Measuring and Mitigating OAuth Access Token Abuse by Collusion Networks

Shehroze Farooqi
The University of Iowa

Fareed Zaffar
Lahore University of Management Sciences

Nektarios Leontiadis
Facebook

Zubair Shafiq
The University of Iowa

ABSTRACT

We uncover a thriving ecosystem of large-scale reputation manipulation services on Facebook that leverage the principle of collusion. *Collusion networks* collect OAuth access tokens from colluding members and abuse them to provide fake likes or comments to their members. We carry out a comprehensive measurement study to understand how these collusion networks exploit popular third-party Facebook applications with weak security settings to retrieve OAuth access tokens. We infiltrate popular collusion networks using honeypots and identify more than one million colluding Facebook accounts by "milking" these collusion networks. We disclose our findings to Facebook and collaborate with them to implement a series of countermeasures that mitigate OAuth access token abuse without sacrificing application platform usability for third-party developers. These countermeasures remained in place until April 2017, after which Facebook implemented a set of unrelated changes in its infrastructure to counter collusion networks. We are the first to report and effectively mitigate large-scale OAuth access token abuse in the wild.

CCS CONCEPTS

• **Security and privacy** → *Authentication*; *Authorization*; *Social network security and privacy*;

KEYWORDS

Access Tokens; Collusion Networks; OAuth; Online Social Networks

ACM Reference Format:
Shehroze Farooqi, Fareed Zaffar, Nektarios Leontiadis, and Zubair Shafiq. 2017. Measuring and Mitigating OAuth Access Token Abuse by Collusion Networks. In *Proceedings of IMC '17, London, United Kingdom, November 1–3, 2017,* 14 pages.
https://doi.org/10.1145/3131365.3131404

IMC '17, November 1–3, 2017, London, United Kingdom

ACM ISBN 978-1-4503-5118-8/17/11…$15.00
https://doi.org/10.1145/3131365.3131404

1 INTRODUCTION

Online social networks have become the primary way people connect and communicate with each other, consume information, and form opinions. Reputation is a fundamental tenet of online social networks. People trust the information that is posted by a reputable social media account or is endorsed (e.g., liked) by a large number of accounts. Unfortunately, reputation fraud is prevalent in online social networks. A number of black-hat reputation manipulation services target popular online social networks such as Facebook and Twitter [29, 53, 63]. These services rely on a large number of online social network accounts to conduct reputation manipulation at scale. To accomplish this goal, fraudsters purchase fake accounts in bulk from underground marketplaces [55, 57], use infected accounts compromised by malware [51], or recruit users to join collusion networks [58].

Online social networks try to counter reputation manipulation activities on their platforms by removing fake likes and suspending suspicious accounts. Prior research on detecting reputation manipulation activities in online social networks can be broadly divided into two categories: *(a)* identifying temporally synchronized manipulative activity patterns [24, 28, 40]; *(b)* identifying individual accounts suspected to be involved in manipulative activity based on their social graph characteristics [26, 58, 64, 65]. Recent studies have shown that fraudsters can circumvent these detection methods by incorporating "normal" behavior in their activity patterns [29, 50, 59]. Defending against fraudulent reputation manipulation is an ongoing arms race between fraudsters and social network operators. According to their own published estimates, Facebook and Twitter currently have tens of millions of "false" accounts [5, 21]. Therefore, the issue of fraudulent reputation manipulation remains an ongoing challenge for social network operators.

In this paper, we uncover a thriving ecosystem of reputation manipulation services on Facebook that leverage the principle of *collusion*. In these collusion networks, members like other members' posts and in return receive likes on their own posts. Such collusion networks of significant size enable members to receive a large number of likes from other members, making them appear much more popular than they actually are. As expected, colluding accounts are hard to detect because they mix real and fake activity. Our goal in this paper is to understand their methods of coordination and execution to develop effective and long-lasting countermeasures.

OAuth Access Token Leakage. To understand the extent of the problem collusion networks pose, we start by analyzing popular

Facebook collusion networks. We find that collusion networks conduct reputation manipulation activities by exploiting popular third-party Facebook applications with weak security settings. Collusion networks collect OAuth access tokens for applications, which utilize the *implicit* mode in OAuth 2.0 [30], with help from colluding members. These access tokens are then used to conduct activities on behalf of these applications and colluding accounts. Using a large pool of access tokens, collusion networks provide likes and comments to their members on an on-demand basis. We find that popular collusion networks currently exploit a few popular Facebook applications. However, our analysis of top 100 Facebook applications reveals that more than half of them are susceptible to access token leakage and abuse by collusion networks. While prior research has reported several security weaknesses in OAuth and its implementations [54, 61, 66], we are the first to report large-scale OAuth access token leakage and abuse. Since OAuth 2.0 is also used by many other large service providers, their implementation may also be susceptible to similar access token leakage and abuse.

Milking Collusion Networks Using Honeypots. We deploy honeypots to conduct a large-scale measurement study of popular Facebook collusion networks. Specifically, we create honeypot Facebook accounts, join collusion networks, and "milk" them by requesting likes and comments on posts of our honeypot accounts. We then monitor and analyze our honeypots to understand the strategies used by collusion networks to manipulate reputation. We identify more than one million unique colluding accounts by milking collusion networks. As part of the milking process, we submit more than 11K posts to collusion networks and receive a total of more than 2.7 million likes. We identify the membership size of collusion networks by tracking the number of unique accounts that like the posts of our honeypot accounts. Our membership estimate of these collusion networks are up to 295K for hublaa.me followed by 233K for official-liker.net in the second place. The short URLs used by collusion networks to retrieve access tokens have more than 289 million clicks to date. Our analysis of short URLs shows that popular collusion networks are daily used by hundreds of thousands of members. Collusion networks monetize their services by displaying advertisements on their heavily visited websites and offering premium reputation manipulation plans.

Countermeasures. We disclose our findings to Facebook and work with them to mitigate these collusion-based reputation manipulation services. While we identify a wide range of possible countermeasures, we decide to implement the countermeasures that provide a suitable tradeoff between detection of access token abuse and application platform usability for third-party developers. For instance, we do not block the third-party applications exploited by collusion networks because it will negatively impact their millions of legitimate users. We do not disallow OAuth implicit mode, which is optimized for browser-based applications, because it will burden third-party developers with prohibitive costs associated with server-side application management. As part of countermeasures, we first introduce rate limits to mitigate access token abuse but collusion networks quickly adapt their activities to avoid these rate limits. We then start invalidating access tokens that are milked as part of our honeypot experiments to mitigate access token abuse by collusion networks. We further rate limit and blacklist the IP addresses and autonomous systems (ASes) used by collusion networks to completely cease their operations.[1]

Key Contributions. Our key contributions are summarized as follows.

- We are the first to report large-scale OAuth access token leakage in the wild. We demonstrate how collusion networks exploit popular third-party Facebook applications with weak security settings to retrieve OAuth access tokens and abuse them for reputation manipulation.
- We deploy honeypots to milk collusion networks, identify colluding accounts, and study the scale of their operation. We identify more than one million unique colluding accounts.
- We disclose our findings to Facebook and investigate countermeasures to detect abuse of leaked OAuth access tokens. Our countermeasures are able to cease collusion network operations without requiring any modifications to the OAuth framework.

Paper Organization: The rest of this paper is organized as follows. In Section 2, we discuss how attackers exploit security weaknesses in third-party Facebook applications for leaking and abusing OAuth access tokens. Section 3 discusses the reputation manipulation operations of collusion networks. In Section 4, we describe our honeypot deployment to measure the activities collusion networks. Section 5 sheds light on the monetization strategies and ownership of collusion networks. Section 6 discusses the design and deployment of our countermeasures. We review the related work in Section 7 and conclude in Section 8.

2 OAUTH ACCESS TOKEN ABUSE

In this section, we first provide a background of Facebook's third-party application ecosystem and then discuss how attackers can exploit these applications to abuse their OAuth access tokens.

2.1 Background

All major online social networks provide social integration APIs. These APIs are used for third-party application development such as games, entertainment, education, utilities, etc. These applications acquire read/write permissions from the social network to implement their functionalities. Popular social network applications have tens of millions of active users and routinely conduct read/write operations on behalf of their users.

Facebook also provides a development platform for third-party applications. Facebook implements OAuth 2.0 authorization framework [30] which allows third-party applications to gain restricted access to users' accounts without sharing authentication credentials (i.e., username and password). When a user authenticates an application using OAuth 2.0, an *access token* is generated. This access token is an opaque string that uniquely identifies a user and represents a specific *permission scope* granted to the application to perform read/write actions on behalf of the user. A permission

[1]The discussed countermeasures remained in place until April 2017, after which Facebook implemented a set of unrelated changes in its infrastructure [11] to counter collusion networks.

scope is a set of permissions requested by the application to perform actions on behalf of the user.

There are two types of permissions that an application may request. The first type of basic permissions do not require Facebook's approval. They include access to profile information, email addresses, and friend lists. The second type of sensitive permissions (e.g., publish_actions) require Facebook's approval [7]. These permissions allow third-party applications to conduct certain actions on behalf of a user, e.g., posting status updates, generating likes and comments.

