3076717*
*Date of current version: 20 May 2021*

In light of our discoveries, we discuss the open problems and challenges in detecting and preventing large-scale automated market manipulation from the perspectives of government regulators, online brokerages, and exchanges. We also draw parallels to recent events, such as the coordinated efforts of a few million Reddit users to artificially inflate the value of GameStop shares to short squeeze hedge funds,[2] to further illustrate the pertinence of these issues.

## Understanding Market Mechanics

Before diving into the design of our simulation and modeled adversary, it is important to understand the relevant market mechanics[3] and techniques currently used by criminals to manipulate stock prices. Basic trading consists of creating asks (offers to sell a quantity of shares in a stock at a particular price) and bids (offers to buy). A limit order will match offers only at a certain price (or better) whereas market orders immediately match the best available offers. These orders can be set to expire automatically after a certain time (e.g., when the market closes at the end of the day) or remain active until they are filled or explicitly canceled by the trader. Active orders are considered open whereas completed or canceled ones are closed. A trader's positions in a stock is the aggregate of all the shares they currently own.

Canceling is important to understand because it allows traders to create orders purely with the intent of canceling them later. These are referred to as *nonbona fide* orders and they form the cornerstone for layering, which we explain later. Making nonbona fide orders is illegal in the United States, but as canceling is not inherently illegal, the distinction from bona fide orders is a matter of determining the trader's intent. As evidenced by the activities of day traders (professionals who watch the markets and make multiple trades daily) and high-frequency trading, even canceling high volumes of orders is not inherently illegal, which creates an opportunity for abuse.

Another important mechanism is margin trading, which consists of margin buying and short selling (shorting). These are analogous to loans a trader can use to borrow stock shares or cash. In short, if a trader believes a stock's price will decline but does not currently own any shares to sell, he/she can borrow shares from a lender with the promise of returning them later. This lets him/her sell, and then if the price does decline he/she can buy back the shares he/she owes at the lower price, yielding a net profit. Margin buying is the same concept, except with cash. The trader borrows money from a lender, enabling him/her to buy more shares to reap greater gains, assuming the price of the stock rises. Margin trading can enable our modeled adversary to generate larger profits with fewer compromised accounts, and shorting plays a critical role in understanding the recent pumping of GameStop share prices.

### Explaining Pump and Dump

A pump starts by buying or selling shares in a stock to drive the price up or down, respectively. Other traders see this momentum and start placing orders based on the wrong assumption that there is a legitimate reason behind the price movement. The perpetrator then stops pumping and reverses the direction of his orders to profit from the momentum. For example, if he/she was buying to pump up the price, he/she would follow up with asks to profit from the artificially increased price.

Table 1 presents the outcomes of two real-world pump-and-dump manipulations, as reported in SEC prosecution case files. Willner (row 1, Table 1) prepared his scam by placing asks to short shares of the target stock at prices significantly higher than the current market value. He then used a hijacked victim account to place a matching bid at the same high price, causing his ask to execute. The victim now owns the overpriced shares.

**Table 1. The example cases of real-world market manipulation.**

| Case title | Scheme | Start date | Instances | Annual revenue (US$) |
|---|---|---|---|---|
| SEC vs. Joseph P. Willner | Pump | September 2014 | 110 | 350,000 |
| SEC vs. Unknown Traders and JSC PAREX Bank | Pump | December 2005 | 16 | 732,941 |
| SEC vs. Taub et al. | Layering | January 2014 | 23,000 | 13,565,217 |
| SEC vs. Milrud | Layering | January 2013 | More than one | 12,000,000 |
| SEC vs. Briargate Trading, LLC | Layering | October 2011 | 242 | 525,000 |
| SEC vs. Visionary Trading LLC et al. | Layering | May 2008 | More than one | 393,759 |
| SEC vs. Hold Brothers On-Line Investment Services LLC et al. | Layering | January 2009 | 325,000 | 1,028,571 |

Willner then forced the hijacked account to sell the shares back to him at below market value, resulting in a significant profit for Willner and a loss for the victim.

In the Unknown Traders case (row 2, Table 1), the criminals prepared by purchasing shares in the target stock using their own accounts. Meanwhile, they liquidated existing stock shares in hijacked victim accounts into cash. They then used the resulting cash to purchase large volumes of shares in the target stock, causing a surge in the market and pumping up the price. The culprits could then sell their shares at the peak of the price surge, yielding a profit.

