Issues in Information Systems
Volume 19, Issue 1, pp. 118-127, 2018

SOCIAL MEDIA BOTS AND AUTOMATION: COMPARING AND CONTRASTING TWO SOCIAL MEDIA PLATFORMS' SUSCEPTIBILITY TO BOT ABUSE

Tucker Schell, Washburn University, tucker.schell@washburn.edu
Aaron Spero, Washburn University, aaron.spero@washburn.edu
Benjamin Wolf, Washburn University, benjamin.wolf@washburn.edu
Christopher Ford, Washburn University, christopher.ford@washburn.edu
Yuquan Hong, Washburn University, yuquan.hong@washburn.edu
Wenying Sun, Washburn University, nan.sun@washburn.edu

ABSTRACT

Bots have become prevalent among various social media platforms. Bot activity may appear harmless in nature but can have a significant impact on public opinion in elections and other societal matters. Automation techniques allow bots to dodge security measures and distribute unsolicited information, often to the detriment of users of online social media platforms. In our study, we developed a bot-deployment process and investigated bot deployment, detection, and prevention on two social media platforms: Twitter and Reddit. We uncovered the vulnerabilities of both platforms as well as the limitations they place on social media bots. After evaluating both systems, we propose suggestions for optimal detection and removal of unwanted bot activity.

Keywords: Bots, Social Media, Account Automation, Reddit, Twitter, Computational Propaganda

INTRODUCTION

The growing popularity of social media platforms like Twitter and Reddit over the last decade and a half has been accompanied by a concurrent rise in the number of social media bots that are aimed at aiding, influencing, or informing users with automatically generated content (Ferrara, Varol, Davis, Menczer, & Flammini, 2016). Social media bots have begun to play an increasingly important and visible role in many important online discussions, including ones pertaining to controversial topics and elections. And because of the use of heavily automated social media accounts in the distribution of propaganda during the 2016 U.S. Presidential Election (Bessi & Ferrara, 2016), the prevalence of bots on social media—and their (potentially harmful) influence on popular opinion—has started to become a serious concern for the public. Thus, the reason why we should be interested in social media bots is apparent. But many still have only a nebulous understanding of what exactly bots are and what they do.

So, what are bots? In the context of this discussion, a bot is a scripted software program that automates tasks that might otherwise be performed by humans. Social media bots are a particular type of bot that run over social media networks, often generating messages—for instance, tweets in the case of Twitter—directed at the end-users of social media.

Several different types of bots are frequently deployed on social media. There are chatbots aimed at providing entertaining responses when prompted by a user, bots that provide helpful support functions, and yet others that monitor sites for information, notifying a user when topics or information pertinent to the user appears.

For our study, we were particularly interested in how propaganda bots, spam bots, and other information distribution bots attempt to advocate certain ideas, persons, or objects work. Specifically, we wanted to know how they are deployed, and what the most effective countermeasures are for unwanted automated activity. We developed three research questions that we hoped to be able to find answers to. They are as follows:

1. *What are the limitations of modern social media bots?*

2. *How do the bot detection and prevention measures of several social media platforms compare?*

3. *Are social media platforms doing enough to detect and manage bot abuse? If not, what more can be done?*

The rest of the paper is organized as follows. The literature review section covers prior research that has been done on the topic of social media bots. In the methodology section, we explain the tools, platforms, and methods used in this research. In the results section, we explain what was discovered through our experimentation and exploratory research. The discussion section delves into an examination and analysis of the results we found. Finally, we conclude with limitations of our study and potential avenues for further research.

LITERATURE REVIEW

The majority of research on social media bots has focused on the role of social media bots in social engineering—that is, the role social bot networks have played in attempting to influence society and implement social changes. Some studies have researched potentially positive uses of social media bots. For instance, one study examined how bot networks could possibly be used in information diffusion—or more specifically, in introducing positive memes, fostering public health, and otherwise promoting good behaviors (Mønsted, Sapieżyński, Ferrara, & Lehmann, 2017). However, most of the existing research on social media bots has been concerned with the potentially negative influence social media bots could have on society.