Access tokens are invalidated after a fixed *expiration duration*. They can be categorized as short-term or long-term based on their expiration duration. Facebook issues short-term access tokens with 1 to 2 hours expiration duration and long-term access tokens with approximately 2 months expiration duration.

OAuth 2.0 [30] provides two workflows to generate an access token: client-side flow (also referred to as *implicit mode*) and server-side flow (also referred to as *authorization code mode*).[2] Both workflows are similar with few changes in request parameters and some additional steps in the server-side flow. Figure 1 illustrates the OAuth 2.0 workflow of a Facebook application to generate an access token for client-side and server-side authorization.

- The flow is initiated by directing a user to Facebook's authorization server by clicking on a login button. The request to the authorization server includes application ID, redirection URI, response type, and a permission scope. The application ID is a unique identifier assigned to every Facebook application. The redirection URI is configured in the application settings. The response type is set as 'token' to return access token in a client-side flow and is set as 'code' to return an authorization code in a server-side flow.
- Facebook's authorization server validates the request and prompts the user to authorize the application and grant permissions in the browser. User authorizes the application and grants the requested permissions to the application.
- Facebook redirects the user to the redirection URI along with an access token or an authorization code in the URL fragment. For the client-side flow, an access token is returned in response which is retrieved and stored by the application terminating the client-side flow. For the server-side flow, an authorization code is returned in response and the following additional step is required.
- The authorization code is exchanged for an access token by requesting Facebook's authorization server through the application's server [14]. The request includes application ID, redirection URI, authorization code, and application secret. The request to exchange an authorization code for an access token is authenticated using the application secret.

The access tokens are then used by applications to perform the Facebook Graph API requests on behalf of users. For each request, an application is generally required to pass on application ID, application secret, and the corresponding access token. As we discuss next, the application secret may not be mandatory to make these requests.

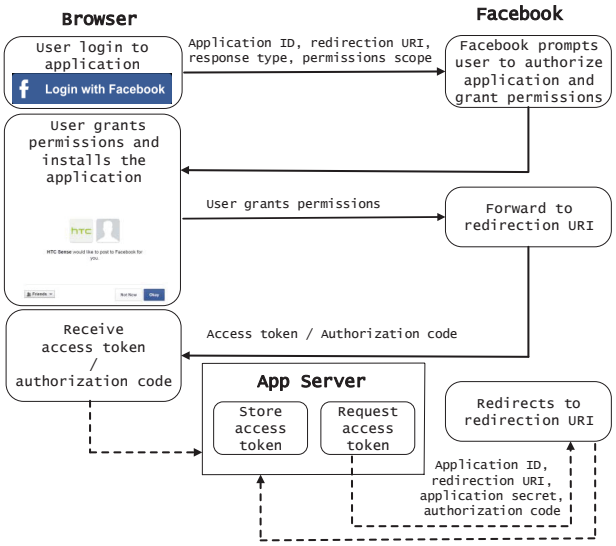


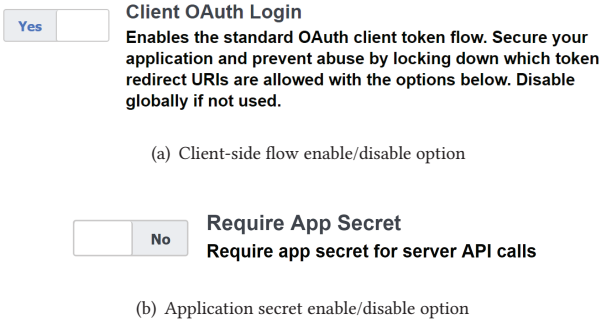
Figure 1: Workflow of Facebook applications

2.2 Identifying Susceptible Applications

Applications select a suitable OAuth flow based on their access token usage scenarios. Server-side flows are by design more secure than client-side flows because they do not expose access tokens at the browser. As shown in Figure 2(a), Facebook provides an option to disable client-side flow from application settings. Facebook recommends third-party applications to disable client-side flow if it is not used [13]. The client-side flow is typically allowed by applications that make Facebook Graph API calls only from the client-side. For example, the client-side flow is used by browser-based applications which cannot include application secret in client-side code. In fact, some client-side applications may not have an application server at all and perform Graph API requests only from the browser using JavaScript. If the application secret is required, as shown in Figure 2(b), applications will have to expose their application secret in the client-side flow. It is noteworthy that application secret is treated like a password and hence it should not be embedded in client-side code.

Prior work has shown that attackers can retrieve access tokens by exploiting security weaknesses in the OAuth protocol and its implementations [16, 32, 49, 54, 61, 66]. Facebook applications that use client-side flow and do not require application secret are susceptible to access token leakage and abuse. For example, attackers can retrieve access tokens in client-side flows by eavesdropping [54], cross-site scripting [49, 54], or social engineering techniques [25, 38]. A leaked access token has serious security and privacy repercussions depending on its authorized resources. Attackers can abuse leaked access tokens to retrieve users' personal information.

[2] In addition to implicit and authorization code modes, OAuth 2.0 supports *resource owner password credentials mode* and *client credentials mode*. The former mode is used by clients to give their credentials (username and password) directly to the applications. The latter mode does not involve any client interaction and is used by applications to access their resources. We do not discuss these modes because they are not used to generate user access tokens.



(a) Client-side flow enable/disable option

(b) Application secret enable/disable option

Figure 2: Security settings of Facebook applications

Application Identifier	Application Name	Monthly Active Users (MAU)
174829003346	Spotify	50 million
100577877361	PlayStation Network	5 million
241284008322	Deezer	5 million
139475280761	Pandora	5 million
193278124048833	HTC Sense	1 million
153996561399852	Flipagram	1 million
226681500790782	TownShip	1 million
137234499712326	Tango	1 million
41158896424	HTC Sense	1 million

Table 1: List of 9 susceptible applications with long-term access tokens among top 100 Facebook applications. These applications can be exploited for reputation manipulation by abusing their access tokens.

Attackers can also abuse leaked access tokens to conduct malicious activities such as spreading spam/malware.

We implement a Facebook application scanning tool to identify applications that are susceptible to access token leakage and abuse. Our tool uses Selenium [19] and Facebook SDK for Python [17] to launch the application's login URL and install the application on a test Facebook account with the full set of permissions. We first infer the OAuth redirection URI used by the application by monitoring redirections during the Facebook login flow. Using the OAuth redirection URI, we install the application on the test Facebook account with the permissions that were originally acquired by the application. If the application is successfully installed, we retrieve the access token at the client-side from the application's login URL. Using the access token, we make an API call to retrieve the public profile information of the test Facebook account and like a test Facebook post. If we are able to successfully conduct these operations, we conclude that the application can be exploited for reputation manipulation using leaked access tokens.

We analyze top 100 third-party Facebook applications [6] using our scanning tool. Our tool has identified 55 susceptible applications, out of which 46 applications are issued short-term access tokens and 9 applications are issued long-term access tokens. Short-term access tokens pose a limited threat because they are required to be refreshed after every 1-2 hours. On the other hand, long-term access tokens provide a 2 month long time window for an attacker. Table 1 lists 9 susceptible applications that are issued long-term access tokens. We report their monthly active users (MAUs) statistics provided by the Facebook Graph API. The highest ranked susceptible application in the list has about 50 million monthly active users. We note that all of these susceptible applications have millions of monthly active users, which can cloak access token abuse by attackers.

3 COLLUSION NETWORKS

A number of reputation manipulation services provide likes and comments to Facebook users based on the principle of collusion: members like other members' posts, and in return receive likes from other members. As discussed earlier, these collusion networks exploit Facebook applications with weak security settings. Collusion networks of significant size can enable members to escalate their reputation, making them appear much more popular than they actually are.

We first survey the landscape of Facebook collusion networks by querying search engines for the relevant keywords, such as 'Facebook AutoLiker', 'Status Liker', and 'Page Liker', that are found on a few well-known collusion network websites. This approach of using seed keywords to find other similar websites via search engines has been used in prior literature [36, 39, 46]. We compile a list of such websites and use Alexa [2] to shortlist popular collusion networks. Table 2 lists top 50 Facebook collusion network websites. It is noteworthy that top 8 collusion networks are ranked within the top 100K. For example, hublaa.me is ranked around 8K and 18% of their visitors are from India, where it is ranked within the top 3K sites. It is interesting to note that other collusion networks also get most of their traffic from countries such as India, Egypt, Turkey, and Vietnam.

We investigate popular collusion networks to understand the features they offer and to identify Facebook applications that they exploit. Collusion networks ask users to install a Facebook application and submit the generated access token in a textbox on their website. The installation link redirects users to a Facebook dialog mentioning the application's name. Table 3 lists the applications used by popular collusion networks along with their statistics retrieved from the Facebook Graph API. Using our tool, we verify that these Facebook applications use client-side flow and do not require application secret for making the Graph API calls. We note that *HTC Sense*, which is used by several popular collusion networks, is ranked at 40 and has on the order of a million daily active users (DAU). *Nokia Account* is ranked at 249 and has approximately a hundred thousand daily active users. Similarly, *Sony Xperia smartphone* is ranked at 886 and has on the order of ten thousand daily active users. It is noteworthy that collusion networks cannot create and use their own applications because they would not pass Facebook's strict manual review process for applications that require write permissions [3]. However, collusion networks can (and do sometimes) switch between existing legitimate applications that are susceptible to access token leakage and abuse.

