SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 17, No. 3, October 2020, 357-375

UDC: 004.056:004.738.5 **DOI: https://doi.org/10.2298/SJEE2003357R**

Malware Command and Control Over Social Media: Towards the Server-less Infrastructure

Vladimir Radunović[1], Mladen Veinović[2]

Abstract: Intrusions into the computer systems are becoming increasingly sophisticated. Command and Control (C2) infrastructure, which enables attackers to remotely control infected devices, is a critical component. Malware is set to connect to C2 servers to receive commands and payloads, or upload logs or stolen files. Since techniques for detecting traditional C2 servers are also advancing, attackers look for ways to make C2 communication stealth and resilient. Increasingly, they hide C2 communications in plain sight, in particular on social media and other cloud-based public services. In this paper, we identify several emerging trends in the use of social media for C2 communications by providing a review of the existing research, discuss how attackers could combine these trends in the future to create a stealth and resilient server-less C2 model, look at possible defence aspects, and suggest further research.

Keywords: Command and control, Botnet, Social media, Encryption, Security.

1 Introduction

Modern cyber-intrusion schemes are becoming increasingly complex. Cyber Kill Chain® framework [1], a recognized model of the structure of complex attacks developed by Lockheed Martin, outlines seven steps required by attackers in order to succeed in a cyber-intrusion: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command & Control (C2), and Actions on objectives. Another recognized model, ATT&CK developed by MITRE [2], includes Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement, Collection, Command and Control, Exfiltration, and Impact. While the former is somewhat broader and the latter instead focuses on the tools and details of steps in delivering attacks, for both models a resilient and stealth C2 infrastructure is essential to enable attackers – malware controllers – to remotely issue commands to the infected devices. Botnets, networks of infected and 'hijacked' devices, in particular, rely heavily on C2 commands [3]. Conducting espionage activities, or other sophisticated intrusions known as an advanced persistent

[1]DiploFoundation, Braničevska 12a, 11000 Beograd, Serbia; E-mail: vladar@diplomacy.edu
[2]Singidunum University, Danielova 32, 11000 Beograd, Serbia; E-mail: mveinovic@singidunum.ac.rs

threat (APT), also relies on two-way C2 communication between the infected devices (here referred to as bots, regardless of whether they are part of a broader botnet) and the controllers: Ussath et al. [4] show that 18 out of 22 analysed well-known APT campaigns used at least one form of C2 for providing commands or exfiltrating sensitive data.

Since most modern cyber-attacks would not be possible – or at least nearly as effective – without C2 infrastructure, finding ways to locate and put down C2 infrastructure is critical for combating cyber-attacks and reducing cyber-threats. As defenders – computer emergency response teams (CERT), cybersecurity companies, or law enforcement authorities – advance in finding methods to discover and put down C2 servers, attackers also explore new and creative ways to remain stealth, or at least maintain the possibility to re-configure the malware under their control without the need to re-infect devices. In this regard, attackers also deploy C2 infrastructure based on social media3 (here referred to as SM C2), with some attempts to form a server-less C2 by using public SM services not only as a C2 channel, but also as C2 server to dispose of stolen information.

To think ahead of attackers, we need to better understand the emerging SM C2 models, and how they may further evolve. By reviewing academic and professional literature, we firstly provide an analysis of the five key trends related to the use of SM C2; secondly, we suggest a possible stealth and resilient server-less C2 model that combines these trends; thirdly, we briefly discuss the strengths and weaknesses of such a model; and fourthly, we offer particular ideas for further research on how to pre-empt such C2 models.

2 Related Work

While C2 infrastructure has been the subject of the scientific research for years, a particular focus was placed on detecting and classifying DNS-based C2; Dietrich et al. provide valuable insight on how DNS C2 is used by botnets [5]. New C2 models based on social media, however, have been in focus in the last several years. One of the first botnets to use Twitter as C2 was discovered in 2009 by Nazario, which abuses trusted popular sites and common ports [6], thus making it harder to be blocked. Brezo et al. [7] analysed various emerging C2 structures, such as IRC, Peer-to-Peer, and HTTP/HTTPS, but concluded that SM C2 will gain popularity, also due to 24/7 availability. Makkar et al. [8] designed their SocioBot model that uses Twitter as C2, and developed a Docker-based simulation which can be used to study it – and other botnets – in a controlled environment. Nagaraja et al. [9] moved in a different direction, proposing a stealth peer-to-peer botnet that uses social media to communicate

[3] Social media is understood as any online services, platforms, and applications that allow users to create and share content or connect socially (social networks, thus, representing only one particular type of platform).

among peers as well as with C2, while embedding communication messages into images by using steganography. While of great value for the analysis of possible components of botnet and C2 infrastructure, this model is of little practicality to attackers due to limiting communications only among peers that have already established social contacts.

Some of the existing botnet detection mechanisms can still apply to those with SM C2, but some are inefficient. Ghafir, Svoboda and Prenosil [10] outline two main approaches for identifying botnets: traffic detection, which is of particular relevance for identifying the existence of botnets, and honeypots, which allow conducting an analysis of attracted botnets and detecting their C2 infrastructure. While honeypots can attract most types of botnets, including those with SM C2, traffic detection is not fully applicable since bots that communicate with SM C2 do not use suspicious domains, protocols, or ports. Sood [11] looked into exploiting the weaknesses of C2 panels through which attackers control C2 - a method which is not applicable to botnets where public SM sites are used as C2 servers, as will be discussed later. Kartaltepe et al [12] looked at server-side techniques for sociobots that use social networks for C2, and tested several application-centric solutions that are complementary to traditional host-centric and network-centric ones. Yet, Ji at al. [13] rightly point out to the limitations of the proposed server-side techniques, which do not cover steganographic format of C2 messages, nor can encompass all the options for encrypting C2 messages. They offered and tested new detection features based on host-side behaviour, including viewing the number of visited social media accounts, messages, and images – factors which are relevant but could be harder to follow as social media platforms become our daily routine.