Research on propaganda bots—bots designed to push a specific agenda, individual, or idea, especially within the context of politics—is one of the most frequently-seen topics when it comes to social media bots. Many papers have been released on the influence such bots have had on elections, both domestically and globally. Multiple sources have suggested that propaganda bots played a large role in the 2016 U.S. election (Bessi & Ferrara, 2016; Kollanyi, Howard, & Woolley, 2016). According to one data memo from the Computational Propaganda Project, sponsored by the Oxford Internet Institute, about 1/3rd of the pro-Trump traffic on Twitter occurring during the first U.S. Presidential Debate of 2016 was heavily automated, with 1/5th of the pro-Clinton traffic having been tagged as automated—unsurprisingly, neutral political tweets had the lowest rate of automation (Kollanyi, et al., 2016). Propaganda bots have also reportedly been used to influence political conversations in other nations. The Computational Propaganda Project has investigated bot usage in the elections of more than 9 different countries; results varied, with the US, Ukraine, and Russia all showing high levels of automation employed for political purposes (Woolley & Howard, 2017). Russian Twitter networks, in particular, have been described to be "almost completely bounded by highly automated accounts" (Woolley & Howard, 2017, p. 9). It is estimated that as much as 45% of Twitter activity in Russia, particularly in political contexts, is managed by highly automated accounts (Woolley & Howard, 2017). Clearly, bots are having a greater impact than might be expected in certain parts of the world.

Although much of the research into social media bots has focused on their role in political elections, there are also papers that have covered the impact of bots in debates over more general political issues. For instance, one study measured automation levels in the debate over Great Britain leaving the European Union (Howard & Kollanyi, 2016). The study investigated the number of tweets performed by perceived bots and by accounts employing heavy automation relative to the total number of tweets, finding that roughly 14% of tweets were generated by accounts appearing to employ heavy automation, with a further 0.8% produced by known, disclosed bots (Howard & Kollanyi, 2016).

Given that bots are becoming a growing nuisance for social media sites—and even the wider internet—it is to be expected that a great deal of research has gone into discovering ways to detect and neutralize unwanted bot activity. Discovery methods include Bayesian Bot Detection systems aimed at identifying DNS traffic by suspected bots, systems geared toward detecting malware-based bot activity, and more (Villamarín-Salomón & Brustoloni, 2009; Liu, Chen, Yan, & Zhang, 2008). Limiting the scope of the discussion to purely the detection of bots on social media, several tools like BotOrNot and DeBot have been developed for identifying Twitter-based bot activity (Davis, Varol, Ferrara, Flammini, & Menczer, 2016; Chavoshi, Hamooni, & Mueen, 2016). Some studies have also investigated and published results on the methods best suited for detecting the increasingly "human-like" bots that are being developed to avoid more traditional methods of bot detection. For example, one study investigated how several heuristic measures (including retweet percentage, number of URLs promoted, and length of tweets on Twitter) compare with several machine learning algorithms. Among the tested algorithms were support vector machines, the AdaBoost machine learning algorithm, and the BoostOr method—one of many proposed "optimized" bot detection algorithms

(Morstatter, Wu, Nazer, Carley, & Liu, 2016). It was found that heuristic measures performed poorly compared to AdaBoost or the authors' proposed algorithm, BoostOr (Morstatter et al., 2016).

What we discovered in the course of our literature review was that much of the current body of work on social media bots focuses on bots within the context of individual social media websites, particularly Twitter. Relatively little research had been done on comparing the limitations and the barriers that each social media networking site places as an obstacle for automated activities (such as account creation, message sending, etc.) with the goal of identifying effective practices. We believe research should be done in this area and that doing so could provide valuable data both for those seeking to utilize propaganda bots as well as those looking to identify methods to combat their use.

RESEARCH METHODOLOGY

In order to address our research questions, we conducted thorough research into our target platforms and their application programming interfaces, or APIs (including what the API could and could not be used to accomplish), and also developed a bot deployment process. The bot deployment process was divided into three core stages: the registration of the accounts used to host bot activity, the binding of social media APIs with existing accounts (referred to as bot creation in this study), and the distribution of select information to select individuals by bots without direction from the user (i.e., through scripted actions).

The platforms and resources critical to implementing this process were Twitter, Reddit, the Windows operating system (OS), the FireFox and Tor web browsers, and the Pynput, Tweepy, and PRAW Python modules.

