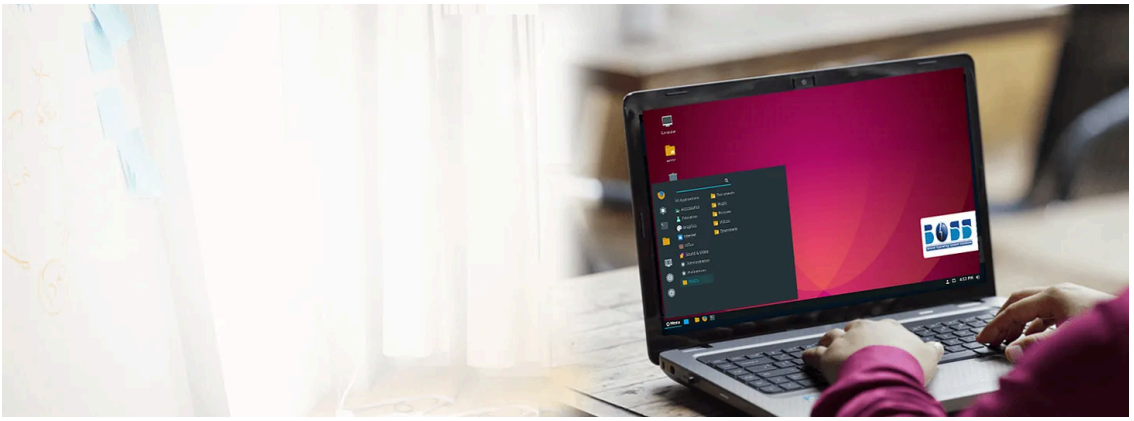
Reversing DISGOMOJI with Malcat like a BOSS

By RJM

Published: 2022-02-12 · Archived: 2026-04-05 22:53:07 UTC



Cover: BOSS is an Indian-based Linux distribution used by the Indian government

Disclaimer: The views, methods, and opinions expressed at Anchored Narratives are the author's and do not necessarily reflect my employer's official policy or position.

It has been a while since I last wrote an anchored narrative on my substack; the reason for it is that I have been quite busy with my Volexity job. The past year, I was part of the investigation team that led to the discovery of the Ivanti zero-day, and a bit later, the investigation led to the zero-day detection of the Palo Alto GlobalProtect firewalls. In April 20204, ForensicFocus interviewed me about the importance of memory forensics. In June, I wrote a peer-reviewed article for the IT-Auditor organization, NOREA, about the significance of memory forensics for auditors in determining the maturity of the CyberDefence capabilities of organizations. So, it was a pretty busy time.

But since the beginning of February 2024, I have found a new tool called Malcat, which has made my job much more convenient. It's a binary analysis software for threat analysts and reversers. It provides all the features I need for my daily work, including analyzing malware and creating Yara rules or other detection opportunities. I favor Malcat over disassemblers like IDA or Ghidra because it has many built-in features and fits me like a glove.

Recently, our threat intelligence team published an excellent investigation into a Pakistani threat actor that successfully compromised multiple Linux BOSS systems of the Indian government. BOSS stands for Bharat Operating System Solutions and is a custom version of Linux widely used by the Indian government. The investigation was fascinating, as the Linux GO malware, dubbed DISGOMOJI, listens for new messages in the command channel on the Discord server as the C2 server. Communication with it takes place using an emoji-based protocol. The attacker sends commands to the malware by sending emojis to the command channel, with additional parameters following the emoji where applicable.

This anchored narrative briefly demonstrates some Malcat features on a Go Linux malware sample that these threat actors used. Why this sample? It is a Linux sample and not a Windows sample, highlighting some of the

features of Malcat discussed in this article. This anchored narrative is based on a lightning talk about Malcat and DISGOMOJI during a threat meetup at Google/Mandiant I presented in Amsterdam on July 18, 2024. I asked a filled room of threat analysts/reversers (70 or 80 folks) if they were familiar with Malcat, but nobody knew about this tool, and that has to change. Are you interested in the tool's capabilities? Let’s go!

Disclaimer: I have no financial interest in the tool; I am just a Malcat fan!

The threat actor that developed DISGOMOJI is currently being tracked as UTA0137 and seen as a Pakistani threat actor. After compromising multiple BOSS systems, the threat actors used an old privilege escalation exploit called DirtyPipe to gain root access. NIST tracked the vulnerability under CVE-2022-0847. Once exploited, they would exfiltrate relevant information from the BOSS systems via the DISGOMOJI malware. From an outside perspective, it appears that the cyber maturity of the Indian government has some opportunities, namely vulnerability management, but also the unrestricted access of BOSS users who can connect to Discord services from their network.