In practice, the pace of innovation (and creativity) by attackers, however, is increasing, and it is mostly the lead cybersecurity companies, like Sofos, ESET, Kaspersky or FireEye, that manage to catch up with trends by providing technical analysis and white papers. Researchers, for their part, are slower to follow the trends, and there is a need to enhance regular scientific mapping and analysis of these trends, by critically reviewing the lead commercial reports.

3 C2 Communication Models and Characteristics

In principle, C2 allows bots to contact a remote server, under the direct control of the attacker, to

– fetch updates, specific payloads, or additional commands;
– deploy backdoor functions such as reconnaissance (getting system information and local drives), uploading stolen content, creating and managing processes, or stealing credentials;
– signal its success or failure by contacting the remote C2 servers.

C2 can also provide a 'kill switch' signal –when to cease operations: in his analysis of the WannaCry malware and its variants, Mackenzie [14] explained a function in which a bot checks a specific domain to see if it is alive, and suspends further operations in case it is. However, he reasonably suggested that it was likely that the function was introduced to avoid possible sandbox environments.

As a rule of thumb, C2 communications should be easy to set up, easy to control, and hard to discover and put down. Importantly, it also needs to avoid easily traceable connections to real persons: in their survey of technical artefacts that allow the attribution of attacks, and thereby possible prosecution, Skopik and Pahi [15] conclude that traces left outside of the victim's organisation – in particular setting up C2 (acquiring domains, renting cloud services, etc.) – are harder to fake and easier to trace.

Commonly, C2 servers are made accessible through DNS addresses consisted of uncommon or incomprehensive strings (to make sure that the domains are not or would not be registered by other users for legitimate purposes). In order to inform malware of the DNS or IP location of a remote C2 server, attackers may hard-code it in the malware code or the supporting files. Marko and Vilhan [16], however, observe that this has become a vulnerable point of botnets: while this facilitates the establishment of the C2 channel, once C2 is discovered and disabled by defenders, the malware is rendered useless wherever it has been implanted, and the new infection cycle with improved malware needs to be conducted. In order to extend the life of the deployed malware, the location of a C2 server should be changed frequently, yet malware still needs to know where to find it. They further analyse a popular model in which the domain of a C2 server is generated from the predefined pool, and propose a solution to slow down the DNS queries rate, to prevent the bot from functionally communicating with a C2 server.

To enable the more dynamic and thus resilient mode of C2 communication, the malware is set to connect with public online services to obtain the location of C2 servers. This allows attackers to change the location of C2 servers as they need, while the infected devices still have their full functionality. Traditionally, the IRC chatrooms were used, particularly with botnets, which enabled bot-masters to control individual bots, while hiding its communications with C2 within other typical IRC traffic [17]. Specific detection systems were suggested, however, to identify IRC-based C2 botnets as well, such as the *Rishi* toolkit devised and tested by Göbel and Holz [18] which passively monitors traffic for unusual or suspicious IRC names, servers, or ports – a method that is worth exploring for some of the SM C2 infrastructures as well. The explosion of social media services has enabled attackers to turn to new modes of securing their channels – by hiding in plain sight. This is becoming the most common C2 approach – the easiest to run and the hardest to identify and combat. Lehtiö [19] provides a very useful overview of practical cases of third-party web services (ab)use as C2, including popular social networks, and concludes that the area needs more research. While his research is based entirely on the reports by the cybersecurity industry, rather than the academic literature, he did manage to cover the most illustrative cases by referencing very credible community sources.

4 New Trends with Social Media C2 Infrastructure

Social media – social networks in particular, but more broadly also cloud-based public services such as discussion forums or shared repositories – have become a medium for establishing hidden communication between bots and C2 infrastructure (and, in some cases, have become C2 infrastructure itself). Recent trends outline the heavy use of a number of social media services, as presented in **Table 1**.

Table 1
Social media platforms used for C2, *with typical locations and type of inputs.*

Public cloud-based service	*Location of* C2 *input*	*Type of* C2 *input*	*Reference*
Twitter	Posts	Links, text commands, images	[8, 20]
Facebook	Posts and comments	Links, text commands, images	[12]
Instagram	Posts and comments	Links, text commands, images	[21, 22]
YouTube	Comments	Links, text commands	[12]
Reddit	Posts	Links, text commands	[23]
Tumblr	Posts	Links, text commands, images	[19, 24]
PasteBin	Public notes	Links, text commands	[25]
Evernote	Public notes	Links, text commands, images	[26]
Google Documents	Shared documents	Links, text commands, images	[19, 25]
Dropbox, SendSpace	File sharing	Images, files	[25, 27]
GitHub	File sharing	Images, files	[25]

Analysis of examples from practice [8, 20 – 25] shows that, typically, locations of C2 servers, or direct commands are posted publicly, in encrypted format, through various online platforms, and bots are instructed to look for those in particular public places. To enable this, perpetrators continuously create numerous fake accounts on social media platforms, and use them to post messages containing encrypted C2 locations or commands. For instance, they may post encrypted code by tweeting from particular bogus accounts, leave comments on Instagram posts of other legitimate users, or contribute to public blogs and forum discussions. As an example, Kartaltepe et al. [12] show that each bot in Nazario's proof-of-concept botnet is instructed to request the URL of C2 through a Twitter message, with an `HTTP GET` command, to a bogus Twitter account. It then returns (via Twitter) Base64-encoded text. After decoding, the bot reveals a URL shortened by a public online links shortener bit.ly, which then leads the bot to a full URL of a server that hosts malicious zip files with the payload. From these and related reports and articles, we identify five notable emerging trends related to the use of SM C2.

