BlindEagle Leveraging BlotchyQuasar | ThreatLabz

By Gaetano Pellegrino

Published: 2024-09-05 · Archived: 2026-04-05 22:24:42 UTC

Technical Analysis

Overview

A BlindEagle attack chain typically originates with a phishing email that contains a PDF attachment and a URL that points to a ZIP archive file. The PDF attachment contains the same URL as the one provided in the email body. In other words, the ZIP file can be either downloaded from the PDF or directly from the email.

Upon clicking the URL (in either the email body or PDF), the victim downloads a ZIP archive from a Google Drive folder. This specific folder is under the ownership of a compromised account belonging to a regional government organization in Colombia. The ZIP archive contains a .NET BlotchyQuasar executable.

The figure below provides for a high-level overview of the attack chain.

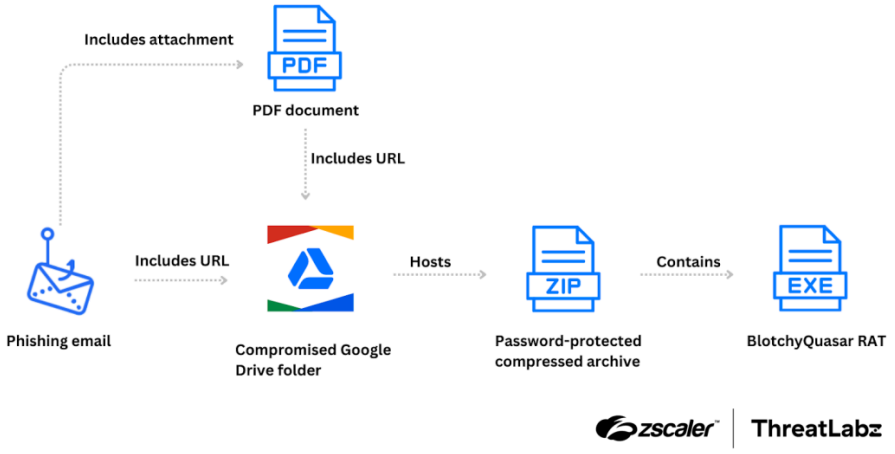

Figure 1: A high-level overview of a BlindEagle attack chain, where the initial phishing email includes a download URL for a password-protected compressed archive and the final payload is a packed BlotchyQuasar sample.

Phishing email as initial vector

In the phishing email, the threat actor impersonated the *Dirección de Impuestos y Aduanas Nacionales (DIAN)*, which is the Colombian National Tax and Customs Authority. The lure used by BlindEagle involved sending a notification to the victim, claiming to be a seizure order due to outstanding tax payments. This is intended to create a sense of urgency and pressure the victim into taking immediate action. Our observations indicate that a substantial number of the targeted individuals are employees within the Colombian insurance industry.

The figure below shows the phishing email, which includes the PDF and download URL, spoofing the Colombian tax authority.

Figure 2: Example BlindEagle phishing email spoofing DIAN with a PDF attachment and malicious link in the email body.

The download URL directs the victim to a password-protected ZIP archive. The password necessary to open the archive is provided within the email body.

This ZIP archive is hosted on a Google Drive folder, which is associated with a compromised Gmail account owned by a government organization with a ".gov.co" top-level domain.

Based on analysis of the phishing email's metadata, the threat actor likely sent the emails from their own infrastructure. Specifically, the first header received in the email indicates that the message originated from the IP address `69.167.8.118`, which is associated with Powerhouse Management VPN. Powerhouse Management is a VPN service known to be utilized by BlindEagle to obfuscate the true source of their malicious activities and acquire IP addresses that are geographically close to their intended targets.

BlotchyQuasar

BlotchyQuasar is a powerful RAT that possesses a wide range of capabilities. It can log keystrokes, execute shell commands, and perform various other functions. Since BlotchyQuasar is a variant of QuasarRAT, we will not delve into a detailed analysis of its functionalities. Instead, in the following sections, we will concentrate on specific aspects that have not been extensively covered in previous publications.

Loader

As shown in the figure below, BlotchyQuasar is concealed within multiple layers of protection. Each layer consists of a .NET executable that has been safeguarded using either commercial or open-source obfuscators like DeepSea or ConfuserEx. These obfuscators are employed to make the code more complex and challenging to analyze, hindering reverse engineering attempts.