In creating our social media bots, we began by choosing the type of bots to implement and the platform on which to deploy the bots. We identified four primary roles that bots often fulfill (support, spam, propaganda, and bot detection) and among them, we chose to implement a bot that could be categorized as either a spam or propaganda bot for the purposes of distributing messages. Twitter was our first choice for a social media platform because it had a seemingly minimal account registration process (initially no user verification) and an open API for users to interface with. Upon researching other options, we chose Reddit as our second platform because the discussion-based format it uses is very different from Twitter's format; we believed that an investigation of two dissimilar social media platforms might yield results that would apply to a larger percentage of existing social media sites. Another advantage of selecting Reddit is that it, like Twitter, provides an open API, simplifying bot creation and deployment.

Windows OS was used because it is the most common platform for casual users. For web browsing, FireFox and Tor were selected for experimentation. When selecting a programming language, we wanted a language with extensive flexibility that could provide support in the form of importable libraries. Python, an all-purpose scripting language, was chosen because of its large online collection of modules and documentation as well as the ease with which repetitive tasks can be automated, thanks to its nature as a scripting language.

We utilized specialized Python modules to interact with the API and automate API calls. Tweepy was selected as the Python module to be used with the Twitter API. The main appeal of using Tweepy is the adaptation of Twitter endpoints (the API version of a URL) into JavaScript Object Notation (JSON), making the code easy to read and write (Roesslein, 2010). With Reddit, a module called Python Reddit API Wrapper (PRAW) was employed (Boe, 2017). Both modules rely on passing authentication parameters, including API keys, to allow access to the API. API keys are tied to a single account and obtained through a registration process in the bot creation stage that is easier to automate compared to the account registration stage (as barriers like reCAPTCHA, to be discussed later, do not exist).

The first step in in the creation of any social media bot network is to assemble a sufficient number of accounts. Thus, for the first experiment we carried out, our goal was to test how many social media accounts could be created within a limited time frame and compare the differences between the Twitter and Reddit account registration processes.

A Python script was written using Pynput to register for an account on both Twitter and Reddit. After running the account creation script with a register limit of 10 accounts, the number of accounts successfully created was recorded. The trial was performed three times for each social media platform and modifications to the testing process were made to explore less obvious limitations. The experiment was designed to test the detection and prevention measures in place to handle automated account creation and to help identify which techniques are most effective and which should

be replaced. In addition, we were interested in what measures are consistently used throughout social media to discover and stop automated registration.

In the second experiment, we utilized the social media platforms' APIs to send messages to real users of the social media platform in order to test the limitations and effectiveness of API-based social media automation. Although the differences between the messaging formats of Twitter and Reddit lead to differences in bot functionality, the core logic was maintained between the two platforms. The logic consisted of searching for new posts containing the words "online shopping", replying to a post with a link to a (legitimate) survey, and recording information about the user or topic that was replied to (in part to prevent reposts/retweets to the same individual). The experiment was left to run for 72 hours on our two designated test computers, and the data collected included how many unique users were replied to, how many users filled out the survey, and how many bots were reported for spam on each social media platform. This helped us to identify the mechanisms in place for monitoring abuse and the limitations our bot deployment experiment had faced.

RESULTS

Through our experiments, we were able to collect ample results for answering the underlying questions of the study. In the process of experimentation, we also made additional discoveries worth further research and testing. The first experiment, concerning account registration, was the crucial first step in our bot deployment process, and several weeks of time were invested on it. The second experiment, bot deployment, was carried out over a shorter time span (72 hours). However, it still provided results that enabled us to identify and analyze a number of the bot prevention measures used by both platforms.

Account Registration

For account registration, both the Twitter and Reddit platforms made use of IP monitoring, a simple feature that tracks user activity by their IP address and is common among sites that require account registration. Aside from IP monitoring, the two websites have different validation requirements that must be met to successfully create an account.

Twitter's account registration page allowed for our account creation program to walk through the registration process using either Selenium WebDriver, a common scripting module discussed more in depth later, or Pynput controls. The first three account registrations were completed with no change or noticeable difference to the registration process. However, after the third account was registered, all subsequent attempts to create an account from the same IP address were met with a phone number validation screen which can be seen in Figure 1 below.