4.1 Hiding C2 commands in text-based SM posts

On the bot side, malware is instructed to search through a specific public page (such as a social media account), and look for commands in the particular SM posts. Analysing backdoor called *Protux*, Chen [28] reflects on the change in C2 infrastructure it used over years: from retrieving C2 IP addresses through DNS server in 2005, to finding information in posts on blog services. In a *Blackgear* cyberespionage campaign, in particular, both the *Marade* downloader and the *Protux* backdoor deployed looked for the public posts on the Facebook profiles controlled by attackers, and snapped the encrypted configurations. Its weak point could be, however, the removal of the C2 Facebook profile which is hard-coded in the malware code.

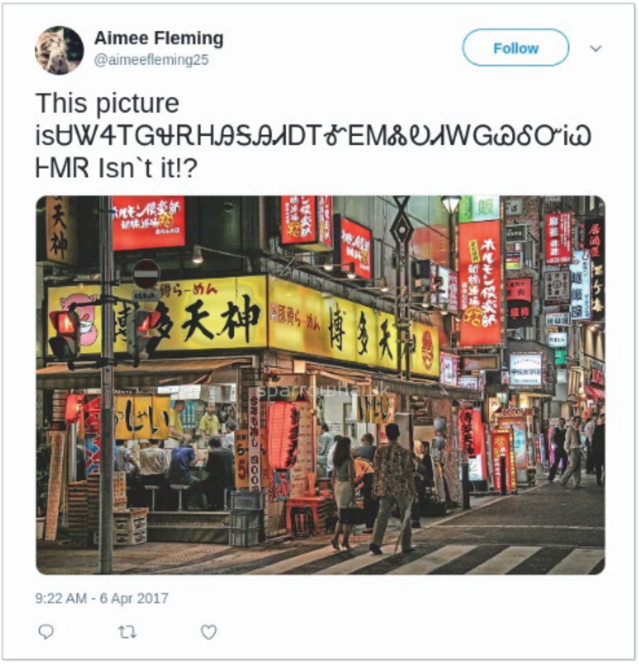
To be able to easily switch to new C2 accounts once old ones have been reported for abuse and suspended, a bot master can mark the beginning of the command string in a public post on any of its accounts with a particular mark. A bot will then not look for a specific account, but instead search for particularly marked text strings, then parse through the marked text, and decode and decrypt. In the technical part of its complaint to the US court, Microsoft [29] revealed that a hacker group known as *Barium* used string delimiters, such as $$, {} or /\, to mark coded locations of C2 servers within account profile descriptions fields on:

– Microsoft's Microsoft Developer Network forum for software developers: `Social.msdn.microsoft.com/Profile/<ActorControlledProfile>`
– Microsoft's TechNet forum for software developers: `Social.technet.microsoft.com/Profile/<ActorControlledProfile>`

– Microsoft's Forums: `Social.microsoft.com/Profile/<ActorControlledProfile>`

The infected device would visit the profile and search for “About Me” or similar section, looking for a string starting and ending with a delimiter `{}` – such as `{OOarFJ9wgqvTVgqHln51ftme+25/}`– then would decode the string to obtain the IP address and the port number of a C2 server.

A more sophisticated approach was discovered by Faou, Tartare, and Dupuy of the ESET research team [21] in the malware dubbed *PolyglotDuke*. They showed that certain malware uses specific and not commonly used character sets - in particular Unicode Katakana, Cherokee, or Kangxi - to mark the command string within public SM posts, such as in Instagram posts as illustrated in Fig. 1. While the appearance of a certain uncommon character set in public SM posts cannot be a single factor in identifying such SM C2 communications, it may be one of the factors to take into account.


Aimee Fleming
@aimeefleming25
Follow

This picture isᏌᏔᏎTᏀᏉRᎻᎯᏚᎯᏗDTᎹEMᎭᏬᏗWᏀᏇᏜᎤiᏇ ᏥMR Isn`t it!?

9:22 AM - 6 Apr 2017

Fig. 1 – *Example of a public post on Instagram used for* C2 *communication by PolyglotDuke malware, with specific Cherokee character set* [21].

Compagno et al. [30] have devised yet a different model. In their peer-to-peer botnet model called ELISA, the bot hides its exchange with a C2 server within the legitimate Facebook post of the ignorant owner of a bot: it attaches a set of non-printing characters, marking the start of the sequence by an invisible separator U+2063. To receive the command from the bot manager, on the other hand, the bot has to continuously parse Facebook posts from friends (since it is a peer-to-peer network) looking for an invisible separator, fetching, decoding, and decrypting the invisible characters that follow. Though the ELISA model has other weaknesses, such as fragile topology and dependence on infections among social contacts, the use of hidden characters makes C2 communications literally invisible (though not stealth).

It is very important to write the paper in clear English language, without spelling and grammar errors. For that purpose, send your paper to professional lecturer. We can recommend you professional lecturers from Proof-Reading-Service (http://www.proof-reading-service.com).

All references should be formatted using examples given in Section 6. References should contain only correct data: names of all authors, paper title, publication name, date of publication, paper pages, and other. Authors should cite more recent references in the subject area. This will give higher value to the paper.

4.2 Hiding C2 commands in shared images through steganography

Steganography is also used to convey C2 communication, hidden within publicly posted images. FireEye Threat Intelligence report [20] shows that *HAMMERTOSS* malware used a bogus Twitter account to provide the bot with a link, along with a hashtag. The link led to a document (stored in GitHub or on a compromised website) containing a photo, which had additional information embedded – the encrypted code – after the end of file marker. The hashtag provided through the Twitter post indicated where in the image is the hidden data (i.e. the offset in bytes into the image file), and the characters used for decryption (which were added to a hardcoded encryption key, to decrypt the message). Faou, Tartare and Dupuy of the ESET team [21], on the other hand, show that *RegDuke* malware downloaded images from Dropbox. Here, however, the LSB steganography technique was used, as illustrated in Fig. 2: the encrypted code was hidden in the two or three least significant bits (LSB) of the 8-bit values of red, green, and blue of each pixel (effectively obtaining 8 bits of hidden information in each pixel, without a noticeable change in the pixel colour). Interestingly, steganography is used not only to convey URLs of C2 servers (and related log-in credentials where needed), but also the payloads, such as Windows executable, DLL, or a PowerShell script.