Screenshot 1: The logo of Malcat

According to its website, Malcat is a feature-rich hexadecimal editor/disassembler for Windows and Linux targeted at IT security professionals. For me, that does not cut it. It is a complete binary analysis platform that can be used in an isolated network without the Internet. Malcat runs natively on Linux and Windows, where its reversing power lies. It’s not that great at reversing macOS binaries. One of the things I enjoy is the tremendous amount of binary file support. For example, when you want to reverse a malicious Microsoft installer package, it is natively supported, and you can navigate to the malicious stuff immediately. But here it comes: Malcat has full support from Yara. This means you can use your Yara rule base to determine if rules already cover a piece of malware or if you need to write an intelligent rule to cover the new sample.

Malcat also has built-in Python support, handy for writing a config extractor for certain malware families. A great example of Qakbot can be found here.

Recently, a new feature called Kesakode has been implemented, a remote hash lookup service that checks unpacked malware samples against an indexed library of known malware samples from Malcat and Malpedia. The following information is listed about it on the Malcat website:

`“a remote hash lookup service exclusive to Malcat users and tightly integrated inside Malcat's UI. It can be used to match known functions, strings and constant sets against a database of known clean, malware and library files. The Kesakode service can be used in various situation, such as:`

- `identify unpacked (e.g. a memory/sandbox dump) malware samples`
- `show similarities shared between malware families`
- `assist in the creation of better Yara rules`
- `speed up reverse engineering by identifying know libraries / runtime code“`

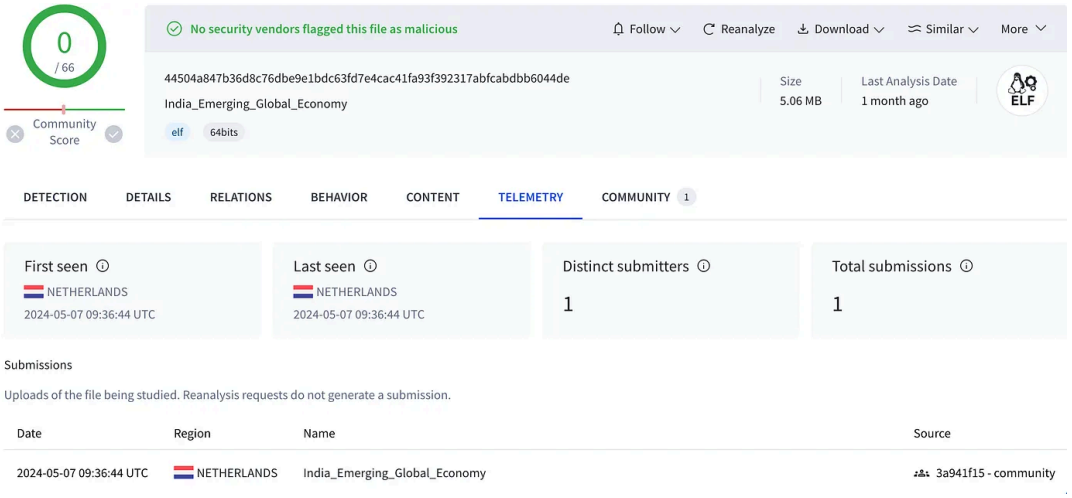
One of the other things I often use in Malcat is the ability to look up pieces of data (bytes or string patterns) in a defined corpus with your own or public repositories of malware samples. I use the samples I encounter in my daily casework and the VXunderground dataset.

Malcat also has code decompilation and built-in support for threat intelligence providers like Intezer, Malware Bazar, VirusTotal, and others.

Malcat has many features, but we will only focus on Transforms today. Transforms is a built-in CyberChef functionality where you can stack recipes and manipulate or convert data to strings.