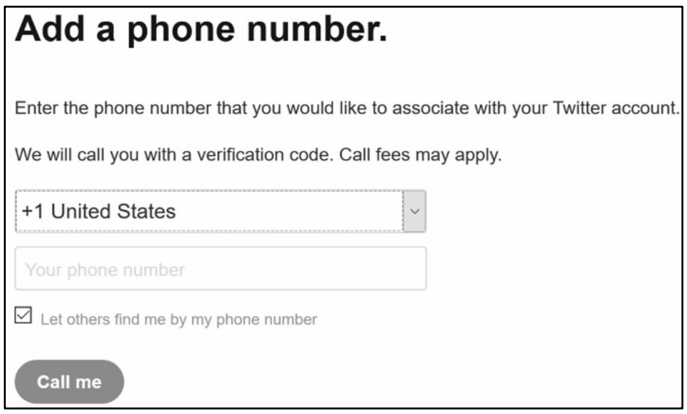

Figure 1. Phone Number Validation Screen (United States)

The validation function calls the number provided by the user and leaves a one-time access code in the form of an audio playback generated by a text-to-speech application. This validation requirement acts as a registration hold that lasts for ten minutes or until an authentic number is provided. The intuitive response to bypass such a validation function is to use phone numbers provided by an online phone service. Phone number generators such as Twilio and

TextNow can generate phone numbers that can be used to send and receive texts or calls. Although these generated phone numbers are also backed by a cellular provider, they explicitly identify their origin to Twitter's verification process and therefore do not qualify as a legitimate account-associated number.

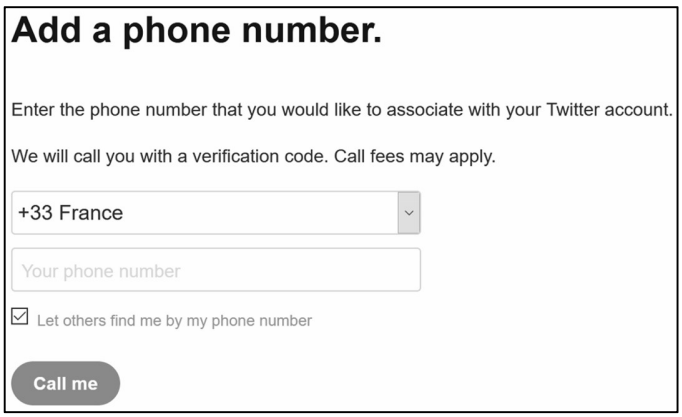

Figure 2. Phone Number Validation Screen (France)

A second possible workaround that was tested was the use of proxy servers and VPN software. However, running our proxy server program, which allowed dynamic manipulation of our IP address, resulted in the same problem we faced when using online phone numbers; the IP addresses pulled from the proxy servers were recognized as illegitimate and flagged accordingly. The VPN software we used, PlexVPN, allowed us to select an authentic IP address from another country and navigate the web with it. Despite that, we found that some websites, including both Twitter and Reddit, monitor the external IP address of the target device which PlexVPN fails to completely conceal. Even with the disguising of the external IP, as was possible using the Tor VPN browser, Twitter still classified foreign IP addresses as suspicious activity and prompted the user for a phone number. Each validation prompt expects a phone number with a country code corresponding to the geographical location associated with the IP address that Twitter observes. In Figure 1, an IP address associated with the United States was detected by Twitter, so a telephone number with a United States country code was expected. Similarly, in Figure 2 above, an IP address associated with France was detected, so Twitter asked for a telephone number from France.
After exhausting the options granted by the resources we had, we began implementing account creation using our working model; that is, create three accounts using Pynput, and then sleep ten minutes before creating each consecutive account thereafter. The automated registration process takes 27 seconds to complete. So, using this system, 139 accounts can be created in a day's time.

Reddit's account registration only allows for the program to register using mouse and keyboard controls from the Pynput module. Traditional bot creation tools such as Selenium cannot be used here for two reasons. First, like many sites, Reddit's biggest bot prevention feature is the reCAPTCHA box embedded into the website's account registration page. The reCAPTCHA software investigates many factors about an online user to determine whether that user is a bot. When a user is flagged as a bot, an audio or visual test is given that requires the reasoning of a human user. Factors that can trigger a test are browsing history, cookies, browsing speed, and other metadata that is less problematic for bypassing reCAPTCHA.