Nagaraja et al. [9] experiment with using JPEG steganography based on the YASS scheme [31], to establish a stealth C2 infrastructure through the exchange of Facebook images posted by social contacts that are part of a peer-to-peer botnet (a concept similar to the one used by the ELISA model analysed earlier, but now with image steganography). This shows that, by the right choice of a steganography scheme that prevents detection of the hidden code in images, one can create probabilistically unobservable C2 communications – though this, in turn, may lower the stenographic capacity, and thus the C2 exchange capacity.

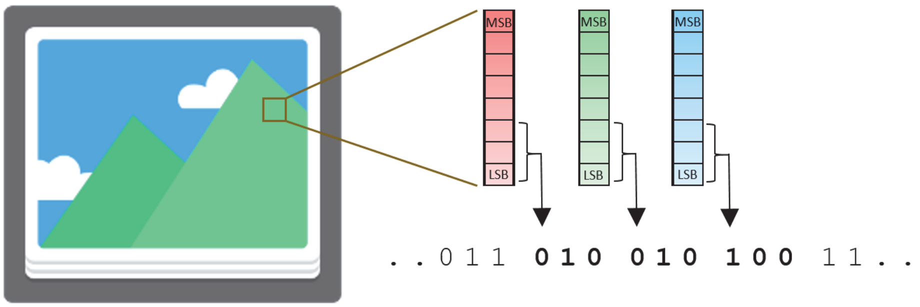


Fig. 2 – *LSB steganography technique for hiding* C2 *commands or locations in images.*

Pantic and Husain [32], on the other hand, showcased an unobservable C2 model based on linguistic steganography: they provided a methodology for hiding C2 messages within Twitter metadata. In their system, the message is not hidden within the text of the Twitter message, but rather created by manipulating the length of the tweet (allowing each tweet to safely convey 4 bits), and the name of the sending account (using Markov Chains for generating usernames); the length of the hidden message per tweet could be extended by manipulating also the date and time of each tweet, the frequency of letters, or the number of spaces within the Twitter message – all without making tweet posts suspicious.

4.3 Use of the public clouds as C2

There is a visible trend to use public cloud-based spaces not only to provide a link to C2 servers, but as C2 servers in some cases. Kaspersky team reports [33] that a hacker group dubbed CloudAtlas used free accounts for cloud storage at the legitimate Swedish public cloud provider CloudMe AB, to download encrypted instructions and the payload, and store the information stolen by bots in their APT operations. Similarly, malware titled *Minidionis* used OneDrive public cloud storage to fetch a payload, logging in with a hardcoded username and password [34].

Borazjani [35] reflected on the LOWBALL backdoor that abused the legitimate commercial public cloud service – Dropbox – to play the entire role of a C2 server. FireEye research [36] further clarified that, with the Dropbox API and the hardcoded access token, this backdoor communicated with the public cloud for all C2 operations: receiving commands, downloading payloads and other code, and the exfiltration of records and stolen files. Since the communication runs through HTTPS on a common TCP port 443, it is very difficult to distinguish it from the regular traffic with common detection tools.

Transferring files to the bots and back through public channels, using the regular HTTPS protocol and related ports, instead of through a dedicated C2 server, reduces the need for the infected devices to connect to a C2 server by querying its DNS (often with a suspicious domain name). This reduces the likelihood that defenders will discover C2 servers by following queries to suspicious domains.

4.4 Reducing dependence on bogus C2 accounts by using DGA

In order for a bot to know which public pages to look at, URLs of possible locations typically have to be hardcoded in the malware code (often in encrypted form), which is similar to when DNS locations of C2 are hardcoded in the malware. The malware then decrypts the hardcoded URL(s) and follow the link(s), to retrieve the content. The weak point of hardcoding is that, once the malware is discovered and analysed by defenders, bogus accounts linked to the hardcoded URLs can be disabled through regular domain take-down procedures. Wright and Bush [37] describe the Domain Generation Algorithm (DGA) as the emerging hackers' tactics that make it harder for defenders to identify and take-down domains. DGA is commonly used by malware to create thousands of possible domain URLs for a C2 server, allowing the malware to then check through them until it connects with the one that actually exists – registered by malware controllers and issues commands. Sood and Zeadally [38] have provided a useful taxonomy of DGA attacks, showing that what they called 'binary-based' DGAs – embedded in the malware code – is a dominant approach for creating a resilient C2 infrastructure for large-scale botnets: by registering timely new C2 domains, attackers make signature and blacklisting approaches to combating C2 domains useless.

Cui at al. [39] constructed an Andbot model with a DGA that creates numerous possible social media usernames – and, thus, URLs leading to possible accounts (like www.twitter.com/username) – according to predefined rules. Such a DGA was dubbed a User Generation Algorithm (UGA). Malware then regularly checks a large number of internally created URLs, until it finds the one that really exists – that leads to an existing account, created by malware controllers. Dong at al. [40] went a step further, combining UGA with Tor anonymous browser for bot's connection, commands posted through a QR on Twitter, and a Twitter search option, to create a resilient botnet that allows bot masters to switch easily Twitter accounts used for C2 communications. The only hardcoded element is DGA (ie. UGA) algorithm.

In practice, the ESET research team [41] discovered the use of username-creation DGA in *MiniDuke* malware: it created hundreds of possible accounts, such as:

```
0OIO2Te
2fjzjntOX
2RobTXBSV
32esNXlFYgpY
49c3auaBXar
4tzQT6FCl
5dVRylmE2b
```

yet only one account was actually working – `AA2ADcAOAA` (with a corresponding URL `www.twitter.com/AA2ADcAOAA`, as one of the many created by DGA).