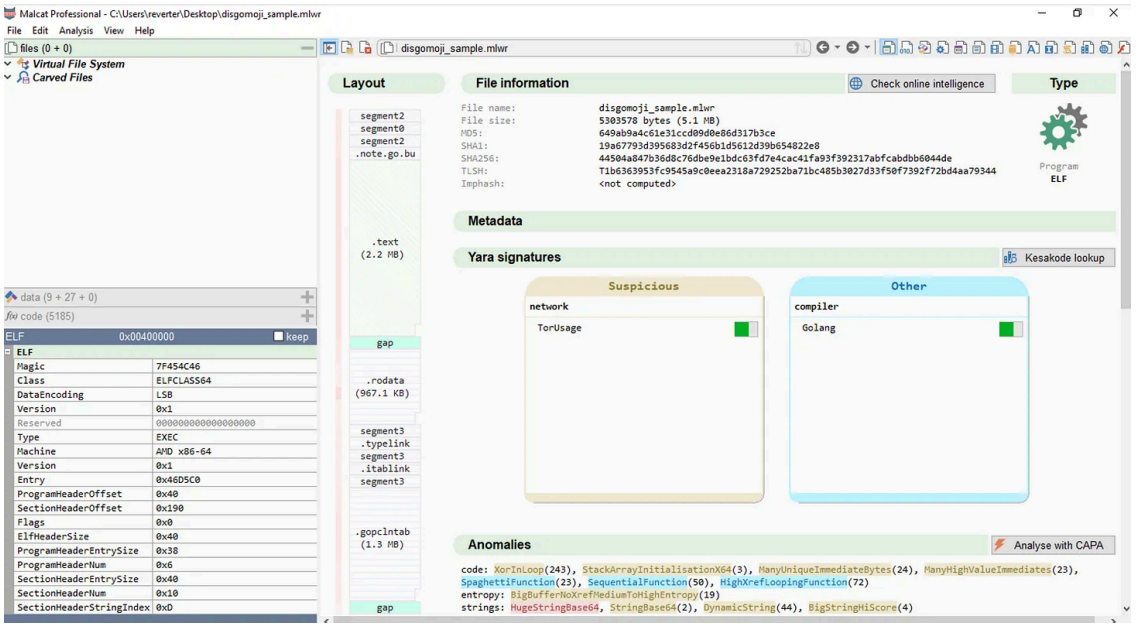
Renaud Tabary is a French person who actively develops Malcat. Since I’m an active user of Malcat, I have been in touch several times. He responded to my questions quickly and fixed a Yara error immediately, where our Yara dataset would match very slowly against a sample.

The sample we will examine, India_Emerging_Global_Economy, with Malcat, has been uploaded from the Netherlands to VirusTotal since May 2024 and can be found here. As of the writing of this anchored narrative in August 2024, the sample is still not detected by any of the leading anti-virus engines.



Screenshot 2: Zero anti-virus detection in August 2024

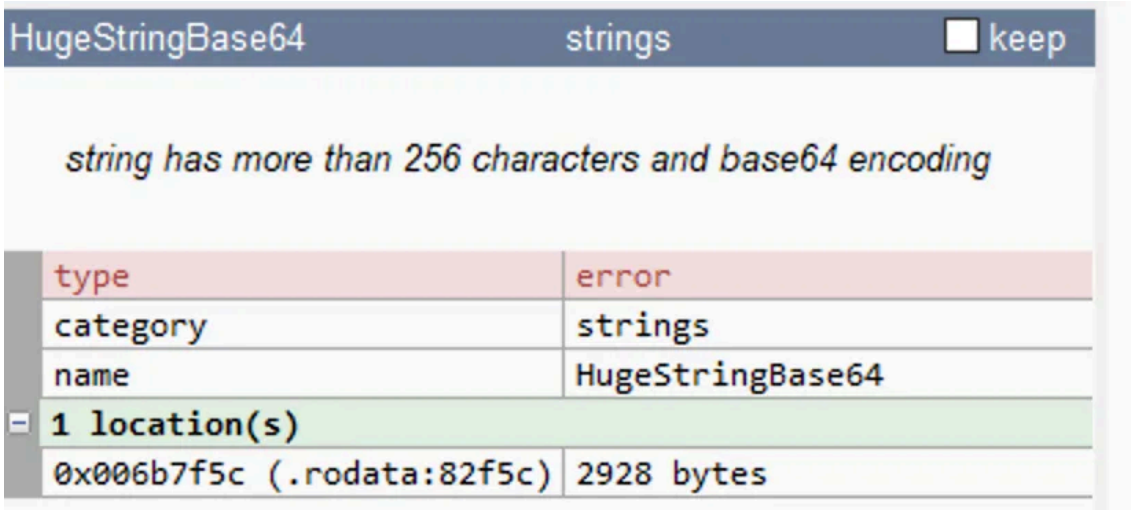
Now, let’s load the sample in Malcat. You can configure Malcat to integrate it into the Explorer context menu.