In contrast to Twitter, Reddit places a 10-minute account creation hold immediately following the first account registration by a given IP address. Familiar errors were seen when trying to bypass the IP constraint using proxy servers and VPN software. Reddit also checks the external IP which is unaffected by PlexVPN, and IP addresses from the proxy server were again recognized as illegitimate.

When using the Tor VPN browser in an effort to bypass the IP constraints, it was found that, by default, Tor does not store cookies or browsing history. This lack of cookies and browsing history immediately raised a red flag for reCAPTCHA. However, even if Tor maintained cookies and other digital artifacts, it is unlikely that it would qualify as a suitable browser type by reCAPTCHA's standards because it is a VPN browser, and reCaptcha would flag it accordingly.

After thoroughly researching the resources available for account registration on Reddit, we proceeded with the following creation pattern: first, we created an account using Pynput, and then we had the program sleep ten minutes for each subsequent account. Using this process, each account can be created in 26 seconds and 138 accounts can be created in a day's time. A side-by-side comparison of the programming logic for automatic account registration on both platforms can be seen in Table 1 below.

Table 1. Account Registration Process (Pseudocode)

Twitter	Reddit
-while (botsCreated < botsNeeded) : -if (botsCreated > 3) : -sleep 10 minutes -create user info -open browser -navigate to Twitter sign up -fill in form with user info -submit -print credentials to text file -close browser * -open browser * -navigate to Twitter -sign out -close browser -botCreated += 1 *included to skip welcoming tutorial	-open browser -navigate to Reddit -while (botsCreated < botsNeeded) : -if (botCreated > 0) : -sleep 10 minutes -create user info -click sign up -fill in form with user info -click reCAPTCHA checkbox -submit -print credentials to text file -click sign out -botCreated += 1 -close browser

Bot Deployment

We wrote Python programs to create bots using the accounts produced by the registration process described earlier. We then dispatched our bots to both Twitter and Reddit using the keyword 'online shopping' to scout the social media platforms and distribute a survey. Aiding in the distribution of a survey allowed us to track both link clicks and survey completions and thus gauge the effectiveness of our bots and how well they were received by the respective social media communities.

With Twitter, we dispatched two bots to assist with posting the survey. Previous testing revealed that the string "online shopping" returned about four tweets per minute. Since Twitter imposes a limit on all users of 50 posts every 30 minutes, using two bots would get around this limitation. A five-minute test run of our bot script was performed. The test run streamed incoming tweets that included the term 'online shopping' and replied to them with the link to the survey. A total of 18 tweets were replied to, and one user reacted to our bot's post by completing the survey; she also retweeted the survey to her followers. Perceiving this test as a success, we prepared to launch the bots for the 72-hour test. Two hours later, the bot script was executed again and a total of 20 replies were made before the test failed. One bot became unable to post after two minutes while the other bot lasted 10 minutes. Upon trying to restart the bot script, we received an error message from the Twitter API stating, "Application cannot perform write action," with a link to Twitter's API Policy Support page. It is unclear whether our accounts were flagged as spam by Twitter's automated monitoring system or by Twitter's users. Although the bots lost their ability to perform write actions, read actions such as searching for tweets were still available. However, the result was that the bots could no longer function as intended.

For our Reddit implementation, we tried deploying several bots from several accounts and came across an interesting challenge. Reddit imposes a limit on new users with a scorecard known as 'karma'. The higher the karma points, the more freedom the user has with making public posts. Reddit defaults new users to one karma point, which is enough to let the user create a new post or comment once every 10 minutes. With this in mind, we attempted to have multiple accounts, each with one bot associated with it, running simultaneously. The main issue that arose during testing is that the posting limit Reddit imposes on new accounts is based on the IP address and location from which the accounts

post from. In other words, Reddit will not allow an IP address to post more than once every 10 minutes if the accounts have low karma and share the same IP. However, high karma accounts or multiple low karma accounts on different IP addresses are unaffected by this automated moderation.