FireEye analysis of the *HAMMERTOSS* malware [20] also shows the use of DGA to create daily Twitter URLs by appending and prepending three CRC32 values (cyclic redundancy check error-detecting code of a 32 bit-binary sequence), created based on the current date, around a common base name (like `1abBob52b`); then checking daily if the related account has been created by controllers, and if it tweeted with a specific hashtag. This technique allows malware controllers to easily change the Twitter account once the previous one was discovered by defenders: it simply needs to create another from the possible large list envisaged by DGA. Defenders, on the other hand, cannot block all the thousands of possible accounts that DGA has envisaged – not least because the vast majority of them do not even exist as accounts yet (and some may actually be used by legitimate users).

4.5 Reducing dependence on bogus C2 accounts by using posts of public figures

One of the main goals of attackers is to create a dynamic C2 communication channel – which can be quickly changed if compromised. In terms of social media, this means finding ways to make bots independent of a single SM account. Another option to avoid hardcoding URLs is to search for the particular public online content for specifically marked string – such as the comments posted under public posts of celebrities. Boutin and the ESET team [22] have discovered and analysed activities of the infamous *Turla* hacking group, which posted encrypted C2 commands as comments on a public Instagram post of Britney Spears. Malware at the infected devices was instructed to look at comments under such a specific Instagram post, compute a hash value of each comment on the post, and locate the one that matches the pre-defined hash (there 183) – in this case, the comment saying

```
#2hot make loved to her, uupss #Hot #X
```

Interestingly, the comment actually also contained the invisible 'zero width joiner' signs (U+200D), so it integrally read as:

```
smith2155<200d>#2hot ma<200d>ke love<200d>d to
<200d>her, <200d>uupss <200d>#Hot <200d>#X
```

Malware then used a regular expression

```
(?:\\u200d(?:#|@)(\\w)
```

over the comment text, to look for letters after signs #, @, or <200d>, and thereby compose a bit.ly shortened URL `http://bit.ly/2kdhuHX`. This URL finally led the malware to a C2 server, located at

`static.travelclothes.org/dolR_1ert.php`.

The more public and popular the post is, the less likely it is that the comment with C2 command will be identified and removed. Even if defenders recognise the location of the comment, and ultimately close the bogus account that posted the comment, attackers may open a new account and re-post the message, so that the defence action does not terminate the malware by obtaining a sample. In essence, C2 command is related to the content, not to the author (account). In addition, many trusted online media leave options for commenting on news and articles without the need for an account, making it easier for attackers to post and repost commands with C2 commands.

5 Stealth Model of Server-less C2 Infrastructure

One of the goals of attackers is to make C2 infrastructure more resilient and stealthy: to make it harder for defenders to identify and disable C2 (even if they get to a sample of the malware code from some of the bots), while keeping it simple for the attackers to change C2 parameters so that the infected bots can continue communicating. Most malware combines multiple approaches for C2 communication channels in order to achieve this. In this regard, the trends showcased in this paper could achieve their full potential when combined together, to strengthen a particular C2 infrastructure. It is, therefore, important to observe what one such model could look like, and discuss its strengths, and possible defence aspects.

5.1 Server-less model of C2 infrastructure

The stealth model of an advanced C2 infrastructure would combine various techniques to effectively eliminate the need of installing and maintaining own C2 servers; instead, it would entirely rely on public cloud-based services – both for issuing commands to bots, and for releasing additional code (such as payloads) and fetching files from bots (such as stolen files).

Wu et al. [42] suggested a server-less bot, dubbed SLBot, which uses the domains and APIs of various authenticated services, thus creating a dynamic C2 communication channel based only on public services. SLBot combines multiple applications for three communication sub-channels: addressing channel to point to an address where a command can be read by the bot, which combines a DGA (or, rather, a UGA) to create accounts on various SM, and steganography that contains links; command channel to issue commands to bots, which uses disposable email or online clipboards to temporarily store ciphertext versions of commands, and; upload channel for bot registration and upload of gathered data and files, which uses network disks and cloud notes. Their model, however, does not take into consideration the use of URL shortening services, and the posting of C2 communications through comments on public posts by regular users (even public figures) – the two components which increase the complexity of identifying and dismantling C2.

A model which combines the trends discussed above can be constructed to steal confidential data.

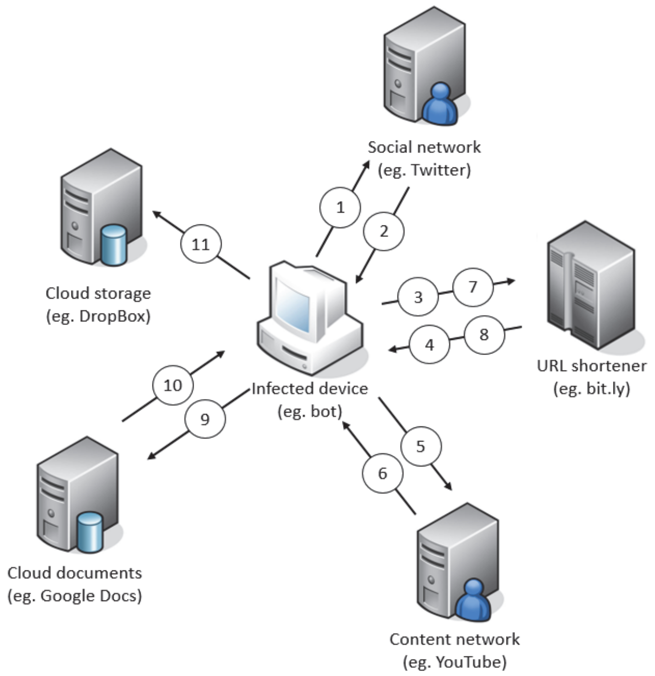


Fig. 3 – *Example of a communication scheme of a server-less* C2.