Screenshot 3: DISGOMOJI loaded into Malcat

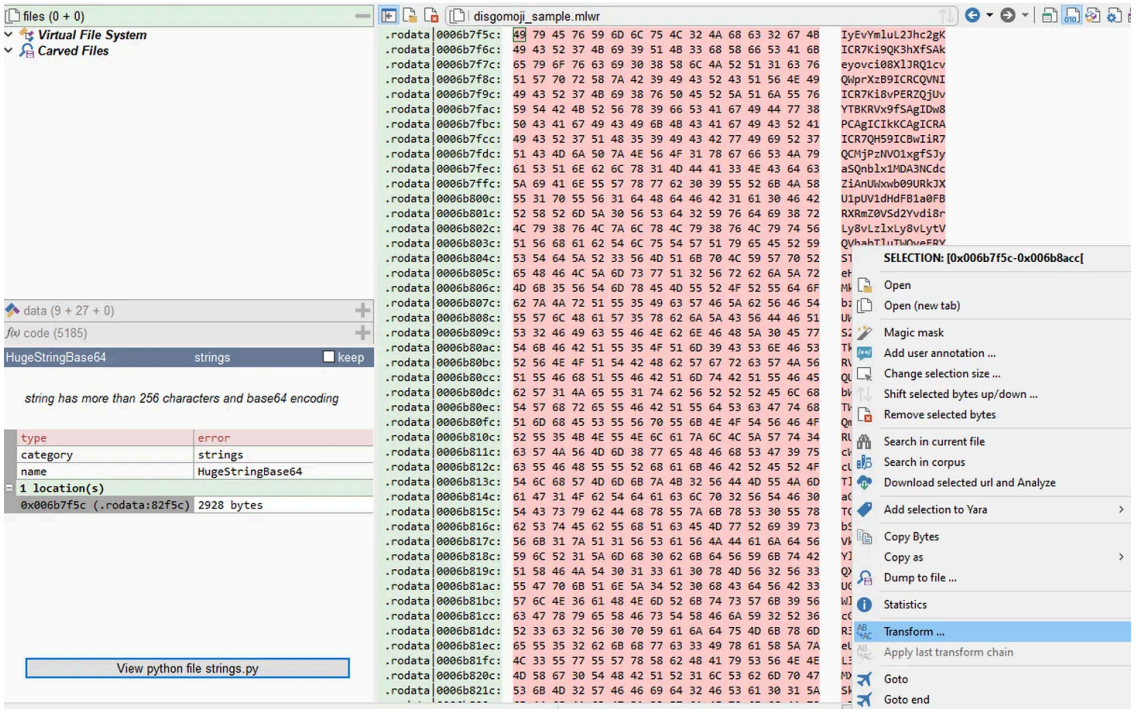
Once you've loaded the sample, you'll see the above screen, where files embedded in the sample will be presented on the left upper side. On the left bottom side, relevant code sections will be displayed. In this example, the sample is not loaded with a custom Yara rule base but only those supported in Malcat. On the top right, you can easily navigate between different tool functionalities. There is a hexadecimal view, assembly, code decompilation, strings, Kesakode, threat intelligence, etc. Malcat also detects strings pushed on the stack during runtime in the strings view and has very decent string support for GO-lang binaries.

During the sample's loading, Malcat also detects relevant anomalies and presents them to the user in the "Anomalies" section of the initial screen. In the screenshot above, "HugeStringBase64" is detected. The anomaly will be displayed on the bottom left screen if you click on it.