Reddit was more successful than Twitter in terms of overall survey responses. From Reddit we received 13 survey responses over a time span of 72 hours. Although the survey response rate was not very high (in part because we wanted to limit spam volume and thus used a single computer and a couple of automated accounts), the posts themselves elicited a number of comments from users who saw them. What's even more interesting is that the commenters often treated the bots as authentic users, responding with comments like: "Uh, who are you? What is this survey for? How long is it?". Although users expressed skepticism toward the link we provided or skepticism toward whether the link was posted in the appropriate subreddit, skepticism toward the account being automated was, interestingly enough, somewhat uncommon. Overall, the posts made by our bots were downvoted significantly, leaving our primary bot-using account with -31 karma—and ultimately causing us to be banned for 3 days from doing anything other than reading public content. Because of this, our bots were unable to perform their script and were rendered useless for the remainder of the experiment.

Table 2 gives an abridged version of the code used during the bot deployment process for both Twitter and Reddit.

Table 2. Bot Deployment Process (Pseudocode)

Twitter	Reddit
-authenticate -continually monitor tweet stream for search term: -choose random bot account to post with -reply to tweet with predefined reply and survey link	-authenticate -continually monitor the Reddit comment stream: -if a comment contains keywords and author is not in an 'already replied to' file: -post predefined reply and survey link -wait 10 minutes

DISCUSSION

After collecting results from our experiments, we are able to address our initial inquiry. First, let us consider the question of, "*What are the limitations of modern social media bots?"*. From our second experiment, it was found that programming with Twitter's or Reddit's API allows for all the functionality of a regular user but with the power and speed of a CPU. Moreover, it is very easy to access information on users or posts we wish to target; a program that is more sophisticated than the one we used might catch a post containing a keyword of interest, investigate the commenting history of the author of that post, and then decide whether it is appropriate to send a designated message—or decide which message would resonate best based on that past commenting history. There are limitations on bots created with the API, however, such as rate limits for using the API and the enforcement of social media rules and policies. With both social media platforms, our bots were limited by the social media organizations disabling or banning the experimental accounts from posting. The cause behind the ban varied from reports by users, moderation by leaders of a community (in Reddit's case), or policy violation and automated detection (for Twitter).

In addressing the question of, "*How do the bot detection and prevention measures of several social media platforms compare?"*, it is easiest to start with the results of the first experiment, as most bot networks will need multiple accounts to function and automated account registration is one good way to acquire those accounts. Twitter and Reddit have contrasting prevention techniques that, although sufficient for stopping smaller penetration testers like ourselves, could be bypassed with more computer power. Twitter includes phone verification and IP monitoring in their registration process. Using phone number validation is a very strong preventative measure as mass generation of qualifying phone numbers is very expensive. However, the phone verification is only as effective as the IP monitoring system is. Although it would require special software or equipment, one could manipulate the external IP address after every three accounts created to continuously generate accounts with no delay.

A weakness of Twitter's bot prevention scheme is that, without reCAPTCHA, it remains completely vulnerable to web scraping programs or programs written using modules like Selenium WebDriver. Selenium, an open-source solution for test automation, is most commonly used as a tool for quickly running repetitive tests against web-based applications as seen in regression testing. Web scraping programs function on the basis that they can pull elements from the HTML DOM, or web page source code, and manipulate those elements however they please. ReCAPTCHA resists these programs by presenting an invisible check box that is hidden from the DOM using Javascript and must be clicked by the user. ReCAPTCHA also checks the browser type of the user. Thus, even if Selenium could find the element, it is identified as a support browser, so the test would still fail even after clicking the reCAPTCHA checkbox.

In contrast to Twitter, Reddit implements reCAPTCHA, making it immune to web scraping attacks. Like Twitter, Reddit also includes IP monitoring for account registration. From experimentation, we found that the combination of IP monitoring and reCAPTCHA is very strong because many of the tools used to bypass reCAPTCHA fail to bypass IP monitoring and vice versa. Therefore, with the use of IP monitoring, both Twitter and Reddit prevent small-scale bot scripters from abusing their systems, but against special networking equipment capable of disguising the external IP, Reddit has the upper hand. This is because, although both platforms' prevention methods can be bypassed with enough effort, it is significantly more time intensive to write scripts using Pynput than it is to write the traditional web scraping program (i.e., using Selenium) that are not restricted on Twitter.
One similarity between the social media platforms we examined is that they both rely on user reports to detect malicious activity. In the case of unsolicited material, a user on Twitter can report a message as spam, and a moderator on Reddit can ban potential bots from their subreddit. However, Twitter also monitors API calls for multiple tweets of the same phrasing emanating from a user and halts such activity (Roth, 2018). Meanwhile, Reddit seems to rely on karma (and specifically, bad karma) as an additional tool to flag activity that deserves a closer look.