It may have the following communication steps as illustrated in Fig. 3:

1. The bot requests content from a bogus account on social network (e.g. Twitter);
2. The bot receives content from the social network, containing a shortened URL (e.g. bit.ly);
3. The b requests the full URL from the shortener service;
4. The bot receives the full URL from the shortener service, which leads to a post on an online content platform (e.g. YouTube);

5. The bot requests specific content from the content platform (e.g. a video with comments to it);
6. The bot receives content, including comments on a particular post, which it parses to obtain shortened URL (e.g. bit.ly)
7. The bot requests the full URL from the shortener service;
8. The bot receives the full URL from the shortener service, which leads to a shared document in the cloud (e.g. Google Doc);
9. The bot requests the content (such as a text and/or pictures) of the shared document;
10. The bot receives a picture with the code hidden through steganography, which it decodes to reveal the command it needs to execute (e.g. to log keystrokes), the location of the cloud storage (e.g. Dropbox) to place stolen data, and access parameters or a token
11. The bot connects to the cloud storage and uploads the stolen content (key log)

5.2 Model strengths

The major advantage of such a model is that C2 commands and files are hidden in plain sight within the public domain. In fact, this C2 infrastructure abuses trusted public services, which are generally secure, as well as popular ports (like Port 80 used for standard web traffic, as well as SSL port 443). Moreover, they piggyback on public posts by regular users, in particular, celebrities which genuinely receive thousands of comments; hiding C2 communications among these resembles a needle in a haystack for defenders, yet it makes it rather easy for bots to find them owing to the search for particular marks or hash.

Attackers can easily set up and maintain bogus social media accounts, which is more flexible and easier than setting up and running remote C2 servers. They can also easily switch to emerging or less protected online services, since malware can read commands from various accounts, on various services, and various online locations. In addition, to hide their communications, bots deploy encoding, encryption, and steganography techniques.

Therefore, defenders cannot easily distinguish C2 communications from legitimate social media posts by users; it is even harder to spot files or loaded images uploaded to the public cloud. Even if they detect and block some of the bogus social media accounts or cloud-based online service locations, due to a very flexible C2 infrastructure, attackers are in a position to replace the compromised components; bots do not really depend on the accounts, but rather on the content shared by whoever has shared it. Such a scenario is ideal for bot masters, as they do not need to re-infect devices with adjusted malware.

5.3 Defence aspects

Certainly, the model also has weaknesses that could be addressed by defenders. In order to detect C2 communications, defenders can look for suspicious behaviour of social media applications on the client side, bogus accounts on social media, and suspicious online content posted through social media. On the device side, the application-centric approach could complement the existing network-centric and host-centric intrusion detection systems [12]: it should analyse the patterns of use of particular applications (such as Twitter, Instagram, Google Docs, or Dropbox) on a particular machine, to detect possible suspicious activities (such as when the device parses through posts that use suspicious character sets). On the side of online service providers, tracking and dismantling bogus social media accounts is becoming increasingly efficient, in particular due to the use of machine learning algorithms [43]. Machine learning can also be deployed to track particular suspicious content posted on social media, based on the communication rules used by discovered malware samples (for instance by taking into account particular delimiters, invisible characters, or unusual character sets).

Defenders may also consider conducting pre-emptive measures that could disable C2 communications. For instance, social media platforms may implement steganography by default: they could slightly alter some LSBs of photos in (suspicious) posts, thereby spoiling possible C2 steganography attempts (effectively rendering C2 LSB-based messages un-decodable); such a slight change in pixels would still not be noticeable by regular end-users. Similar pre-emptive changes in the textual content of (suspicious) posts – such as randomly embedding invisible characters (like zero width joiner U+200D), or altering character sets of some letters without changing their visual appearance – would alter the possible C2 code, rendering it un-decodable.

6 Conclusion

Command and control (C2) infrastructure, which enables communications between the infected devices (bots) and attackers has become a critical component of today’s cyber-attacks. Conventional models are based on remote servers set up by attackers, which are contacted by bots through specific URL addresses. As defenders have advanced with techniques of identifying and dismantling the bogus domains used for C2, attackers are turning to stealthier models.

In this paper, we have provided a review of academic and professional literature that addresses the emerging trends related to the use of C2 infrastructure based on social media, including online social networks, and public cloud storage. We have compared different research attempts to create malware and botnets that use C2 based on social media, and cross-referenced them with the reports about real-world malware, as analysed by the lead cybersecurity corporations. In particular, we have identified five specific trends that attackers use: hiding communications within the text-based SM posts (using string delimiters or specific character sets); hiding communications through image and linguistic steganography; using public cloud storage for unobservable exchange of communications and upload of stolen files; using Domain Generation Algorithms, and in particular the Username Generation Algorithm, and; conveying C2 messages through comments on public SM posts, in particular those of public figures. By analysing the emerging trends, the paper aims to improve defence capacities to better spot and dismantle such C2 infrastructures.

We have shown that C2 commands are hidden within posts shared by the bogus social media accounts, such as Twitter or Facebook, as well as in comments to public posts of regular users (even celebrities), such as Instagram, YouTube, or professional forums, which makes them hard to identify; at the same time, such an approach makes it easier for attackers to change C2 parameters easily – such as by switching to new accounts – without a need to re-infect bots. Attackers also deploy steganography techniques to hide code of commands – and even payloads – within photos which they share through social media, or post to public cloud spaces like Google Documents. To decrease the reliance on own C2 servers, which may be more easily detected, attackers are increasingly using cloud storage, like Dropbox, GitHub or other, to store malware code, or collect files stolen by bots. Advanced techniques, like Domain Generation Algorithm, are deployed to reduce the likelihood of bogus social accounts, used as C2, to be discovered and closed. All the operations rely heavily on the use of encryption, encoding, and steganography to make their commands unreadable.