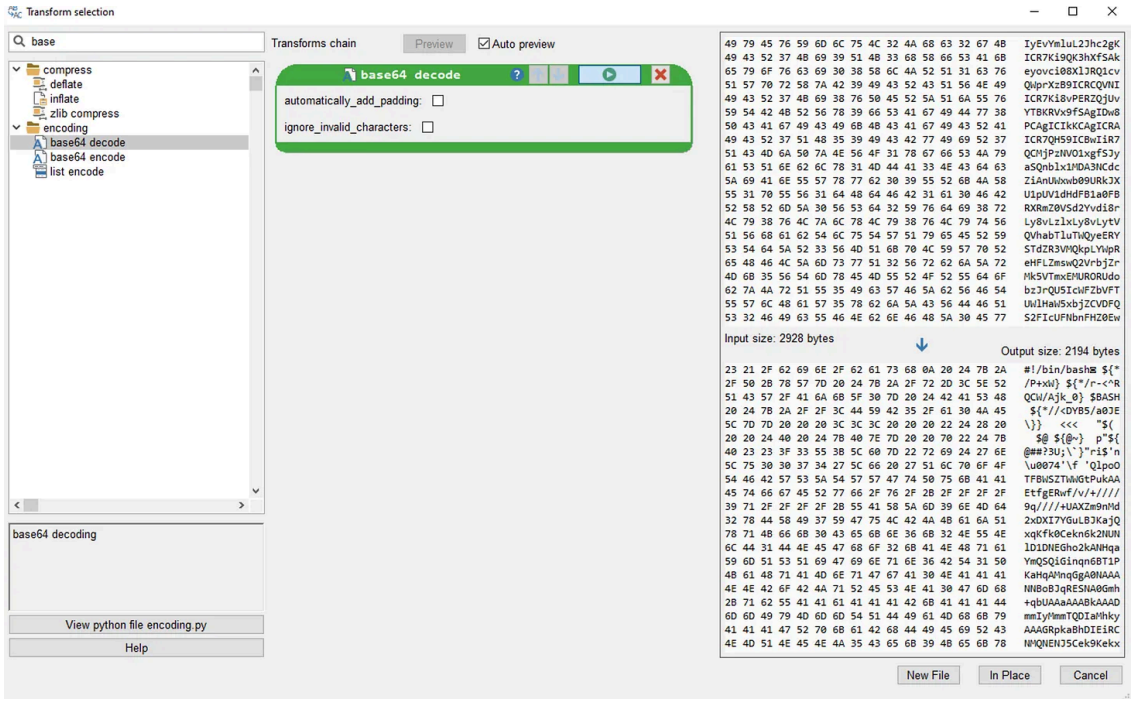
Screenshot 4: A large block of Base64 is automatically detected

By clicking on the bytes, the entire selection will be displayed in the hexadecimal view, as seen below.



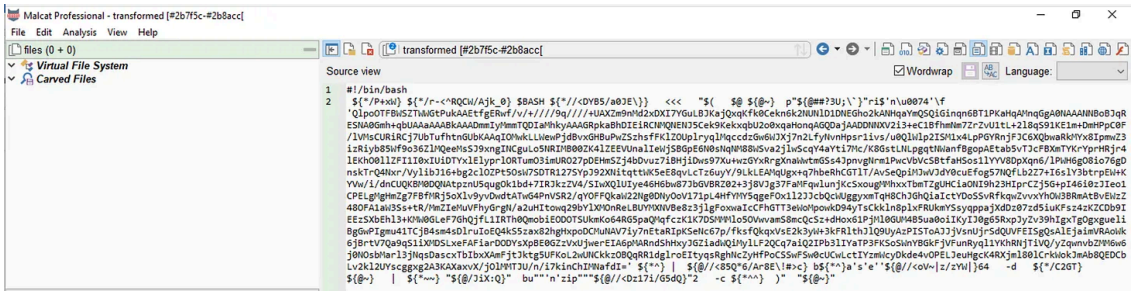
Screenshot 5: Transform the selected base64

Once the bytes have automatically been selected, you can right-click to display Malcat's context menu. In this case, I have chosen Transform because we want to decode the base64-encoded blob.