Workflow of collusion networks. Most collusion networks have a similar web interface and they all provide a fairly similar user experience. Figure 3 illustrates the workflow of Facebook collusion networks.

Collusion Network	Alexa Rank	Top Country	Top Country Visitors
hublaa.me	8K	India	18%
official-liker.net	17K	India	26%
djliker.com	39K	India	55%
autolikesgroups.com	54K	India	30%
myliker.com	55K	India	45%
mg-likers.com	56K	India	50%
4liker.com	81K	India	33%
fb-autolikers.com	99K	India	44%
autolikerfb.com	109K	India	62%
cyberlikes.com	119K	India	78%
postliker.net	132K	India	63%
oneliker.com	136K	India	58%
f8-autoliker.com	136K	India	74%
postlikers.com	148K	India	83%
fblikess.com	150K	India	64%
way2likes.com	154K	India	74%
kdliker.com	154K	India	80%
topautolike.com	192K	India	60%
royaliker.net	201K	India	86%
begeniyor.com	205K	Turkey	85%
autolike-us.com	227K	India	52%
royaliker.net	210K	India	59%
autolike.in	216K	India	74%
likelikego.com	232K	India	52%
myfbliker.com	238K	India	58%
vliker.com	273K	India	43%
likermoo.com	296K	India	62%
f8liker.com	296K	India	80%
facebook-autoliker.com	312K	India	87%
kingliker.com	351K	India	72%
likeslo.net	373K	India	61%
machineliker.com	386K	India	59%
likerty.com	393K	India	60%
monkeyliker.com	410K	India	80%
vipautoliker.com	448K	India	64%
likelo.me	479K	India	16%
loveliker.com	491K	India	59%
autoliker.com	496K	India	56%
likerhub.com	498K	India	69%
monsterlikes.com	509K	India	82%
hacklike.net	514K	Vietnam	57%
rockliker.net	530K	India	92%
likepana.com	545K	India	57%
autolikesub.com	603K	Vietnam	92%
extreamliker.com	687K	India	50%
autolikesub.com	721K	Vietnam	84%
autolike.vn	969K	Vietnam	94%
fast-liker.com	1,208K	-	-
arabfblike.com	1,221K	Egypt	43%
realliker.com	1,379K	-	-

Table 2: List of popular collusion networks in the descending order of their Alexa rank

Application Identifier	Application Name	DAU	DAU Rank	MAU	MAU Rank
41158896424	HTC Sense	1M	40	1M	85
200758583311692	Nokia Account	100K	249	1M	213
104018109673165	Sony Xperia	10K	866	100K	1563

Table 3: Facebook applications used by popular collusion networks.

- A user visits the collusion network's website and clicks on the button to install the application. The website redirects the user to the application authorization dialog URL. The user is asked to grant the requested permissions and install the application.
- The user returns to the collusion network website after installing the application and clicks on the button to retrieve the access token. The website again redirects the user to the Facebook authorization dialog URL with *view-source* appended. The authorization dialog redirects the user to a page that contains the access token as a query string in the URL. The use of *view-source* stops the authorization dialog from further redirections. The user manually copies the access token from the address bar and submits it at a textbox on the collusion network website.
- The collusion network saves the access token and redirects the user to an admin panel, where the user can request likes and comments. Some collusion networks require users to solve CAPTCHAs and/or make users watch ads before allowing them to request likes and comments.

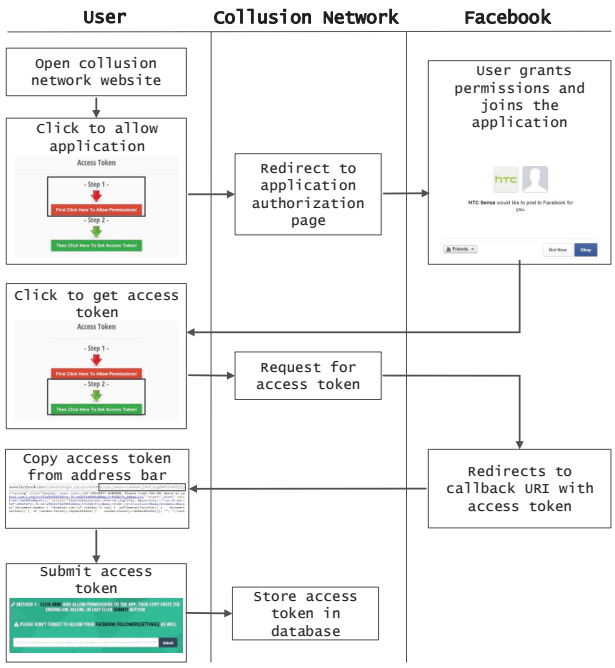
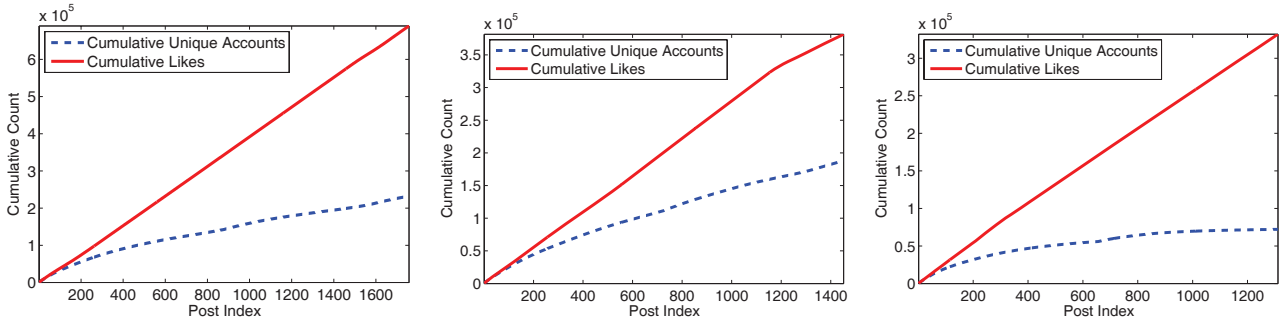


Figure 3: Workflow of Facebook collusion networks

4 MEASURING COLLUSION NETWORKS

Honeypots have proven to be an effective tool to study reputation manipulation in online social networks [29, 42, 52, 62]. The basic principle of honeypots is to bait and deceive fraudsters for surveilling their activities. In order to investigate the operation and scale of Facebook collusion networks, we deploy honeypots to "milk" them.



(a) official-liker.net (b) mg-likers.com (c) f8-autoliker.com

Figure 4: Cumulative distribution of likes and unique accounts. We note that collusion networks provide a fixed number of likes on a post per request. However, the rate of new unique accounts decreases steadily indicating repetition in accounts.

We create new Facebook honeypot accounts and join different collusion networks using the workflow described in Section 3. Our honeypot accounts regularly post status updates and request collusion networks to provide likes/comments on these posts. Soon after the submission of our requests to collusion networks, we notice sudden bursts of likes and comments by a large number of Facebook accounts which are part of the collusion network. Since repeated requests result in likes/comments from many unique Facebook accounts, we can uncover the memberships of collusion networks by making a large number of reputation manipulation requests. Our goal is to estimate the scale of collusion networks by tracking their member Facebook accounts. We also want to understand the tactics used by collusion networks to stay under the radar and avoid detection.

Experimental design. We register 22 new Facebook accounts intended to be used as active honeypots for studying popular collusion networks. Each honeypot account joins a different collusion network in Table 2. In an effort to actively engage collusion networks, our honeypot accounts regularly post status updates on their timelines and request the collusion networks to provide likes/comments on them.[3] It is challenging to fully automate this process because collusion networks employ several tactics to avoid automation. For example, some collusion networks impose fixed or random delays between two successive requests. Many collusion networks redirect users through various redirection services before allowing request submission. Several collusion networks require users to solve a CAPTCHA in order to login and before making each request. To fully automate our honeypots, we use a CAPTCHA solving service [4] for automatically solving CAPTCHAs and Selenium [19] for submitting requests to collusion networks. We continuously post status updates and requests to collusion networks over the duration of approximately three months from November 2015 to February 2016.

Data collection. We regularly crawl the timelines of our honeypot Facebook accounts to log incoming likes and comments provided by collusion networks. The number of unique Facebook accounts who like or comment on a honeypot account is an estimate of the collusion network's size. Note that our membership estimate is strictly a lower bound because we may not observe all collusion network accounts, which are randomly picked from a large pool of access tokens. We also crawl the activity logs of our honeypot accounts to collect outgoing likes and comments.

[3]Our honeypot accounts do not take part in any other activity. Therefore, it is reasonable to assume that all activities involving our honeypot accounts are performed by collusion networks.