The analysed trends show a tendency of creating a server-less C2 infrastructure, by basing it entirely on public cloud-based services. To explore the potentials for a stealth C2 infrastructure if several of the observed techniques are combined, we have suggested and discussed a possible model of complex C2 infrastructure based on social media. We have observed that, since C2 communications are thus hidden in plain sight, i.e. within an abundance of legitimate public posts, defenders are faced with 'a needle in a haystack' problem to identify C2 sources and messages. Even when some accounts are compromised, attackers can easily switch to other online accounts or even different online services. This makes such an infrastructure stealthier and more resilient to identification and take-downs.

Finally, we have discussed possible defence approaches – for addressing the observed trends, and for identifying server-less C2 infrastructure. In this regard, further research is needed to look into the ways to attack such C2 infrastructure – in particular what can be done on the level of online service providers to pre-empt and respond to the presence of C2 in their services. Possible actions include reactive measures, such as using machine learning algorithms for recognising bogus accounts and C2 messages, as well as proactive measures, like smartly altering the online content (through steganography or embedding invisible characters) to render C2 communications un-decodable. This would require coordinated efforts by lead online platform operators, most likely with the support of – or under pressure from– the governments and regulators. Such joint actions in combating C2 infrastructure could piggyback on the growing global cooperation among lead social media companies and governments to recognise and remove extremist content online.

7 References

[1] E. M. Hutchins, M. J. Cloppert, R. M. Amin: Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains, Leading Issues in Information Warfare & Security Research, Vol. 1, No. 1, April 2011, pp. 80 – 106.

[2] MITRE: ATT&CK Matrix for Enterprise, 2019, Available at: https://attack.mitre.org/

[3] V. J. Radunovic: DDoS - Available Weapon of Mass Disruption, Proceedings of the 21st Telecommunications Forum Telfor (TELFOR), Belgrade, Serbia, November 2013, pp. 5 – 9.

[4] M. Ussath, D. Jaeger, F. Cheng, C. Meinel: Advanced Persistent Threats: Behind the Scenes, Proceedings of the Annual Conference on Information Science and Systems (CISS), Princeton, USA, March 2016, pp. 181 – 186.

[5] C. J. Dietrich, C. Rossow, F. C. Freiling, H. Bos, M. van Steen, N. Pohlmann: On Botnets that Use DNS for Command and Control, Proceedings of the 7th European Conference on Computer Network Defense, Gothenburg, Sweden, September 2011, pp. 9 – 16.

[6] J. Nazario: Twitter Based Botnet Command and Control, 2009. Available at: https://www.secureworks.com/blog/twitter-based-botnet-command-and-control

[7] F. Brezo, J. G. de la Puerta, I. Santos, D. Barroso, P. G. Bringas: C&C Techniques in Botnet Development, Proceedings of the International Joint Conference CISIS'12-ICEUTE'12-SOCO'12, Ostrava, Czech Republic, September 2012, pp. 97 – 108.

[8] I. K. Makkar, F. Di Troia, C. A. Visaggio, T. H. Austin, M. Stamp: SocioBot: A Twitter-Based Botnet, International Journal of Security and Networks, Vol. 12, No. 1, March 2017, pp. 1 – 12.

[9] S. Nagaraja, A. Houmansadr, P. Piyawongwisal, V. Singh, P. Agarwal, N. Borisov: Stegobot: A Covert Social Network Botnet, Proceedings of the International Workshop on Information Hiding, Prague, Czech Republic, May 2011, pp. pp 299 – 313.

[10] I. Ghafir, J. Svoboda, V. Prenosil: A Survey on Botnet Command and Control Traffic Detection, International Journal of Advances in Computer Networks and Its Security, Vol. 5, No. 2, October 2015, pp. 75 – 80.

[11] A. K. Sood: Exploiting Fundamental Weaknesses in Botnet Command and Control (C&C) Panels, Proceedings of the BlackHat Security Conference, Las Vegas, USA, August 2014, pp. 1 – 32.

[12] E. J. Kartaltepe, J. A. Morales, S. Xu, R. Sandhu: Social Network-Based Botnet Command-and-Control: Emerging Threats and Countermeasures, Proceedings of the International Conference on Applied Cryptography and Network Security, Beijing, China, June 2010, pp. 511 – 528.

[13] Y. Ji, Y. He, X. Jiang, J. Cao, Q. Li: Combating the Evasion Mechanisms of Social Bots, Computers & Security, Vol. 58, No. C, May 2016, pp. 230 – 249.

[14] P. Mackenzie: WannaCry Aftershock, SophosLabs, 2017, pp. 1-14, Available at: https://www.sophos.com/en-us/medialibrary/PDFs/technical-papers/WannaCry-Aftershock.pdf

[15] F. Skopik, T. Pahi: Under False Flag: Using Technical Artifacts for Cyber Attack Attribution, Cybersecurity, Vol. 3, No. 1, March 2020, pp. 8:1 – 8:20.

[16] P. Marko, P. Vilhan: Efficient Detection of Malicious Nodes based on DNS and Statistical Methods, Proceedings of the IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, Slovakia, January 2012, pp. 227 – 230.

[17] A. Singh, A. H. Toderici, K. Ross, M. Stamp: Social Networking for Botnet Command and Control, International Journal of Computer Network and Information Security, Vol. 5, No. 6, May 2013, pp. 11 – 17.

[18] J. Göbel, T. Holz: Rishi: Identify Bot Contaminated Hosts by IRC Nickname Evaluation, Proceedings of the 1st Conference on First Workshop on Hot Topics in Understanding Botnets (HotBots'07), Cambridge, USA, April 2007, pp. 1 – 12.