### Explaining Layering Manipulation

Layering can be used to raise or lower the price of a target stock by creating the illusion that there are significantly more buyers or sellers in the market, respectively. To start, we will explain how it is used to lower the price. The perpetrator begins by placing asks at prices slightly worse than the current best ask. As his/her open orders accumulate, it creates pressure, i.e., the expectation among other background traders that the price is going to dip. In response, the background traders sell their shares, driving the price down. To maintain pressure, the perpetrator cancels and replaces his/her open orders, always staying slightly worse than the best ask. This drives the price of the stock down further, despite none of the perpetrator's orders actually executing. Once the price is sufficiently deflated, the perpetrator bids at the manipulated price, allowing him/her to obtain shares at an artificial low. He/she then cancels all his/her remaining open orders, waits for the price to revert back to its original value, and then sells his/her shares for a profit.

The reason why applying pressure like this works is because the background traders cannot know that the perpetrator does not actually intend to sell the shares he/she is placing asks for. In fact, they cannot even tell that all the asks belong to one person due to how the orders are routed to the exchange via the brokers. We elaborate on this relationship and the challenges it presents in our discussion of open problems.

All of the listed cases (rows 3–7, Table 1) started with layering to deflate the target stock's price, using the steps we just described. After they sold their acquired shares, they then repeated the manipulation in the opposite direction, starting with a series of nonbona fide buy orders to artificially inflate the price, followed by bona fide sells and cancels. Here pressure is being used in the same way. By creating the illusion that there are pending buyers, background traders are manipulated into thinking that the price will rise, provoking them to buy their own shares to get ahead of the curve.

## Designing a Stock-Trading Botnet

Based on our observations, we create a fully functional proof-of-concept malware, focusing on the scenario where a criminal wants to manipulate a stock market using a botnet of infected victims. We consider a malware that behaves similarly to banking trojans (e.g., Zeus and GameOver) by infecting a victim's web browser. The way these infections occur (e.g., phishing or software exploitation) is already studied in prior work.[4]

### Hijacking Accounts at Scale

We focus on the brokerage defenses that prevent or create awareness of account intrusions. We collect our data from three of the six most popular online brokerages in the United States. Our analysis includes the availability of 2FA and default settings for event alerts, namely, when they are triggered and where they are delivered. Our findings are summarized in Table 2.

By default, the three online brokerages do not require 2FA. Other work has shown that users are unlikely to take the initiative to set up 2FA if it is not mandatory,[5] lowering the bar for attacks like phishing. All three brokerages support software TOTP. Brokerages A and B also support TOTP via a dedicated hardware token, whereas C uses SMS or phone calls. Although hardware TOTP is regarded as more secure than its software alternative, SMS and phone calls are weaker due to threats like SIM swapping, which can allow an adversary to intercept login codes. All of these 2FA schemes can be phished or intercepted, which works in the botmaster's favor.

By default, all three online brokerages generate alerts when security settings are modified and when orders are created or their status changes (e.g., filled or canceled). A summary is also generated at the end of each day when orders occurred. Users can disable the per-order notifications to avoid being bombarded during frequent trading.

By default, all three brokerages deliver their alerts via email only. Emails can be silently deleted by the malware using server-side filter rules that can be inserted

**Table 2. A comparison of brokerage security features.**

| | TOTP 2FA | | | Alerts | |
|---|---|---|---|---|---|
| | Software | Hardware | SMS | Email | Mobile |
| A | ◐ | ◐ | — | ● | ◑ |
| B | ● | ◐ | — | ● | ◑ |
| C | ● | — | ● | ● | ◑ |

*TOTP ◐ = Requires a call, an alert ◑ = Disabled by default* by infecting the victim's browser on any device. Conversely, the brokerages also support mobile notifications; however, they are disabled by default. Were they to be enabled, they could pose an additional challenge because then the malware would have to specifically infect the victim's mobile device (as opposed to any device) to silence them; however, this is not the case by default.

### Malware Architecture

Our proof-of-concept malware is able to 1) read and modify any web page visited by the user; 2) record all HTTP(S) header information, including cookies; 3) spawn additional browser sessions to perform arbitrary web requests; 4) add and remove filter rules from popular email services (i.e., Google, Yahoo, and Microsoft); and 5) create trade orders in popular brokerage services (and simulators for testing purposes).