As for the question of, "*Are social media platforms doing enough to detect and manage bot abuse?*", we discovered sufficient prevention methods are in place, but improvements are possible. One suggestion we came up with to further reduce unwanted bot activity is the periodic deletion of inactive accounts. Accounts formerly used by real users look more trustworthy, may have a valid phone number on Twitter, and will have more permissions on Reddit. Deleting inactive accounts would be particularly useful with Reddit, since new accounts are limited in what they can do, therefore increasing the demand for old accounts. With periodic deletion of accounts, there would be little risk of artificial "aging" of accounts, where accounts are activated, engage in limited activity, and then are left dormant until needed by bot networks.

With regards to account registration, we suggest that social media platforms implement one of the following three tiers of bot prevention. At the very least, social media platforms should make use of IP monitoring to stop users that are utilizing simple web scraping programs from creating multiple accounts from the same IP within a given frame of time. The second tier would consist of fortifying the IP monitoring with either reCAPTCHA or phone validation. This will stop users who are employing software such as Tor VPN browser or other VPN alternatives. Finally, the most secure line of defense against automated registration is the combination of IP monitoring with a modified version of reCAPTCHA where users are presented with a visual test upon every registration. Such a test would require human reasoning, something that is much harder to imitate than simply human-like page navigation. Simulation of visual-problem solving is very complicated in parallel with IP monitoring, and overcoming this third tier requires more resources than the large majority of users have access to. The remaining portion of users who can manage such a workaround could then be pursued by a spam prevention team.

The philosophy of allowing anyone access to API keys necessarily leads to automated, programmed bot accounts because of the nature of application programming. Both platforms seem to encourage automated accounts ("Automation Rules," 2017). Even if most bots are harmless, opening the floodgates by having an open API does increase the number of harmful bots. What this potentially says about Twitter and Reddit is that they expect more positive outcomes from allowing anyone to write an application interfacing with their platform than negative outcomes associated with bot abuse.

LIMITATIONS

One factor that limited our experiments was the use of new accounts rather than aged accounts. Since new accounts on Reddit have restrictions on how often they can post, our Reddit bots were severely limited. Another issue is that

we did not use programs that could effectively change or mask our IP addresses. Since both Twitter and Reddit used IP tracking to disallow multiple accounts being created in succession (as well as to throttle the activity levels of newly created accounts, in the case of Reddit), changing our IP address (as well as all other potential user identifiers, like cookies) would allow us to increase the speed at which we created accounts. However, accomplishing this is challenging, as many of the means of circumventing IP tracking also cause sites to become more "suspicious" of accounts and trigger more rigorous checks on the part of reCAPTCHA.

While we used the Standard Twitter API for our Twitter experimentation, there is also the Premium API and the Enterprise API that include more features and allow more API calls (Tornes, 2017). Another limitation we faced on Twitter was that the initial scenario that we envisioned was not feasible in the timeframe we had. Originally, we wanted to measure the effects of an army of bots promoting a fake account, but by the time we figured out how bots and account automation worked, it was too late to enact that scenario.

FUTURE RESEARCH

An important avenue of further research would be measuring the actual effectiveness of propaganda distributed via social media bots. Do propaganda bots really have a meaningful impact on public opinion and what are the rates at which people change their opinion after being exposed to online propaganda? Experiments could be conducted using small-scale political events (like community elections), where sampling the population would be relatively easy.

The process of bot creation is also an area to be explored further. With the use of the Selenium WebDriver module mentioned earlier, the bot creation process could be fully automated, producing spam bots that distribute propaganda to a specific online audience. This experiment would include piecing together the three steps of bot deployment discussed in the methodology section to form a single, all-encompassing bot program.

Another research direction would be to investigate how skilled humans are at detecting automated accounts. There has been significant research into the algorithmic detection of bots and bot networks, but what about from the user's perspective? At what rate can humans detect bots? Answering these questions through experimentation could help determine the significance of the social media bot problem that we face today.