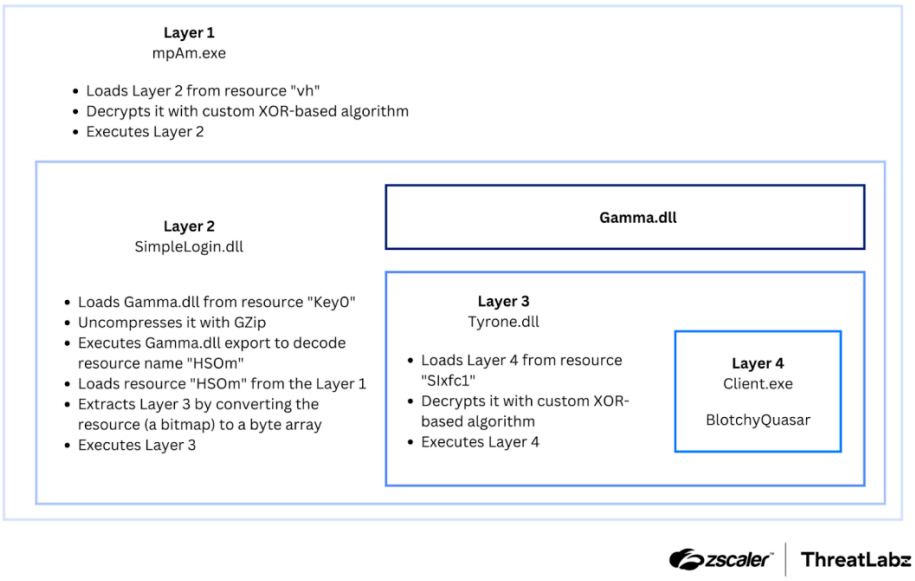

Figure 3: Nested structure of the BlotchyQuasar sample.

Layer 1 is the outer executable file that is contained within the ZIP archive. It decrypts the Layer 2 data that is contained in a resource named `vh` by utilizing a custom XOR-based algorithm. Layer 2 consists of a DLL with the name `SimpleLogin.dll`. When executed, `SimpleLogin.dll` loads and extracts the contents of a GZip-compressed resource named `key0`. Within this resource lies another DLL, `Gamma.dll`, which provides a utility for converting integers to Unicode characters. This utility is used to compose the name of a resource within Layer 1, which is subsequently loaded by `SimpleLogin.dll`. This resource is named `HSOm` and is stored as a bitmap image that undergoes a transformation process. This transformation involves discarding the last 150 rows and the last 150 columns of the image. Additionally, the row and column pixel coordinates are inverted. The figure below shows the bitmap when rendered.

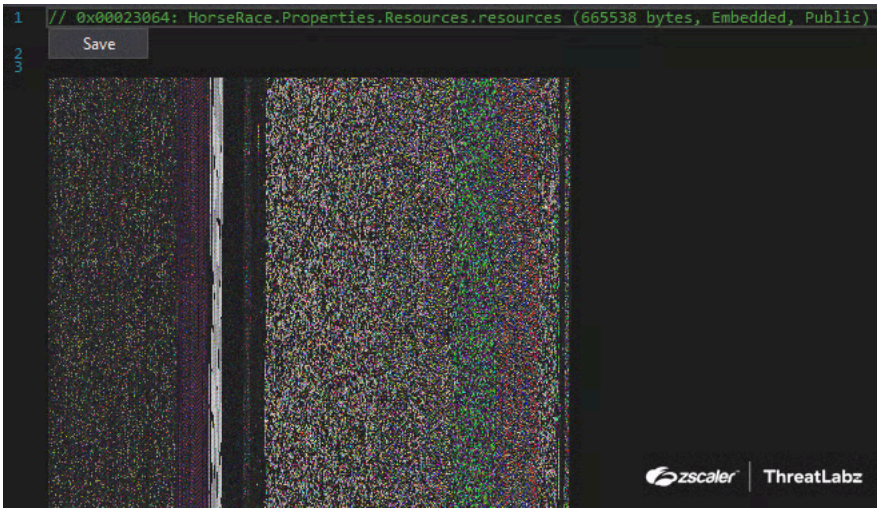

Figure 4: The resource `HSOm` rendered as a bitmap containing the Layer 3 data.