Figure 1 shows the steps in performing the attack. First, the malware adds itself as a certificate authority to the victim's browser. It then spawns a local proxy server and configures the browser to route all traffic through it. The malware silently captures credentials and session cookies as the user navigates sites. Once it has the necessary materials to access the user's email and brokerage accounts, a new browser session is spawned with these data.

Before performing any manipulations, the malware adds filter rules to the victim's email account so that trade notifications will be silently deleted. These rules are very flexible, allowing the malware to delete the brokerage notifications its actions generate while allowing any notifications made by the victim to pass through.

The malware then contacts the botmaster and makes its malicious trades while keeping a transaction log. If the victim accesses their history in the brokerage site, the malware will intercept the response and remove the malicious orders to hide them, presenting the user with a falsified transaction history.

### Leverage and the Availability of Hijacked Accounts

Once our modeled adversary has gained control of the victim's brokerage account, any owned assets can be used to manipulate the market. For our scenario, he/she does not touch any of the securities already held in the account because rapidly liquidating an account's assets is a red flag for brokerage anomaly detectors. However, he/she can use the cash in the account that is readily available for trading. We refer to the combined cash across all the accounts controlled by the adversary as the *trading leverage*.

To estimate how much leverage an adversary can expect to gain per compromised account, we turn to dark web marketplaces for data. Our data was collected over several months in 2016 from the now-dismantled AlphaBay marketplace,[6] which was accessible via the Tor network. We focus on accounts belonging to Charles Schwab, a major U.S. brokerage.

We collected and parsed listings using a website scraper to perform periodic keyword searches. The listings on AlphaBay displayed the price and number of units (i.e., accounts) sold. The criminals priced accounts based on the amount of cash they contained. However, only the approximate cash values were displayed. We use these data as a proxy for how much effort the hacker exerted to hijack accounts. Specifically, we divide the price of the stolen account by the amount of cash it contained to approximate how much cash leverage is gained per dollar spent hacking accounts.

Our data are summarized in Table 3. Between September and December 2016, we found 1,005 sold accounts, priced from US$50 to US$100 per account. They were advertised as having at least US$5,000 in cash and some claimed to contain more than US$100,000, although the upper bound was not provided.

Based on the data, the average leverage is US$660 for every US$1 spent with a minimum of US$100 for US$1 spent. To keep our calculations conservative throughout this work, we use the minimum-observed leverage for an account that was actually sold. This stolen account went for US$50 and contained US$5,000 in cash.

## Building a Stock Market Simulation

Stock markets are extremely difficult to model, and simulations are hard (if not impossible) to

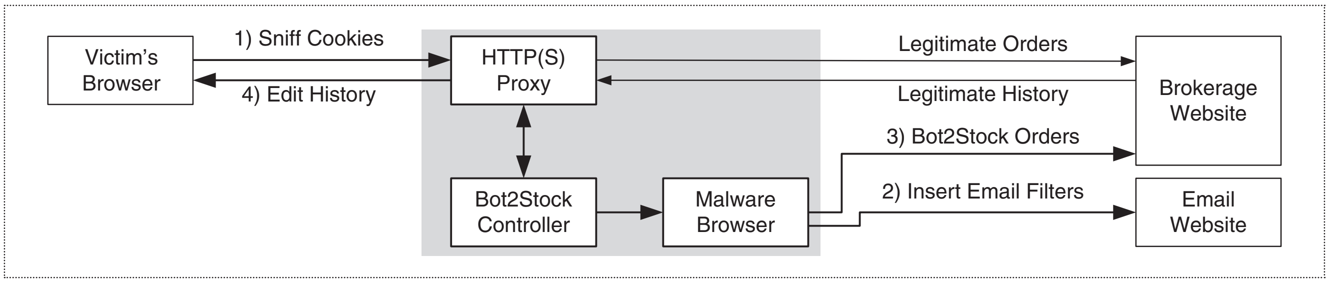

**Figure 1.** The malware architecture. Using a proxy and an Internet Content Adaption Protocol server, it is able to read, modify, and spoof web traffic.

validate for realism. To simulate the attack, we leverage an agent-based discrete-event simulator, the most advanced type of simulation accepted for modeling stock markets.[7–9] It is also used in industry to evaluate multiagent interactions.[10]

Our simulation consists of a collection of background agents that trade a stock based on a mixture of trading strategies. Orders are placed by sending messages to an exchange agent, which maintains an order book for the stock. The messages follow the same protocol used by NASDAQ, which includes allowing agents to query for the latest order stream and current order book spread. Thus, the agents can adjust their strategies in reaction to the current state of the open orders at the exchange. To account for the time it would take a real-world agent to perform its computations, we apply a constant computational delay factor whenever an agent wakes up, along with a latency delay, which is calculated as the sum of a constant minimum latency and a nonnegative random noise factor.