4.1 Size of Collusion Networks

Milking Collusion Networks. We post status updates from our honeypot accounts and request collusion networks to provide likes on the posts. Figure 4 plots the cumulative distribution of likes and unique accounts milked by our honeypots for three collusion networks. We observe that the count of new unique accounts steadily declines even though the new like count remains constant. The decline represents diminishing returns due to the increased repetition in users who likes the posts of our honeypot accounts. Specifically, due to the random sampling of users from the database of access tokens, the likelihood of user repetition increases as we post more status updates for a honeypot account. It is important that we milk collusion networks as much as possible to accurately estimate their membership size. While we are able to max out many collusion networks, we face some issues for a few collusion networks. For example, djliker.com and monkeyliker.com impose a daily limit of 10 requests, thus we are not able to fully max out these collusion networks. Moreover, arabfblike.com and a few other collusion networks suffer from intermittent outages when they do not respond to our requests for likes. The set of unique accounts who like posts of our honeypot accounts are collusion network members. Table 4 shows that the membership size of collusion networks vary between 295K for hublaa.me to 834 for fast-liker.com. We note that hublaa.me has the largest membership size at 295K accounts, followed by official-liker.net at 233K and mg-likers.com at 178K. The membership size of all collusion networks in our study sum up to 1,150,782. As we discuss later, some accounts are part of multiple collusion networks. After eliminating these duplicates, the total number of unique accounts across all collusion networks is 1,008,021.

	Incoming Activities			Outgoing Activities			
Collusion Network	Total Number of Posts Submitted	Total Number of Likes	Avg. Number of Likes Per Post	Number of Activities Performed	Number of Target Accounts	Number of Target Pages	Membership Size
hublaa.me	1,421	496,714	350	145	46	47	294,949
official-liker.net	1,757	685,888	390	1,955	846	253	233,161
mg-likers.com	1,537	379,475	247	1,524	911	63	177,665
monkeyliker.com	710	165,479	233	956	356	19	137,048
f8-autoliker.com	1,311	331,923	253	2,542	1,254	118	72,157
djliker.com	471	70,046	149	360	316	23	61,450
autolikesgroups.com	774	202,373	261	1,857	885	189	41,015
4liker.com	269	71,059	264	2,254	1,211	301	23,110
myliker.com	320	32,821	102	1,727	983	33	18,514
kdliker.com	599	82,736	138	1,444	626	79	18,421
oneliker.com	334	24,374	72	956	483	81	18,013
fb-autolikers.com	244	19,552	80	621	397	32	16,234
autolike.vn	139	35,425	254	2,822	1,382	144	14,892
monsterlikes.com	495	72,755	146	2,107	671	39	5,168
postlikers.com	96	8,613	89	2,590	1,543	94	4,656
facebook-autoliker.com	132	4,461	33	2,403	1,757	15	3,108
realliker.com	105	19,673	187	2,362	846	61	2,860
autolikesub.com	286	25,422	88	1,531	717	100	2,379
kingliker.com	107	5,072	47	1,245	587	136	2,243
rockliker.net	99	4,376	44	82	39	1	1,480
arabfblike.com	311	4,548	14	68	31	14	1,328
fast-liker.com	232	10,270	44	1,472	572	102	834
All	11,751	2,753,153	238	33,023	16,459	1,944	1,150,782

Table 4: Statistics of the collected data for all collusion networks

Analyzing Short URLs. Some collusion networks use URL shortening services to redirect users to the Facebook application installation page and retrieve access tokens. URL shortening services such as goo.gl publicly provide information about short URLs such as total clicks, referrers, browsers, platforms, and geolocation. We use this publicly available information to further understand the scale of collusion network operations. Table 5 lists the short URLs that are used by collusion networks. The oldest short URL was created on June 11, 2014, with more than 147 million clicks to date. The creation dates of older short URLs likely represent the launch dates of different collusion networks. To confirm this hypothesis, we further use Google Trends [10] and the Internet Archive's Wayback Machine [12] to analyze popular collusion networks. Our analysis confirms that most of the collusion networks were publicly launched around late 2013 to mid 2014. The top three short URLs redirect users to install the *HTC Sense* application and retrieve access tokens. Their referrer domains are mg-likers.com, djliker.com, and hublaa.me, each of which has hundreds of thousands of members (see Table 4). The top 3 short URLs currently receive around 308K, 139K, and 122K daily clicks. We note that several short URLs point to the same long URL. The sum of click counts for all unique long URLs in Table 5 exceeds 289 million. The click counts of short URLs are reasonable estimates of traffic on collusion network websites. However, these click counts may not give us an accurate estimate of collusion network membership because a user can click on a short URL several times to submit multiple likes/comments requests to the collusion network. The geolocation statistics of short URLs indicate that a vast majority of users are from India, Egypt, Vietnam, Bangladesh, Pakistan, Indonesia, and Algeria. This finding corroborates the geolocation statistics from Alexa in Table 2.

4.2 Collusion Network Activities

Incoming Activities. Table 4 summarizes the statistics of the data collected for different collusion networks using our honeypot accounts. In total, we submit more than 11K posts to collusion networks and garner more than 2.7 million likes. As shown in Figure 4, we observe that status updates typically receive a fixed number of likes per request, ranging between 14-390 across different collusion networks. For example, official-liker.net, f8-autoliker.com, and myliker.com provide approximately 400, 250, and 100 likes per request, respectively.

Outgoing Activities. Collusion networks also use our honeypot accounts to conduct reputation manipulation activities on other Facebook accounts and pages. In total, our honeypot accounts are used by collusion networks to like more than 33K posts of 16K accounts and 2K pages. We observe that some collusion networks use our honeypots more frequently than others. For example, autolike.vn uses our honeypot accounts to provide a maximum of 2.8K likes on posts of 1.3K users and 144 pages.

Analyzing Comments. Several collusion networks also provide "auto-comment" services. Users are required to specify target posts to get comments. Only 7 out of the 22 collusion networks listed in Table 4 provide this service. The remaining collusion networks either do not provide comments or they ask users to input comments. We submit at least 100 posts of our honeypot accounts to garner comments from each of these seven collusion networks. Table 6 summarizes key statistics for our lexical analysis of these comments. We observe that collusion networks provide an average of only 16 comments per post. It is noteworthy that the total number of unique comments constitutes a small fraction of all comments provided by collusion networks. Overall, only 187 comments are unique out of a total of 12,959 comments. We also compute lexical richness (i.e.,

Short URL	Date Created	Short URL Click Count	Long URL Click Count	Applications	Top Referrer
goo.gl/jZ7Nyl	June 11, 2014	147,959,735	236,194,576	HTC Sense	mg-likers.com
goo.gl/4GYbBl	June 30, 2014	64,493,698	236,194,576	HTC Sense	djliker.com
goo.gl/rHnKIv	May 3, 2015	28,511,756	29,211,768	HTC Sense	sys.hublaa.me
goo.gl/2hbUps	October 4, 2014	7,000,579	7,289,920	Page Manager For iOS	autolike.vn
goo.gl/KJnSnH	November 19, 2014	7,582,494	8,223,464	HTC Sense	m.machineliker.com
goo.gl/QfLHlq	June 13, 2014	2,269,148	236,194,576	HTC Sense	begeniyor.com
goo.gl/zsaJ61	May 23, 2015	2,721,864	2,766,805	HTC Sense	www.royaliker.net
goo.gl/civ2CS	December 29, 2014	1,288,801	1,288,902	HTC Sense	oneliker.com
goo.gl/ZQwU5e	June 21, 2014	1,005,471	1,005,698	Nokia Account	adf.ly
goo.gl/nC9ciz	September 6, 2015	1,009,801	1,034,299	Sony Xperia smartphone	refer.autolikerfb.com
goo.gl/kKPCNy	January 24, 2015	297,915	236,194,576	HTC Sense	realliker.com
goo.gl/uIv2OS	February 1, 2015	355,405	1,019,830	Sony Xperia smartphone	Unknown
goo.gl/5XbAaz	January 26, 2015	165,345	1,887,940	HTC Sense	postlikers.com

Table 5: Statistics of short URLs used by collusion networks. Several short URLs point to the same long URL.

Collusion Network	Number of Posts	Average Comments Per Post	Number of Comments	Number of Unique Comments	Percentage of Unique Comments	Number of Words	Number of Unique Words	Lexical Richness (%)	ARI	% of Non-dictionary Words
myliker.com	128	19	2,499	42	1.7	7,023	80	1.1	17.8	16.2
monkeyliker.com	115	9	1,074	45	4.2	2,983	82	2.7	20.0	21.9
mg-likers.com	120	17	1,486	16	1.1	5,059	40	0.8	25.2	20.0
monsterlikes.com	100	9	930	41	4.4	3,784	102	2.7	21.6	9.8
kdliker.com	119	47	5,664	31	0.5	14,852	64	0.4	22.3	28.1
arabfblike.com	130	2	304	37	12.2	1,091	96	8.8	16.9	29.1
djliker.com	104	9	1,002	52	5.2	3,449	93	2.7	13.2	20.4
All	816	16	12,959	187	1.4	38,241	553	1.4	19.6	20.6

Table 6: Lexical analysis of comments provided by collusion networks

fraction of unique words) to quantify how many unique words are used in these comments [41]. Small lexical richness values indicate that collusion networks rely on a finite sized dictionary that results in repetition of comments. We also use Automated Readability Index (ARI) to understand the syntactic and semantic complexity of comments [41]. Larger ARI values indicate better text readability. ARI values range from 13.2 (djliker.com) to 25.2 (mg-likers.com). However, these ARI values do not necessarily indicate a high level of understandability because of *(a)* unnecessarily lengthened words like "bravooooo" and "ahhhhh", *(b)* unnecessary use of punctuation such as "?? A W E S O M E ??? ??????? ????", and *(c)* large nonsensical words such as "bfewguvchieuwver". We further use the Python Natural Language Toolkit [15] to identify English dictionary words in these comments. We note that 20.6% words are not found in English dictionary. Non-dictionary words include incorrectly spelled words such as "gr8" and "w00wwwwwww", and Hindi phrases such as "SARYE THAK KE BETH GYE". We conclude that collusion networks generate very few unique comments with limited vocabulary.