REFERENCES

Automation rules. (2017). Retrieved from: https://help.twitter.com/en/rules-and-policies/twitter-automation

Bessi, A., & Ferrara, E. (2016). Social bots distort the 2016 US Presidential election online discussion. *First Monday*, *21*(11). DOI: http://dx.doi.org/10.5210/fm.v21i11.7090

Boe, B. (2017). PRAW: The Python Reddit API Wrapper. Retrieved from: https://praw.readthedocs.io/en/latest/index.html

Chavoshi, N., Hamooni, H., & Mueen, A. (2016). DeBot: Twitter Bot Detection via Warped Correlation. *ICDM*, 817-822. DOI: 10.1109/ICDM.2016.0096

Davis, C. A., Varol, O., Ferrara, E., Flammini, A., & Menczer, F. (2016). Botornot: A system to evaluate social bots. *Proceedings of the 25th International Conference Companion on World Wide Web*, 273-274. DOI: 10.1145/2872518.2889302

Ferrara, E., Varol, O., Davis, C., Menczer, F., & Flammini, A. (2016). The rise of social bots. *Communications of the ACM*, *59*(7), 96-104.

Howard, P. N., & Kollanyi, B. (2016). Bots, #strongerin, and #brexit: Computational propaganda during the uk-eu referendum. *Comprop research note, 2016-1.* DOI: 10.2139/ssrn.2798311

Kollanyi, B., Howard, P. N., & Woolley, S. C. (2016). Bots and automation over Twitter during the first US

Presidential debate. *Comprop data memo, 2016-1*. Retrieved from: http://blogs.oii.ox.ac.uk/politicalbots/wp-content/uploads/sites/89/2016/10/Data-Memo-First-Presidential-Debate.pdf

Liu, L., Chen, S., Yan, G., & Zhang, Z. (2008). Bottracer: Execution-based bot-like malware detection. *International Conference on Information Security*, 97-113. DOI: 10.1007/978-3-540-85886-7_7

Mønsted, B., Sapieżyński, P., Ferrara, E., & Lehmann, S. (2017). Evidence of complex contagion of information in social media: An experiment using Twitter bots. *PloS one*, *12*(9). DOI: 10.1371/journal.pone.0184148

Morstatter, F., Wu, L., Nazer, T. H., Carley, K. M., & Liu, H. (2016, August). A new approach to bot detection: striking the balance between precision and recall. *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 533-540. DOI: 10.1109/ASONAM.2016.7752287

Reddit API Documentation. (2018). *Reddit.* Retrieved from: https://www.reddit.com/dev/api/

Roesslein, J. (2010). Tweepy Documentation. Retrieved from: https://tweepy.readthedocs.io/en/v3.5.0/

Roth, Y. (2018, February 21). Automation and the use of multiple accounts. Retrieved from: https://blog.twitter.com/developer/en_us/topics/tips/2018/automation-and-the-use-of-multiple-accounts.html

Selenium Webdriver. (2018). *Selenium HQ.* Retrieved from: https://www.seleniumhq.org/docs/

Sivakorn, S., Polakis, J., & Keromytis, A. D. (2016). I'm not a human: Breaking the Google reCAPTCHA. *Black Hat*. Retrieved from: https://www.blackhat.com/docs/asia-16/materials/asia-16-Sivakorn-Im-Not-a-Human-Breaking-the-Google-reCAPTCHA-wp.pdf

Tornes, A. (2017, November 14). Introducing Twitter premium APIs. Retrieved from: https://blog.twitter.com/developer/en_us/topics/tools/2017/introducing-twitter-premium-apis.html

Villamarín-Salomón, R., & Brustoloni, J. C. (2009). Bayesian bot detection based on DNS traffic similarity. *Proceedings of the 2009 ACM symposium on Applied Computing*, 2035-2041. DOI: 10.1145/1529282.1529734

Woolley, S. C., & Howard, P. N. (2017). Computational propaganda worldwide: Executive summary. *Computational Propaganda Research Project*. Retrieved from: http://comprop.oii.ox.ac.uk/wp-content/uploads/sites/89/2017/06/Casestudies-ExecutiveSummary.pdf