[19] A. Lehtiö: C&C-As-A-Service: Abusing Third-Party Web Services as C&C Channels, Proceedings of the 25th Virus Bulletin International Conference, Prague, Czech Republic, September 2015, pp. 316 – 321.

[20] FireEye Threat Intelligence: HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group, Special Report, July 2015, Available at: https://www.fireeye.com/blog/threat-research/2015/07/hammertoss_stealthy.html

[21] M. Faou, M. Tartare, T. Dupuy: Operation Ghost: The Dukes aren't Back - They Never Left, ESET Research White papers, October 2019, pp. 1 – 40.

[22] J.- I. Boutin: Turla's Watering Hole Campaign: An Updated Firefox Extension Abusing Instagram, WeLiveSecurity by ESET, June 2017, Available at:

https://www.welivesecurity.com/2017/06/06/turlas-watering-hole-campaign-updated-firefox-extension-abusing-instagram/

[23] D. Erwin: iWorm Botnet Uses Reddit as Command and Control Center, Intego Mac Security Blog, Intego, October 2014, Available at: https://support.intego.com/hc/en-us/articles/207113608-iWorm-Botnet-Uses-Reddit-as-Command-and-Control-Center

[24] J. Gardiner, M. Cova, S. Nagaraja: Command & Control: Understanding, Denying and Detecting, February 2014, pp. 1 – 38.

[25] S. Miller, P. Smith: Rise of Legitimate Services for Backdoor Command and Control, Anomali, 2017, 1 – 58.

[26] Evernote as Command-and-Control Server, Security Intelligence Blog by TrendMicro, Mart 2013, Available at: https://blog.trendmicro.com/trendlabs-security-intelligence/backdoor-uses-evernote-as-command-and-control-server/

[27] R. Dela Paz: Malware Uses Sendspace to Store Stolen Documents, Security Intelligence Blog by TrendMicro, February 2012, Available at: https://blog.trendmicro.com/trendlabs-security-intelligence/malware-uses-sendspace-to-store-stolen-documents/

[28] J. Chen: Blackgear Cyberespionage Campaign Resurfaces, Abuses Social Media for C&C Communication, Security Intelligence Blog by TrendMicro, July 2018, Available at: https://blog.trendmicro.com/trendlabs-security-intelligence/blackgear-cyberespionage-campaign-resurfaces-abuses-social-media-for-cc-communication/

[29] J. Does: Controlling a Computer Network and Thereby Injuring Plaintiff and Its Customers, In the United States District Court for the Eastern District of Virginia, by Microsoft Corporation, October 2017, pp. 1 – 33.

[30] A. Compagno, M. Conti, D. Lain, G. Lovisotto, L. Mancini: Boten ELISA: A Novel Approach for Botnet C&C in Online Social Networks, Proceedings of the IEEE Conference on Communications and Network Security (CNS), Florence, Italy, September 2015, pp. 74 – 82.

[31] K. Solanki, A. Sarkar, B. S. Manjunath: YASS: Yet Another Steganographic Scheme that Resists Blind Steganalysis, Proceedings of the International Workshop on Information Hiding, Saint-Malo, France, June 2007, pp. 16 – 31.

[32] N. Pantic, M. I. Husain: Covert Botnet Command and Control Using Twitter, Proceedings of the 31st Annual Computer Security Applications Conference (ACSAC 2015), New York, USA, December 2015, pp.171 – 180.

[33] GReAT: Cloud Atlas: RedOctober APT is Back in Style, Kaspersky APT Reports, December 2014, Available at: https://securelist.com/cloud-atlas-redoctober-apt-is-back-in-style/68083/

[34] S. Lozhkin: Minidionis – One More APT with a Usage of Cloud Drives, Kaspersky APT Reports, July 2015, Available at: https://securelist.com/minidionis-one-more-apt-with-a-usage-of-cloud-drives/71443/

[35] P. N. Borazjani: Security Issues in Cloud Computing, Proceedings of the 12th International Conference (GPC 2017), Cetara, Italy, May 2017, pp. 800 – 811.

[36] FireEye Threat Intelligence: China-Based Cyber Threat Group Uses Dropbox for Malware Communications and Targets Hong Kong Media Outlets, December 2015, Available at: https://www.fireeye.com/blog/threat-research/2015/11/china-based-threat.html

[37] E. Wright: Hacker Tactics - Part 1: Domain Generation Algorithms, Anomali, August 2017, Available at: https://www.anomali.com/blog/hacker-tactics-part-1-domain-generation-algorithms

[38] A. K. Sood, S. Zeadally: A Taxonomy of Domain-Generation Algorithms, IEEE Security & Privacy, Vol. 14, No. 4, July 2016, pp. 46 – 53.

[39] X. Cui, B. Fang, L. Yin, X. Liu, T. Zang: Andbot: Towards Advanced Mobile Botnets, Proceedings of the 4th USENIX Workshop on Large-Scale Exploits and Emergent Threats, Boston, USA, March 2011, pp. 1 – 7.

[40] Y. Dong, J. Dai, X. Sun: A Mobile Botnet that Meets Up at Twitter, Proceedings of the International Conference on Security and Privacy in Communication Systems - Security and Privacy in Communication Networks, Singapore, Singapore, August 2018, pp. 3 – 21.

[41] ESET Research: Miniduke Still Duking it Out, WeLiveSecurity by ESET, May 2014 Available at: https://www.welivesecurity.com/2014/05/20/miniduke-still-duking/

[42] D. Wu, B. Fang, J. Yin, F. Zhang, X. Cui: SLBot: A Serverless Botnet Based on Service Flux, Proceedings of the IEEE 3rd International Conference on Data Science in Cyberspace (DSC), Guangzhou, China, June 2018, pp. 181 – 188.

[43] I. Aydin, M. Sevi, M. U. Salur: Detection of Fake Twitter Accounts with Machine Learning Algorithms, Proceedings of the International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, September 2018, pp. 1 – 4.