By extracting the ARGB coordinates from each pixel, another DLL named `Tyrone.dll` is obtained. This DLL represents Layer 3, which decrypts the final payload by loading a resource named `SIxfc1` and applying a custom XOR-based algorithm. This produces an executable file named `Client.exe`, which is a BlotchyQuasar malware sample.

Obtaining the C2

The installation steps completed by BlotchyQuasar are discussed in a previous publication. In the sample we analyzed, we observed a similar process. However, the procedure employed to obtain the command-and-control (C2) domain has not been previously analyzed.

When BlotchyQuasar is executed, the C2 server location is retrieved from Pastebin. The specific URL used to fetch the paste is `hXXps://pastebin[.]com/raw/XAfmb6xp`. The content of the paste is an encrypted string, as shown in the example below. (The relevant part is highlighted in bold and separated by the two "`¡`" symbols.)

GNNwsubynrt5oCZ+pAP97K9Sizq1eRn8XQQ8yxktdrbYQL263pZf+aQwkap8YEa09tg1w69qsZYEwGWF482CW3WBNKOJESQBz8IXYNzbbf+jrHU

This encrypted string is divided into three parts, with the symbol "¡" serving as a separator. Of particular interest is the middle part, enclosed by the separators (i.e., `ClllOSeGR/pSE1OqzWOtN5zIKVp5TOLPJ1rBUGNg5fA=`).This string is Base64 decoded and decrypted using standard 3DES encryption in ECB mode with PKCS7 padding. The 3DES key used is derived from the MD5 hash of the string `qualityinfosolutions`. In the provided example, the resulting C2 domain is `edificiobaldeares.linkpc[.]net`. The C2 communication for this sample leveraged the hardcoded port 9057.

Monitoring the consumption of banking & payment services

BlotchyQuasar implements a multitude of features, including the ability to monitor a victim's interactions with specific banking and payment services. In order to identify such events, the malware examines the title of each newly opened window. If the window title contains certain predefined strings associated with the targeted services, BlotchyQuasar logs a reference to indicate the occurrence of the interaction.

The figure below shows an example log collected with references to several banking and payment services. In the example provided, websites for Banco Coomeva, Banco of Machala, and PayPal services were accessed. The log, in this case, is a simple XML document that contains all the references within elements labeled as `NameCliente`. This log file, named `settings.xml`, is stored on the disk within the startup folder of the compromised system.

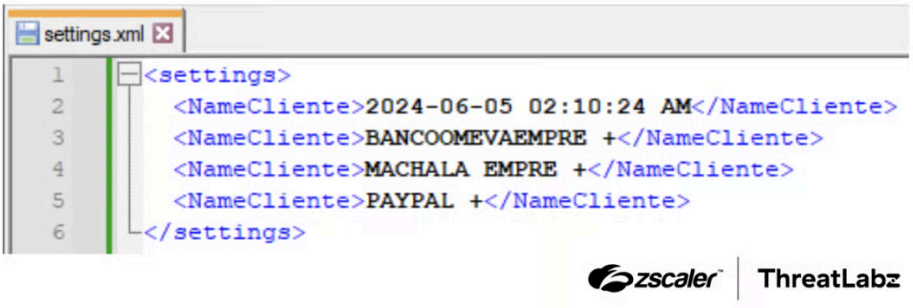

Figure 5: Example BlotchyQuasar log containing references to the victim's interaction with specific banking and payment service providers.

The table below lists the organizations that BlotchyQuasar targets. Since the list mainly includes Colombian and Ecuadorian banks, the malware was most likely designed to target individuals in those countries.

Organization	**Location**
BBVA	Global
Banco AV Villas	Colombia
Banco Bolivariano	Ecuador
Banco Caja Social	Colombia
Banco Coomeva	Colombia
Banco Davivenda	Colombia
Banco Guayaquil	Ecuador
Banco Internacional	Ecuador
Banco Pichincha	Ecuador

Organization	Location
Banco Popular Colombia	Colombia
Banco de Bogotá	Colombia
Banco de Machala	Ecuador
Banco de la Producción	Ecuador
Banco del Austro	Ecuador
Banco del Pacifico	Ecuador
Bancolombia	Colombia
PayPal	Global
Scotiabank Colpatria	Colombia
TransUnion	Global

Table 1: List of banking and payment service providers targeted by BlotchyQuasar.