### Modeling Background Agents

We use a combination of zero intelligence (ZI) and heuristic belief learning (HBL) agents to represent the benign traders.[11] Both agents rely on a fundamental belief value for the worth of the stock, which they derive from noisy observations provided by an oracle. At the start of the simulation, these agents enter the market with a Poisson distribution and place their orders based on their trading strategies, which we elaborate on next.

For the agents to create reasonable trades, they each need a fundamental belief of what the stock is worth. To control these fundamental belief values, we use a special agent to act as an oracle. This agent has no computational delay or network latency and when agents query it to update their beliefs, they receive a noisy reading of the fundamental value of the stock at that particular time-step. To avoid adding unnecessary complexity to our results, we use a mean-reverting oracle, which maintains a constant fundamental belief value prior to adding noise for particular observations by the background agents. With this oracle, the background agents will tend to drive the stock price toward the fundamental value in the absence of manipulation.

ZI agents randomly buy and sell shares based on the current price of the stock and their fundamental belief, which they regularly observe from the oracle. More specifically, the decision to buy or sell is picked randomly, and the limit price is a bounded, uniformly random offset from the fundamental belief value. There is also a strategic threshold, which allows the ZI agent to place an order at the current price if it is within a certain threshold of the fundamental belief.

HBL agents start with the same strategy as the ZI agents but also track the stream of recent orders up to a configured memory length. Once enough orders exist to fill the memory, the HBL agents start adjusting the limit prices of their orders based on the transacted and rejected bids and asks. In other words, unlike the ZI agents, these agents are influenced by the order book pressure, which is necessary to model the impact of layering.

**Table 3. The stolen Schwab accounts (16 September 2016–12 December 2016).**

| Selling price (US$) | Minimum cash (US$) | Maximum cash (US$) |
|---|---|---|
| 50 | 5,000 | 20,000 |
| 75 | 20,000 | 100,000 |
| 100 | More than 100,000 | — |
| Accounts sold | (1,005) | — |

### An Adversary's Attack Strategy

The botmaster's strategy is 1) buy shares when the market opens at the best available price, 2) wait for a predetermined attack time, 3) signal the bots (i.e., the hijacked accounts) to begin their manipulation, 4) wait for a predetermined duration of time, 5) sell the previously acquired shares, and then 6) signal the bots to cease manipulation. Note that in a real-world setting, the botmaster would more than likely buy shares slowly over an extended period of time to reduce the risk of being detected, but for simplicity, we reduce the pre-attack setup to a single bulk buy. To keep our findings conservative in light of this simplification, we limit the adversary to one attack per trading day to allow for ample setup time in between. As we show, the attack can be completed in seconds, leaving the majority of each trading day available for setup.

For the bots, they periodically poll the botmaster and wait quietly for the attack signal. When it is raised, the bots begin layering. They periodically poll the exchange to track the order book spread while placing and canceling orders accordingly to maintain open bids that are always slightly worse than the current best bid. For simplicity, we simulate only the bots placing bids to drive the price up, but it is feasible for them to also place asks to drop the price. When they receive the signal to stop, they cancel all the remaining open orders.

## Evaluation

Each experiment consists of several hundred trial pairs for each tested value of the independent variable. The pairs consist of a "control" simulation with no bot agents and a "treatment" simulation with bots. The same random seed is used within each pair so that the background agents will make the same decisions, thus isolating the impact caused by the bots. The dependent variable we measure is the difference in the botmaster's profits with and without the bots.

Each trial simulates 2.5 s of real-world time, with a computational delay of 10 ms and a Poisson distribution function for deciding when the background agents enter the market. The background agents consist of 49 ZI and 16 HBL agents. All of the parameters are chosen to match the 2017 layering evaluation performed by Wang and Wellman.[11]

For the evaluation, we perform two different experiments. In the first, we vary the ratio of bot to background agents and measure the impact on the botmaster's profits. In the second, we steadily increase the latency between the bots, the exchange, and the botmaster, starting with the same latency as the background agents.