5 DISCUSSIONS

5.1 Monetization

Collusion network operators use two main mechanisms for monetizing their services: (1) advertising and (2) premium plans.

Advertising. Collusion networks attract a large number of users to their websites everyday. For example, recall from Table 5 that the short URLs used by collusion networks daily receive hundreds of thousands of clicks. Collusion networks display ads from different ad networks to monetize user traffic on their websites. We identify ad networks and trackers on collusion network websites using Ghostery [9]. We note that some collusion networks do not directly use reputable ad networks, such as DoubleClick, which is likely because collusion networks domains are blacklisted by these ad networks. To overcome this issue, several collusion networks use URL redirections to temporarily redirect users to other whitelisted domains and display ads from different ad networks on the redirected page. For example, mg-likers.com redirects users to kackroch.com where ads are served by AdSense and Atlas. Users are typically required to stay on the redirected pages for several seconds before they are redirected back to the original page. Some collusion networks redirect to paid URL shortening services such as adf.ly and sh.st. It is interesting to note that most collusion networks require users to disable ad-block extensions [44] to use their services. To this end, collusion networks use anti-adblocking scripts [37, 45] to detect popular ad-blockers and prompt users to disable ad-block extensions or whitelist their websites.

Premium Plans. Collusion networks place artificial restrictions on the number of likes/comments allowed on free plans. For example, each collusion network restricts the number of likes per post to a pre-defined limit. Most collusion networks also restrict activities by making users wait for fixed/random time between consecutive like requests. Collusion networks sell premium plans that allow users to get more likes/comments and overcome other restrictions.

Premium plans claim to provide much more likes (e.g., up to 2000 likes for the most expensive plan by mg-likers.com). They also automatically provide likes without requiring users to manually login to collusion network sites for each request.

5.2 Ownership

We want to identify the culprits behind collusion networks for designing technical and legal countermeasures. To this end, we perform WHOIS lookups of popular collusion network domains in Table 2. We observe that collusion network websites use various techniques to hide their identity. First, we find that 36% (18 out of 50) of the domains have anonymized WHOIS records using privacy protection services such as WhoisGuard [20]. For the remaining collusion networks, domain registrants are mostly located in Asian countries such as India, Pakistan, and Indonesia. In fact, more than 40% of these domain registrants are located in India. Second, we observe that most collusion network domains resolve to an IP address of CloudFlare which helps them to conceal the original hosting location. Other collusion network domains resolve to IP addresses of web hosting and cloud infrastructure providers. We also search the registrant names on popular search engines to identify their matching social media accounts. As a note of caution, we point out that the registrant information extracted from WHOIS records and social media accounts may be staged by the actual owners. Our analysis reveals that Facebook accounts of collusion network owners have a large number of followers. For example, the Facebook account of mg-likers.com's owner had more than 9 million followers at the time of this writing. We observe that the timeline posts of these Facebook accounts often have hundreds of thousands of likes. It is also noteworthy that our honeypot accounts are frequently used to like the profile pictures and other timeline posts of these Facebook accounts.

6 COUNTERMEASURES

Ethical Considerations. Before conducting any experiments, we received a formal review from our local Institutional Review Board (IRB) because we collect some publicly available account information such as posts and likes. We enforce several mechanisms to protect user privacy. For example, we do not store any personally identifiable information. We are aware that our honeypot accounts are used by collusion networks to conduct some reputation manipulation activities. We argue that these activities represent a small fraction of the overall reputation manipulation activities of collusion networks. Thus, we do not expect normal user activity to be significantly impacted by our honeypot experiments. The benefit of our honeypot approach in detecting collusion network accounts far outweighs the potential harm to regular Facebook users. To further minimize harm, as discussed next, we disclose our findings to Facebook to remove all artifacts of reputation manipulation during our measurements as well as investigate countermeasures to mitigate collusion network activities.

Before implementing any countermeasures in collaboration with Facebook, we perform honeypot experiments for approximately ten days to establish a baseline of collusion network activities. We repeat the honeypot milking experiments for popular collusion networks starting August 2016 (and continue until mid October 2016). Figure 5 shows the average number of likes received by our honeypots for two popular collusion networks.[4] As we discuss next, while we consider a wide range of countermeasures, we decide to implement countermeasures that provide a suitable tradeoff between detection of access token abuse and application platform usability for third-party developers.

Collusion networks currently exploit a few applications (listed in Table 3) to conduct reputation manipulation activities. Therefore, we can immediately disrupt all collusion networks by suspending these applications. Since collusion networks can switch between other susceptible applications listed in Table 1, we would need to suspend them as well. Suspending these applications is a relatively simple countermeasure to implement; however, it will negatively impact their millions of legitimate users. We can also make changes in Facebook's application workflow to stop access token abuse by collusion networks. For example, we can mandate application secret (thereby forcing server-side operations) for liking/commenting activities that require publish_actions permissions [1, 18]. As a result of this restriction, collusion networks will not be able to conduct reputation manipulation activities even if they retrieve access tokens from colluding users. However, many Facebook applications solely rely on client-side operations for cross-platform interoperability and to reduce third-party developer costs of server-side application management [13, 30]. Therefore, mandating application secret would adversely impact legitimate use cases for these Facebook applications.

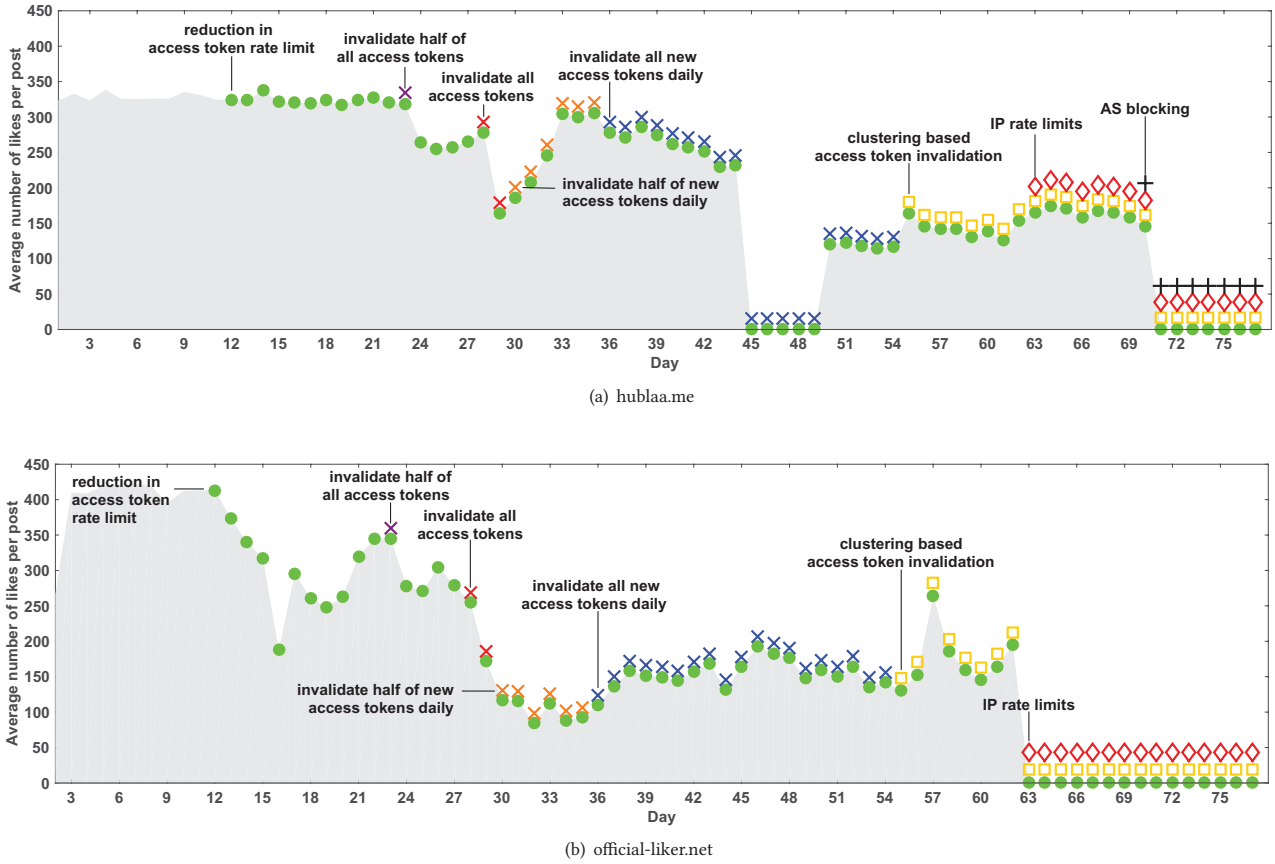
6.1 Access Token Rate Limits

As the first countermeasure, we impose restrictions on access tokens to mitigate abuse by collusion networks. Facebook employs rate limits to restrict excessive activities performed by an access token. Since collusion network activities slip under the current rate limit, we reduce the rate limit by more than an order of magnitude on day 12 as marked by green circles in Figure 5. We observe a sharp initial decrease in activities for official-liker.net. Specifically, the average number of likes provided by official-liker.net decreases from more than 400 to less than 200 on day 16. However, official-liker.net starts to bounce back after approximately one week. Moreover, this countermeasure does not impact hublaa.me. We surmise that both of these collusion networks have a large pool of access tokens which limits the need to repeatedly use them. Therefore, these collusion networks can stay under the reduced access token rate limit while maintaining their high activity levels. We do not reduce the rate limit further to avoid potential false positives.