Keylogging

BlotchyQuasar provides keylogging functionality, with the keylogging module set to flush logs every 15 seconds. These logs are stored in the `%APPDATA%\GPrets` directory with the filename format `MM-dd-yyyy` (e.g., `06-18-2024`). The log file is structured according to the figure below, which details the captured keylogging data.

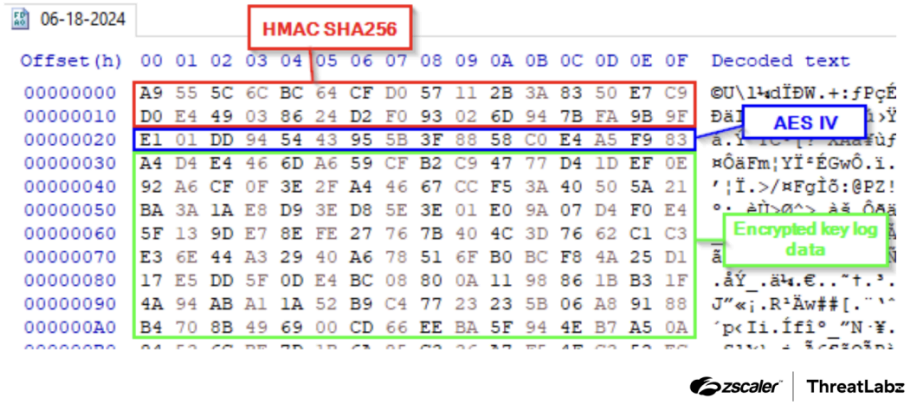

Figure 6: Structure of a BlotchyQuasar key log.

The initial 32 bytes of the encrypted log file comprise an HMAC SHA256 hash of the remaining content that is used as an integrity check. The subsequent 16 bytes store an AES initialization vector (IV) that is randomly generated per file. The AES key is hardcoded within the malware's configuration class. In the sample analyzed by ThreatLabz, the AES key was represented by the Base64-encoded string `1WvgEMPjdwfqIMeM9MclyQ==`. BlotchyQuasar uses AES in CBC mode (Cipher Block Chaining) with PKCS7 padding. The remaining portion of the file following the IV encompasses the encrypted log data itself.

A Python implementation of the BlotchyQuasar keylogging decryption routine is shown in the code sample below.

```
 from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

def decrypt(log: bytes, key: bytes) -> bytes:
    encrypted_payload = log[48:]
    iv = log[32:48]
    cypher = AES.new(
        key,
        AES.MODE_CBC,
        iv
    )
    decrypted_payload = cypher.decrypt(encrypted_payload)
    block_size = cypher.block_size
    decrypted_payload = unpad(decrypted_payload, block_size, "pkcs7")
    return decrypted_payload
```

As illustrated in the figure below, the decrypted logs are stored in HTML format.

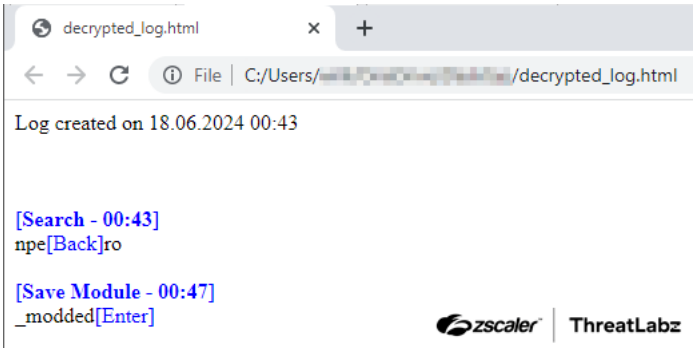

Figure 7: Example decrypted key log data created by BlotchyQuasar.

Stealing capabilities

BlotchyQuasar targets the browser and FTP client applications shown in the table below.

Application	Type	Targeted Data
Chrome	Browser	• Saved passwords • Cookies
Chromium	Browser	• Saved passwords • Cookies
Internet Explorer	Browser	• URL history
Firefox	Browser	• Saved passwords • Cookies
Opera	Browser	• Saved passwords • Cookies
Yandex	Browser	• Saved passwords • Cookies
FileZilla	FTP client	• Saved passwords

Application	Type	Targeted Data
WinSCP	FTP client	• Saved passwords

Table 2: Applications targeted by BlotchyQuasar for information-stealing purposes.