## How Profitable Is Manipulation?

Figure 2 shows the impact to the botmaster's return on investment (ROI) of changing the ratio of bots to background agents. As the number of bots increase, the ROI also increases. In the worst-tested case, with a 1.5% ratio of bots to background agents, the ROI is 2.8% over 2.5 s while the bots lose no more than 0.18% of their starting cash from accidental executions.

How does this translate to real-world market environments? In March 2020, IBM had an average, minute-traded volume of 9,120 shares, worth US$1,241,141 at its highest price that month of US$136.10 per share. To sustain 1.5% of this volume for 1 min, the bots would need collectively only US$18,617, which can be achieved with as few as four hijacked accounts, based on our conservative results from the dark web data. If the botmaster invested the same amount of money as each bot (US$18,617), he/she would make 2.8% ROI while the bots collectively lose 0.72% ROI (0.18% per bot), yielding a net ROI of 2.08%.

To calculate a conservative annual ROI for the botmaster, we assume that one attack is carried out per trading day to give the botmaster ample time to perform the preattack setup. Assuming 252 trading days in a year, the botmaster's noncompounding ROI is 252*r*, where *r* is the ROI for a single attack. We use noncompounding ROI to be conservative, although realistically, a criminal is likely to reinvest their earnings, resulting in higher profits. Given our minimum estimated ROI of 2.8%, the botmaster would achieve a 1,022% annual noncompounding ROI. In other words, if the botmaster started with US$100,000, he/she would have US$1,022,000 after a year. This matches the order of magnitude of the real layering manipulations prosecuted by the SEC (Table 1).

The fact that the botmaster's gains outweigh the botnet's loses carries additional interesting implications. For example, if the botmaster were to allow bots to occasionally "win" at his/her expense, the botnet could achieve self-sustainability or even yield a profit for the victims by redistributing the botmaster's ROI. Alternatively, as only four bots are needed per day at one attack per day, for 252 trading days in a year, and we observed more than 1,000 stolen accounts sold in a three-month period, the botmaster could also conduct the campaign without ever reusing a bot, if he/she so desired.

## How Robust Is the Attack?

Figure 3 depicts the impact of network latency on the stability of the attack. Latency is presented relative to the latency of the background agents for one direction of communication, with 0% denoting identical latency. Note that querying the exchange requires a full round trip, so the total additional latency for round-trip time is double. Surprisingly, even with 200% additional one-way latency, the layering remains effective. On closer examination, we discover that because background agents wait for an order confirmation from the exchange before issuing another order and some orders never end up executing, the rate at which orders execute at the exchange remains slow enough to render the additional bot latency moot. For example, when background agents have a 10-ms network latency, trades

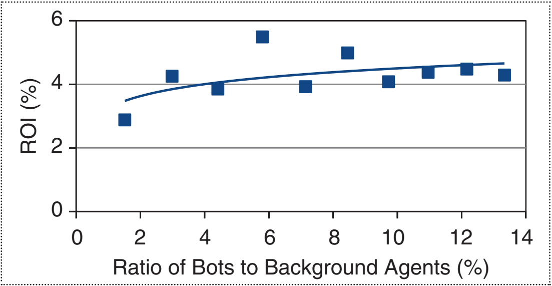

**Figure 2.** The ROI relative to the ratio of bots to background traders.

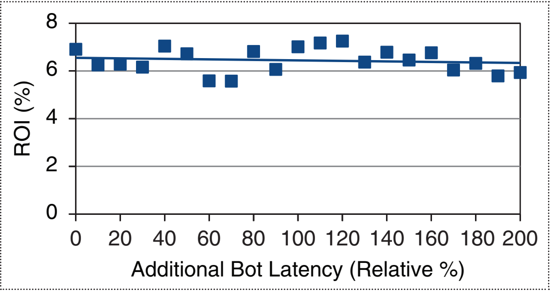

**Figure 3.** The ROI relative to bot network latency.

execute every 30 ms on average. Thus, even if the bots have a 30-ms one-way latency (200% added), they can still keep up with the price movement, preserving the botmaster's and botnet's ROI.

## Open Problems in Remediation

So what can be done to prevent such an attack from occurring in real open markets? We observe that one of the key fundamental challenges in preventing market manipulation stems from each stakeholder having an incomplete picture of trading activity. For example, the finest level of data available to traders is order book layers. Unfortunately, as layers aggregate all the open orders at a given price, the trader does not know if a layer is made up of many tiny orders or a few large ones or even how many traders are involved.