6.2 Honeypot based Access Token Invalidation

We next invalidate access tokens of colluding accounts which are identified as part of our honeypot experiments. In the first 22 days, we have milked access tokens of 283K and 41K users for hublaa.me and official-liker.net, respectively. We expect that invalidation of these access tokens will curb collusion network activities. To this end, we invalidate randomly sampled 50% of the milked access tokens on day 23 as marked by a black cross in Figure 5.[5] We observe

[4] The results for other collusion networks are not shown due to space constraints.
[5] We decided to initially invalidate only half of the milked access tokens to avoid alerting collusion networks about our honeypots and countermeasures.



(a) hublaa.me

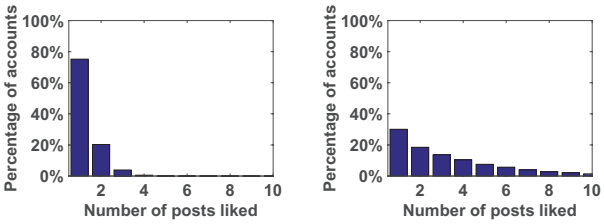
(b) official-liker.net

Figure 5: The impact of our countermeasures on two popular collusion networks. We observe that collusion network activities are not impacted by reduction in access token rate limit. While access token invalidation significantly reduced collusion network activities, it cannot completely stop them. Clustering based access token invalidation also does not help. Our IP rate limits effectively counter most collusion networks that use a few IP addresses. We can target autonomous systems (ASes) of collusion networks that use a large pool of IP addresses.

a sharp decrease in collusion network activities. Specifically, the average number of likes provided by hublaa.me decreases from 320 to 250 and for official-liker.net decreases from 350 to 275. Unfortunately, this decline was not permanent and the average number of likes gradually increase again over the next few days. We surmise that collusion networks gradually replenish their access token pool with fresh access tokens from new and returning users.

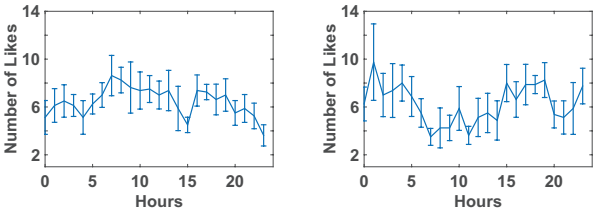
To mitigate this, we next invalidate all access tokens that are observed until day 28 (marked by red cross) and also begin invalidating 50% of newly observed access tokens on a daily basis (marked by orange cross). We observe a sharp decline for both hublaa.me and official-liker.net on day 28 when we invalidate all access tokens. However, average likes by hublaa.me start to bounce back and those by official-liker.net stabilize at 100 over the next few days. We suspect that the rate of fresh access tokens from new and returning users exceeds our rate of daily access token invalidation. This is due to the rather small number of distinct new colluding accounts milked daily by our honeypots.

To increase our access token invalidation rate, starting day 36, we begin invalidating all newly observed access tokens on a daily basis as marked by blue cross in Figure 5. We observe a steady decrease in average likes by hublaa.me from day 36 to day 44. hublaa.me's site was temporarily shutdown on day 45. The site resumed operations on day 51 and their average number of likes decreased to 120. official-liker.net sustained their likes between 110-192 despite our daily access token invalidation. While regular access token invalidation curbed collusion network activities, we conclude that it cannot completely stop them because honeypot milking can only identify a subset of all newly joining users. Therefore, we decide not to pursue regular access token invalidation further.



(a) hublaa.me (b) official-liker.net

Figure 6: Number of our honeypot posts liked by collusion network accounts. We observe that a small fraction of collusion network accounts like multiple honeypot posts.

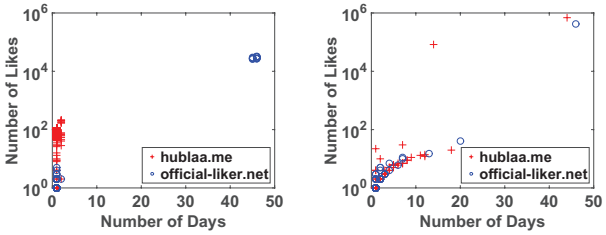


(a) hublaa.me (b) official-liker.net

Figure 7: Hourly timeseries of number of likes performed by our honeypot accounts. We observe that collusion networks spread out liking activities performed by our honeypot accounts over time.

6.3 Temporal Clustering

Collusion networks provide likes on submitted posts in less than one minute. Such bursts of liking activity can be detected by temporal clustering algorithms [24, 28, 40] which are designed to detect accounts that act similarly at around the same time for a sustained period of time. Starting day 55, as marked by cyan squares in Figure 5, we use SynchoTrap [28] to cluster synchronized access token abuse by collusion network accounts. Surprisingly, we do not observe any major impact on collusion network activities. Our drill-down analysis showed that collusion networks avoid detection by (1) using different set of accounts to like target posts and (2) spreading out liking activities performed by each access token over time. Figure 6 shows that different sets of accounts like posts of our honeypot accounts. We note that 76% and 30% accounts like at most one post of our honeypots for hublaa.me and official-liker.net, respectively. Figure 7 shows that collusion networks do not binge use our honeypot accounts within a short timeframe. We note that the hourly average of likes performed by our honeypot accounts ranges between 5-10. Therefore, collusion network accounts do not behave similarly at around the same time for a sustained period of time.



(a) IP addresses (b) Autonomous Systems (ASes)

Figure 8: Source IP addresses and ASes of Facebook Graph API requests by hublaa.me and official-liker.net to like posts of our honeypot accounts

6.4 IP and AS based Limits

We next target the origin of collusion networks to further mitigate their activities. To this end, we track the source IP addresses of the Facebook Graph API requests for liking posts of our honeypot accounts. Figure 8(a) shows the scatter plot of these IP addresses where x-axis represents the number of days an IP address is observed during our countermeasures and y-axis represents the total number of likes generated by each IP address. It is noteworthy that a few IP addresses account for a vast majority of likes for official-liker.net. Therefore, we imposed daily and weekly IP rate limits on the like requests beginning day 46. Note that this rate limit will not impact activities of normal users (e.g., regularly accessing Facebook via commodity web browsers) because this IP rate limit is only applicable to like requests by the Facebook Graph API using access tokens. Figure 5 shows that official-liker.net stops working immediately after we impose IP rate limits. Although not shown in Figure 5 due to space constraints, other popular collusion networks in Table 4 also stopped working on day 63. The only exception is hublaa.me, which uses a large pool of more than 6,000 IP addresses and circumvents the IP rate limits. Further analysis in Figure 8(b) reveals that all of hublaa.me's IP addresses belong to two distinct autonomous systems (ASes) of bulletproof hosting providers [23]. On day 70, we start to block like requests from these ASes for susceptible applications (e.g., listed in Table 1) which helps in ceasing all likes from hublaa.me. Note that we target a small set of susceptible applications for AS blocking to mitigate the risk of collateral damage to other applications.

6.5 Limitations

First, our countermeasures should not result in collateral damage while being robust to evasion attempts by collusion networks. To date, we have not received any collateral damage complaints from popular third-party developers. Therefore, we conclude that our countermeasures do not result in significant false positives. Second, our countermeasures need to be robust against potential evasion attempts by collusion networks. Our countermeasures have proven to be long-lasting for several months now. In future, collusion networks can try to evade our countermeasures in several ways. For example, collusion networks can use many different IP addresses and ASes (e.g., using botnets and proxies) to circumvent our IP- and

AS-based countermeasures. If and when that happens, we can again use honeypots to swiftly identify IP addresses and ASes used by collusion networks. Third, collusion networks may try to identify the honeypot accounts that we use to infiltrate them. For example, collusion networks can try to detect our honeypots accounts which currently make very frequent like/comment requests. To circumvent such detection, we can create multiple honeypot accounts to decrease the frequency of per-account like/comment requests.

7 RELATED WORK

7.1 Third-party Applications

Twitter and Facebook allowed third-party developers to create applications starting 2006 and 2007, respectively. Millions of applications use third-party application platforms supported by popular online social networks. For example, more than 30 million applications and websites currently use Facebook's third-party application platform [8].