Infrastructure

BlotchyQuasar accesses Pastebin to retrieve the current C2 domain. The structure of the Pastebin content and the decryption procedure is unique, which enabled us to identify additional pastes consumed by BlotchyQuasar samples. By successfully decrypting these pastes, we uncovered three more C2 domains:

- equipo.linkpc[.]net
- perfect5.publicvm[.]com
- perfect8.publicvm[.]com

All those domains share a few characteristics:

- First, these domains are extensions of second-level domains (SLDs) associated with Dynamic DNS service providers. Second, they exhibit a consistent pattern in their resolution history. Specifically, they predominantly resolve to IP addresses that belong to two primary sets. The first set comprises nodes associated with specific VPN services, namely Powerhouse Management, PrivateVPN, and ParadiseNetworks.
- The second set comprises IP addresses associated with specific Colombian internet service providers (ISPs), namely Colombia Movil, Telmex Colombia, and Tigo. These IP addresses are likely indicative of compromised routers. This information aligns with publicly disclosed findings about the infrastructure under the control of the BlindEagle threat actor.

By shifting our focus towards resolving IP addresses, we gained further insights into the infrastructure underpinning operations similar to the one described in this blog. We discovered additional domains that exhibited similar characteristics. While we lack sufficient information to definitively establish that these domains are controlled by the same threat actor, they continue to pose threats to individuals and organizations. Notably, these domains have been utilized, and may still be in use, as C2 servers for various commodity malware families, including njRAT, QuasarRAT, RevengeRAT, and others. It is crucial to remain vigilant as these domains could potentially be employed for malicious activities in the future.

As an example, the table below displays the date of first submission on VirusTotal of various QuasarRAT samples communicating with the domain edificiobaldeares.linkpc[.]net. This domain has been utilized as a C2 server since July 2022 and active until March 2024. Since a similar pattern repeats in other domains, we strongly recommend blocking them.

First Submission Date	MD5	Malware Family
18-07-2022	a73057824a65a5ac982e298a80febf61	QuasarRAT
21-07-2022	bd4505316254f00329431fb8b2888643	QuasarRAT
22-07-2022	d2fc372302180fbabe18c425aa4a0a72	QuasarRAT
22-07-2022	c944cb638364c74431bf1dbe7dd329ff	QuasarRAT
24-07-2022	64e6ad512eff12e971efdd8979086c5c	QuasarRAT
26-07-2022	a1f5091ad4e12f922a8e760e0980ab66	QuasarRAT
29-07-2022	ad578125b337168c976ff5e7e1b190b8	QuasarRAT
01-08-2022	e21b4c9d9da81deea2381f9b988b0f99	QuasarRAT

First Submission Date	MD5	Malware Family
04-08-2022	07f661aeeb0774f0cb84b0a5e970c2a5	QuasarRAT
09-08-2022	c4a946903cc9e9a84763ac1731cdd7dd	QuasarRAT
11-08-2022	75a40cc019c39e3c2800fb2fe5aba1d3	QuasarRAT
12-08-2022	0fa40788b75896a452398b6a49cc62b6	QuasarRAT
15-08-2022	59a4f7aed1e3a0718592fb536e987a1d	QuasarRAT
16-08-2022	456211df625002df378cf0f4af9d1a6f	QuasarRAT
17-08-2022	0f35306ad4fede9a9ba0276a5e788138	QuasarRAT
19-08-2022	6044b126afb86682b4a3440e2924c079	QuasarRAT
19-08-2022	b432e8ff5797fbaf5808d95d46524647	QuasarRAT
20-08-2022	a31ff54f33ced7b4180f87afb18185a7	QuasarRAT
20-08-2022	e3239ac16c6fe9c99d6fac0867121a88	QuasarRAT
07-07-2023	2784a9fc64d244b14e7d8e4d03f41265	QuasarRAT
06-03-2024	3125ae6b1462b0b48dc06bc47d8ddbc7	QuasarRAT

Table 3: The most recent recorded interactions between various QuasarRAT malware samples and the domain `edificiobaldeares.linkpc[.]net`.