Conversely, for the exchanges and regulators, they can see the orders, but they only know the identity of the broker, not the brokerage or trader for whom the broker is trading. If the auditors want to investigate a trading event, they have to ask the exchange to identify the broker, then ask the broker for the brokerage, and then finally, ask the brokerage for the customer's identity. In the coming years, companies in U.S. stock markets will be required to participate in the Consolidated Audit Trail program, which aims to optimize this process.[12] Deployment is still ongoing as of 2021, and the program raises its own security challenges because it creates a centralized treasure trove of highly sensitive financial data (for example, names, social security numbers, and trading histories). In short, transparency is not a solved problem for U.S. stock markets.

There is also more work to be done in detecting anomalous trading patterns.[13–15] Unfortunately, the existing solutions struggle as the number of coordinating parties increase, which favors our modeled adversary. The transparency challenge also applies to this context. For example, brokerages are in a prime position to detect when a particular account's activity abruptly changes (e.g., trading frequency) but lack a complete context across brokerages to draw correlations from highly distributed, large-scale manipulation. Conversely, exchanges lack the per-account activity hidden behind brokers but have a better picture of the market overall.

### Securing Retail Brokerage Accounts

Although layering attacks may be too quick to stop individually because they take less than a minute to perform, providing customers with better notifications can reduce the risk that an intrusion goes undetected for an extended period. First, our proof-of-concept malware exploits that notifications default to email only, which can be filtered at the server side. Also, sending notifications to a mobile device via SMS or an application would remove this choke point. Second, sending alerts for new logins would make it harder for adversaries to scope out victim accounts prior to committing fraudulent transactions.

It may be tempting to declare that this can all be solved by mandating the use of 2FA, but unfortunately, that would be oversimplifying the problem. Mandating 2FA is already easily within the current capabilities of brokerages, and yet, they choose not to go down this route. One reason is because the financial industry highly values availability, so they are concerned about customers losing their second factor. As one industry expert we interviewed simply stated, customers panic if they cannot access their money. 2FA is also at odds with algorithmic trading, which accounts for more than 85% of U.S. market volume.

### Case Study: WallStreetBets

In January 2021, a Reddit forum called *WallStreetBets* galvanized millions of small retail traders to aggressively buy and hold shares in GameStop, a financially struggling company. The rationale behind this move was to artificially increase share prices, thereby "squeezing" the hedge funds that held large short positions in the stock. If they could keep the price raised long enough, eventually, the short sellers would have to buy shares to cover their short positions, resulting in a huge profit for the Reddit users and potentially bankrupting the hedge funds.

Although this event was not driven by malware and was certainly not designed to be covert, the outcome nevertheless demonstrated the importance of understanding large-scale market manipulation. Similar to our modeled adversary, the Reddit users were able to inflict significant damage on their targets while enriching themselves using orders distributed over millions of retail brokerage accounts. At the time of writing, we have witnessed the stock price move more than 400% in a few days, turning some users into millionaires overnight while short seller losses are estimated in excess of US$5 billion. The aftermath has ignited a heated discussion over the legality of blatant price manipulation and the roles of stakeholders in preventing such volatility in the future. For example, some brokerages enacted restrictions on buying GameStop shares and now face multiple class-action lawsuits as a result. There is also a pending SEC investigation in the context of protecting fair access to open markets. In summary, stakeholders were caught off guard by this event and did not have sufficient mechanisms in place to control it.

Our work provided evidence that large-scale market manipulation is possible using retail brokerage accounts, and such attacks would be highly profitable for criminals. We also highlighted the challenges in detecting and preventing such attacks, such as the current limitations in transparency, the incompatibility between 2FA and algorithmic trading, and the shortcomings of anomaly detectors to detect widely distributed patterns. We hope that our results will serve as motivation for future work in this area. ■

## Acknowledgments

The authors would like to thank Yanick Fratantonio for his contributions during the early stages of this work.