Online social networks use OAuth for granting access to user accounts to third-party applications. OAuth [30, 31] is an IETF standardized authorization framework which allows third-party applications to gain restricted access to user accounts without sharing authentication credentials (i.e., username and password). OAuth was designed by a community of web developers to enable delegated access to protected resources. OAuth 1.0 was published as RFC 5849 [31] and later OAuth 2.0 was published as RFC 6749 [30]. Online social networks implement different variations of the OAuth protocol. For example, Twitter implements OAuth 1.0a[6] while Facebook, YouTube, Instagram, and SoundCloud implement OAuth 2.0. Researchers have uncovered many security issues in the OAuth framework and its implementations over the years[33, 35, 49, 54, 61, 66]. In particular, researchers have highlighted security vulnerabilities that arise due to the underlying issues in the protocol and bad implementation practices that prefer simplicity over security. For example, Sun *et al.* [54] discussed the inherent security issues in client-side flows which can be exploited by an attacker to steal access tokens by cross-site scripting or eavesdropping. The OAuth 2.0 specification [30] also highlights security implications of the client-side flow and potential access token leakage. While the security issues in the OAuth protocol have been previously reported [16], we are the first to report on and effectively disrupt large-scale abuse of leaked access tokens for reputation manipulation. Below we discuss prior work on exploitation of third-party applications in Facebook and Twitter.

Makridakis *et al.* [43] showed that malicious Facebook applications can leak users' private data, can retrieve remote files from a user's machine, and even launch denial of service attacks. Rahman *et al.* [47, 48] proposed FRAppE to detect malicious Facebook applications. They identified distinguishing features of malicious Facebook applications (e.g., name sharing, ask for fewer permission requests, have redirect URIs to domains with poor reputation scores) and trained a machine learning classifier to detect them. In contrast, Facebook applications used by collusion networks studied in our work request all read/write permissions and use Facebook's recommended redirect URIs. Thus, the feature set used by FRAppE will not detect the legitimate Facebook applications exploited by collusion networks.

Twitter applications are also abused to launch spam campaigns. For example, Stringhini *et al.* [53] and Thomas *et al.* [56] showed that Twitter spammers rely on third-party applications to gain control of user accounts. Twitter blocks the third-party applications that are used by spammers; however, spammers periodically create new applications to overcome this problem [53]. Note that Facebook collusion networks studied in our work do not create new applications; instead, they exploit existing legitimate third-party applications for access token leakage and abuse.

7.2 Honeypots

Honeypots have proven to be an effective tool to study reputation manipulation in online social networks. The basic principle of honeypots is to bait and deceive fraudsters for understanding their activities. In a seminal work, Webb *et al.* [62] introduced the idea of using honeypots to study spam in online social networks. They used 51 honeypot accounts on MySpace to identify 1,570 bogus accounts over the course of 5 months. Lee *et al.* [42] used an undisclosed number of honeypot accounts on Twitter to identify 500 bogus accounts over the course of 2 months. Stringhini *et al.* [52] used 900 honeypot accounts on Facebook, MySpace, and Twitter to identify 15,762 bogus accounts over the course of 12 months. Boshmaf *et al.* [27] designed and deployed a Socialbot network consisting of approximately 100 interactive honeypot accounts to infiltrate more than 8,000 Facebook users with a success rate of up to 80%. Wang et al. [60] paid users on two crowdturfing websites in China (zhubajie.com and sandaha.com) to post spam messages to Weibo, QQ instant message groups, and discussion forums. Cristofaro *et al.* [29] paid 4 different Facebook like farms to like their honeypot Facebook pages, and identified 4,145 bogus accounts over the course of 15 days. Viswanath *et al.* [58] paid black-market services and collusion networks to like their honeypot Facebook pages, and identified 5,647 bogus accounts. Aggarwal *et al.* [22] created honeypot accounts on Twitter to acquire more than 170K followers from 57 different premium and freemium collusion services on Twitter. Song *et al.* [50] paid 9 different crowdturfing services to get more than 19K retweets on the tweets posted by their honeypot Twitter accounts. Some Twitter black-market and collusion services ask for users' credentials, while others ask a user to install their third-party applications with write permissions. In contrast, Facebook collusion networks operate differently as they exploit security weaknesses in legitimate Facebook applications.

The collusion networks studied by Viswanath *et al.* [58] (addmefast.com and likesasap.com) use a web-based credit system to perform reputation manipulation on Facebook, Twitter, YouTube, and SoundCloud. These collusion networks require users to manually perform and report their reputation manipulation activities to receive credit. In contrast, Facebook collusion networks studied in our work are fully automated using the Facebook Graph API; thus, the scale of their reputation manipulation activities is much larger than addmefast.com and likesasap.com. In contrast to prior work, we deploy honeypot accounts to join collusion networks and milk them by repeatedly requesting likes on the posts of our honeypot

[6]OAuth 1.0a is a revision of OAuth 1.0 to address some security issues identified in OAuth 1.0.

accounts. An in-depth investigation of these large-scale Facebook collusion networks is lacking in prior literature.

7.3 Detecting Reputation Manipulation

There is a large body of research on detecting reputation manipulation in online social networks. Lee *et al.* [42] trained a machine learning classifier based on content features, social graph characteristics, and posting patterns to detect spammers on MySpace and Twitter. Stringhini *et al.* [52] trained a machine learning classifier based on features such as message similarity and URL ratio to detect spammers on Facebook and Twitter. Viswanath *et al.* [58] designed a Principal Component Analysis (PCA) based technique to detect anomalous Facebook accounts based on features such as timeseries of like count and distribution of liked pages across content categories. Boshmaf *et al.* [26] detected fake accounts on Facebook and Tuenti by predicting targets based on features such as gender, number of friends, time since last update, etc. Song *et al.* [50] trained a machine learning classifier to detect reputation manipulation targets on Twitter based on features such as retweet time distribution, ratio of the most dominant application, number of unreachable retweeters, and number of received clicks. Jian *et al.* [40] proposed an outlier detection approach to detect synchronized and rare activity patterns on Twitter and Weibo. Recent studies have also proposed graph based algorithms to detect reputation manipulation[24, 28, 34]. Beutel *et al.* [24] proposed a bipartite graph clustering algorithm to detect suspicious lockstep activity patterns on Facebook. Cao *et al.* [28] proposed graph clustering algorithm to detect large groups of malicious accounts that act similarly for a sustained period of time on Facebook and Instagram. Hooi *et al.* [34] proposed a camouflage-resistant bipartite graph clustering algorithm to detect fraudulent users in online social networks.

These methods can be leveraged to detect both collusion network member accounts and their reputation manipulation activities. However, collusion networks are different in the sense that the accounts used for fraudulent activities are not all fake. Our manual analysis of collusion network accounts indicated that many accounts likely belong to real users who voluntarily allow collusion networks to conduct reputation manipulation on their behalf. As our experiments shows, these colluding accounts are hard to detect because they mix real and fake activity.

8 CONCLUDING REMARKS

We presented a comprehensive measurement study of collusion-based reputation manipulation services on Facebook. We made three major contributions. First, we uncovered that collusion networks exploit Facebook applications with weak security settings to retrieve access tokens and abuse them for reputation manipulation. Second, we deployed honeypots to understand the operations of collusion networks and to demonstrate the scale of the problem. We were able to identify more than one million active collusion network members. Third, we implemented a series of countermeasures to effectively thwart the operations of collusion networks.

Our results raise a number of questions that motivate future research. First, we would like to investigate potential access token leakage and abuse on other popular online services that implement OAuth 2.0. For instance, YouTube, Instagram, and SoundCloud implement OAuth 2.0 to support third-party applications. Second, in addition to reputation manipulation, attackers can launch other serious attacks using leaked access tokens. For example, attackers can steal personal information of collusion network members as well as exploit their social graph to propagate malware. We plan to investigate other possible attacks as well. Third, while our simple countermeasures have been effective now for more than six months, collusion networks may start using more sophisticated approaches to evade them in future. We plan to investigate more sophisticated machine learning based approaches to robustly detect access token abuse. We are also interested in developing methods to detect and remove reputation manipulation activities of collusion network members. Finally, a deeper investigation into the economic aspects of collusion networks may reveal operational insights that can be leveraged to limit their financial incentives.

ACKNOWLEDGMENTS

We would like to thank Usama Naseer, Minhaj-Us-Salam Khan, and Mohammad Raza Hussain for their help with an initial investigation of the problem. We would also like to thank our shepherd, Andrew Moore, and the anonymous reviewers for their useful feedback on this paper. This work is supported in part by the National Science Foundation under grant number CNS-1715152 and by an unrestricted gift from Facebook.