## References

1. B. Stone-Gross et al., "Your botnet is my botnet: Analysis of a botnet takeover," in *Proc. 16th Conf. Comput. Commun. Security*, 2009, pp. 635–647.
2. W. Mullery and M. Pepper. "Inside the Reddit army that's crushing Wall Street," CNN, 2021. https://www.abc57.com/news/inside-the-reddit-army-thats-crushing-wall-street (accessed May 1, 2021).
3. "Trading basics: Understanding the different ways to buy and sell stock." Office of Investor Education and Advocacy, 2021. https://www.sec.gov/investor/alerts/trading101basics.pdf (accessed May 1, 2021).
4. C. Putman and L. Nieuwenhuis, "Business model of a botnet," in *Proc. 26th Euromicro Int. Conf. Parallel, Distrib. Netw.-based Process.*, 2018, pp. 441–445.
5. T. Petsas, G. Tsirantonakis, E. Athanasopoulos, and S. Ioannidis, "Two-factor authentication: Is the world ready?: Quantifying 2FA adoption," in *Proc. 8th Eur. Workshop Syst. Security*, 2015, pp. 1–7.
6. M. Paquet-Clouston, D. Décary-Hétu, and C. Morselli, "Assessing market competition and vendors' size and scope on AlphaBay," *Int. J. Drug Policy*, vol. 54, pp. 87–98, Apr. 2018. doi: 10.1016/j.drugpo.2018.01.003.
7. M. Shearer, G. Rauterberg, and M. Wellman, "An agent-based model of financial benchmark manipulation," in *Proc. ICML-19 Workshop on AI Finance*, 2019, pp. 1–8.
8. J. Li, X. Wang, Y. Lin, A. Sinha, and M. Wellman, "Generating realistic stock market order streams," in *Proc. 34th AAAI Conf. Artif. Intell.*, Feb. 2020, pp. 727–734.
9. D. Byrd, M. Hybinette, and T. Balch, "ABIDES: Towards High-Fidelity Market Simulation for AI Research," 2019, arXiv:1904.12066.
10. S. Assefa, D. Dervovic, M. Mahfouz, T. Balch, P. Reddy, and M. Veloso, *Generating Synthetic Data in Finance: Opportunities, Challenges and Pitfalls*. New York: JPMorgan Chase & Co., 2020.
11. X. Wang and M. Wellman, "Spoofing the limit order book: An agent-based model," in *Proc. 16th Conf. Auton. Agents Multiagent Syst.*, 2017, pp. 651–659.
12. J. Clayton. "Statement on status of the consolidated audit trail," FINRA, New York, 2019. [Online]. Available: https://www.sec.gov/news/public-statement/statement-status-consolidated-audit-trail
13. A. Li, J. Wu, and Z. Liu, "Market manipulation detection based on classification methods," *Proc. Comput. Sci.*, vol. 122, pp. 788–795, Dec. 2017.
14. Y. Cao, Y. Li, S. Coleman, A. Belatreche, and T. McGinnity, "Detecting wash trade in financial market using digraphs and dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2351–2363, 2016. doi: 10.1109/TNNLS.2015.2480959.
15. A. S. Kyle and S. Viswanathan, "How to define illegal price manipulation," *Amer. Econ. Rev.*, vol. 98, no. 2, pp. 274–279, 2008. doi: 10.1257/aer.98.2.274.

**Carter Yagemann** is a Ph.D. student in computer science at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. His research interests include software vulnerability detection, binary analysis, forensics, and machine learning. Yagemann received an M.S. in computer science from Syracuse University, New York. Contact him at yagemann@gatech.edu.

**Pak Ho Chung** is a research scientist at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. His research interests include systems and mobile security. Chung received a Ph.D. from the University of Texas at Austin. Contact him at pchung34@mail.gatech.edu.

**Erkam Uzun** is a Ph.D. student in computer science at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. His research interests include applied cryptography, data privacy, and mobile security. Uzun received an M.S. in computer engineering from TOBB University of Economics and Technology, Ankara, Turkey. Contact him at euzun@gatech.edu.

**Sai Ragam** is an M.S. student in computer science at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. Ragam received a B.Tech from Vellore Institute of Technology, India. His research interests include systems and network security. Contact him at sragam3@gatech.edu.

**Brendan Saltaformaggio** is an assistant professor at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. His research interests include systems security, forensics, and vetting untrusted software. Saltaformaggio received a Ph.D. from Purdue University. Contact him at brendan@ece.gatech.edu.

**Wenke Lee** is a professor and John P. Imlay Jr. Chair in the School of Computer Science at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. His research interests include systems and network security, applied cryptography, and machine learning. Lee received a Ph.D. from Columbia University. Contact him at wenke@cc.gatech.edu.