REFERENCES

[1] Access Tokens. https://developers.facebook.com/docs/facebook-login/access-tokens.
[2] Alexa - Actionable Analytics for the Web. http://www.alexa.com.
[3] App Review. https://developers.facebook.com/docs/apps/review.
[4] Death By Captcha | Best and cheapest captcha service. http://www.deathbycaptcha.com/.
[5] Facebook - Annual Report. http://investor.fb.com/secfiling.cfm?filingID=1326801-16-43&CIK=1326801.
[6] Facebook Apps Leaderboard. http://www.appdata.com/leaderboard/apps.
[7] Facebook Login for Apps - Developer Documentation. https://developers.facebook.com/docs/facebook-login.
[8] Facebook's F8 developers conference by the numbers. http://fortune.com/2015/03/25/facebook-f8-developers-conference-numbers/.
[9] Ghostery Tracker Browser Extension. https://www.ghostery.com/our-solutions/ghostery-browser-extention/.
[10] Google Trends. https://www.google.com/trends/.
[11] Graph API Reference /object/likes. https://developers.facebook.com/docs/graph-api/reference/v2.9/object/likes.
[12] Internet Archive: Wayback Machine. https://archive.org/web/.
[13] Login Security - Facebook Login. https://developers.facebook.com/docs/facebook-login/security.
[14] Manually Build a Login Flow - Facebook Login. https://developers.facebook.com/docs/facebook-login/manually-build-a-login-flow.
[15] Natural Language Toolkit. http://www.nltk.org.
[16] OAuth 2.0 Threat Model and Security Considerations. https://tools.ietf.org/html/rfc6819.
[17] Python SDK for Facebook's Graph API. https://github.com/mobolic/facebook-sdk.
[18] Securing Graph API Requests. https://developers.facebook.com/docs/graph-api/securing-requests.
[19] Selenium - Web Browser Automation. http://www.seleniumhq.org/.
[20] WhoisGuard: Protect your privacy using WhoisGuard (Whois protection). http://www.whoisguard.com/.
[21] Twitter Inc. - Quarterly Report. http://files.shareholder.com/downloads/AMDA-2F526X/3022492942x0xS1564590-16-21918/1418091/filing.pdf, August 2016.
[22] A. Aggarwal and P. Kumaraguru. What they do in shadows: Twitter underground follower market. In *13th IEEE Annual Conference on Privacy, Security and Trust (PST)*, 2015.

[23] S. Alrwais, X. Liao, X. Mi, P. Wang, X. Wang, F. Qian, R. Beyah, and D. McCoy. Under the Shadow of Sunshine: Understanding and Detecting Bulletproof Hosting on Legitimate Service Provider Networks. In *IEEE Symposium on Security and Privacy*, 2017.

[24] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos. CopyCatch: Stopping Group Attacks by Spotting Lockstep Behavior in Social Networks. In *WWW*, 2013.

[25] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda. All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks. In *WWW*, 2009.

[26] Y. Boshmaf, D. Logothetis, G. Siganos, J. Leria, J. Lorenzo, M. Ripeanu, and K. Beznosov. Integro: Leveraging Victim Prediction for Robust Fake Account Detection in OSNs. In *NDSS*, 2015.

[27] Y. Boshmaf, I. Muslukhov, K. Beznosov, and M. Ripeanu. The socialbot network: when bots socialize for fame and money. In *ACM ACSAC*, 2011.

[28] Q. Cao, X. Yang, J. Yu, and C. Palow. Uncovering Large Groups of Active Malicious Accounts in Online Social Networks. In *ACM CSS*, 2014.

[29] E. D. Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and M. Z. Shafiq. Paying for Likes?: Understanding Facebook Like Fraud Using Honeypots. In *ACM Internet Measurement Conference (IMC)*, 2014.

[30] E. D. Hardt. The OAuth 2.0 Authorization Framework. IETF RFC 6749, October 2012.

[31] E. E. Hammer-Lahav. The OAuth 1.0 Protocol. IETF RFC 5849, April 2010.

[32] D. Fett, R. Kusters, and G. Schmitz. A Comprehensive Formal Security Analysis of OAuth 2.0. In *ACM CCS*, 2016.

[33] E. Hammer. OAuth 2.0 and the Road to Hell. https://hueniverse.com/oauth-2-0-and-the-road-to-hell-8eec45921529.

[34] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *ACM KDD*, 2016.

[35] P. Hu, R. Yang, Y. Li, and W. Lau. Application impersonation: problems of OAuth and API design in online social networks. In *ACM COSN*, 2014.

[36] L. Invernizzi, P. Milani, S. Benvenuti, C. Kruegel, M. Cova, and G. Vigna. EvilSeed: A guided approach to finding malicious web pages. In *IEEE Symposium on Security and Privacy*, 2012.

[37] U. Iqbal, Z. Shafiq, and Z. Qian. The Ad Wars: Retrospective Measurement and Analysis of Anti-Adblock Filter Lists. In *ACM Internet Measurement Conference (IMC)*, 2017.

[38] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer. SOCIAL PHISHING. *Communications of the ACM*, 2007.

[39] M. Javed, C. Herley, M. Peinado, and V. Paxson. Measurement and Analysis of Traffic Exchange Services. In *ACM Internet Measurement Conference (IMC)*, 2015.

[40] M. Jian, P. Cui, A. Beutel, C. Faloutsos, and S. Yang. CatchSync: catching synchronized behavior in large directed graphs. In *ACM KDD*, 2014.

[41] D. Jurafsky, J. H. Martin, P. Norvig, and S. Russell. *Speech and Language Processing*. Pearson, 2014.

[42] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: social honeypots + machine learning. In *ACM SIGIR*, 2010.

[43] A. Makridakis, E. Athanasopoulos, S. Antonatos, D. Antoniades, S. Ioannidis, and E. P. Markatos. Understanding the behavior of malicious applications in social networks. *IEEE Network*, 24(5):14–19, 2010.

[44] M. H. Mughees, Z. Qian, and Z. Shafiq. Detecting Anti Ad-blockers in the Wild. In *Privacy Enhancing Technologies Symposium (PETS)*, 2017.

[45] R. Nithyanand, S. Khattak, M. Javed, N. Vallina-Rodriguez, M. Falahrastegar, J. E. Powles, E. D. Cristofaro, H. Haddadi, and S. J. Murdoch. Adblocking and Counter-Blocking: A Slice of the Arms Race . In *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2016.

[46] M. Z. Rafique, T. V. Goethem, W. Joosen, C. Huygens, and N. Nikiforakis. It's Free for a Reason: Exploring the Ecosystem of Free Live Streaming Services . In *Network and Distributed System Security Symposium (NDSS)*, 2016.

[47] M. S. Rahman, T.-K. Huang, H. V. Madhyastha, and M. Faloutsos. FRAppE: Detecting Malicious Facebook Applications. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2012.

[48] S. Rahman, T.-K. Huang, H. V. Madhyastha, , and M. Faloutsos. Detecting Malicious Facebook Applications. *IEEE/ACM Transactions on Networking*, 24(2):773–787, 2016.

[49] E. Shernan, H. Carter, D. Tian, P. Traynor, and K. Butler. More Guidelines Than Rules: CSRF Vulnerabilities from Noncompliant OAuth 2.0 Implementations. In *Proceedings of the International Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, 2015.

[50] J. Song, S. Lee, and J. Kim. CrowdTarget: Target-based Detection of Crowdturfing in Online Social Networks. In *ACM CCS*, 2015.

[51] T. Stein, E. Chen, and K. Mangla. Facebook immune system. In *Workshop on Social Network Systems (SNS)*, 2011.

[52] G. Stringhini, C. Kruegel, and G. Vigna. Detecting Spammers on Social Networks. In *Annual Computer Security Applications Conference (ACSAC)*, 2010.

[53] G. Stringhini, G. Wang, M. Egele, C. Kruegel, G. Vigna, H. Zheng, and B. Y. Zhao. Follow the Green: Growth and Dynamics in Twitter Follower Markets. In *ACM Internet Measurement Conference (IMC)*, 2013.

[54] S.-T. Sun and K. Beznosov. The Devil is in the (Implementation) Details: An Empirical Analysis of OAuth SSO Systems. In *ACM CCS*, 2012.

[55] K. Thomas, D. latskiv, E. Bursztien, T. Pietraszek, C. Grier, and D. McCoy. Dialing Back Abuse on Phone Verified Accounts. In *ACM CCS*, 2014.

[56] K. Thomas, F. Li, C. Grier, and V. Paxson. Consequences of Connectivity: Characterizing Account Hijacking on Twitter. In *ACM CCS*, 2014.

[57] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson. Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *USENIX Security Symposium*, 2013.

[58] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove. Towards Detecting Anomalous User Behavior in Online Social Networks. In *USENIX Security Symposium*, 2014.

[59] G. Wang, T. Wang, H. Zheng, and B. Y. Zhao. Man vs. Machine: Practical Adversarial Detection of Malicious Crowdsourcing Workers. In *USENIX Security Symposium*, 2014.

[60] G. Wang, C. Wilson, X. Zhao, Y. Zhu, M. Mohanlal, H. Zheng, and B. Y. Zhao. Serf and Turf: Crowdturfing for Fun and Profit. In *WWW*, 2012.

[61] R. Wang, Y. Zhou, S. Chen, S. Qadeer, D. Evans, and Y. Gurevich. Explicating SDKs: Uncovering Assumptions Underlying Secure Authentication and Authorization. In *USENIX Security Symposium*, 2013.

[62] S. Webb, J. Caverlee, and C. Pu. Social Honeypots: Making Friends With A Spammer Near You. In *Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, 2008.

[63] H. Xu, D. Liu, H. Wang, and A. Stavrou. E-commerce Reputation Manipulation: The Emergence of Reputation-Escalation-as-a-Service. In *WWW*, 2015.

[64] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks. In *IEEE Symposium on Security and Privacy*, 2008.

[65] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: defending against sybil attacks via social networks. In *ACM SIGCOMM*, 2006.

[66] Y. Zhou and D. Evans. SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities. In *USENIX Security Symposium*, 2014.