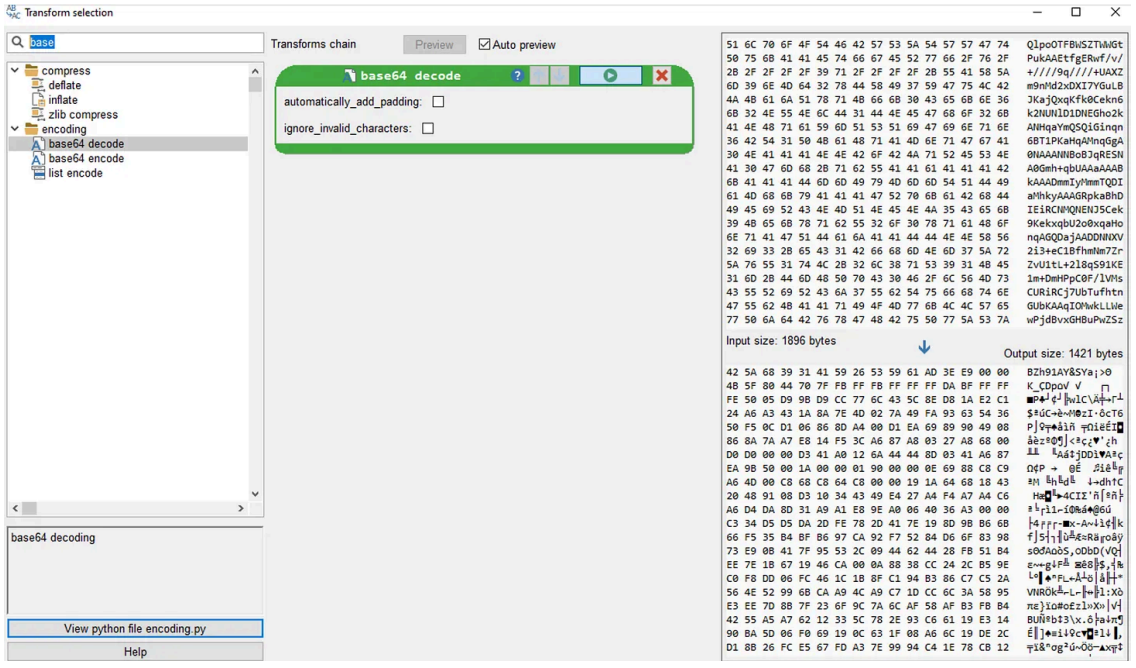
Screenshot 6: Base64 decode

As you can see in the screenshot above, the selected data is automatically decoded, and you can see a piece of a bash script with some base64 encoded data. Then, you can choose to have the file presented “In Place” or as a “New File.” If you enabled your own Yara rule set, the rule set is matched against the new file. In this case I will choose for “New File”.



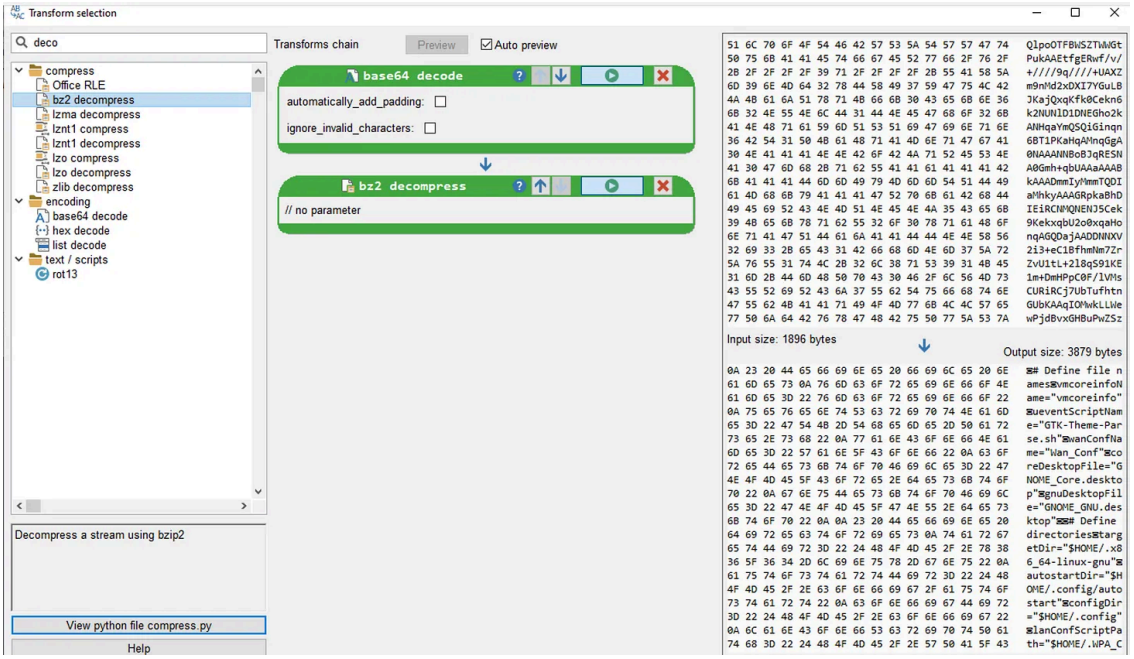
Screenshot 7: Transformed file in Malcat

If you examine the obfuscated bash script, you'll see that it's base64 encoded data which is then piped against "bu""'n'zip". Let's select the base64 data and transform it once more.



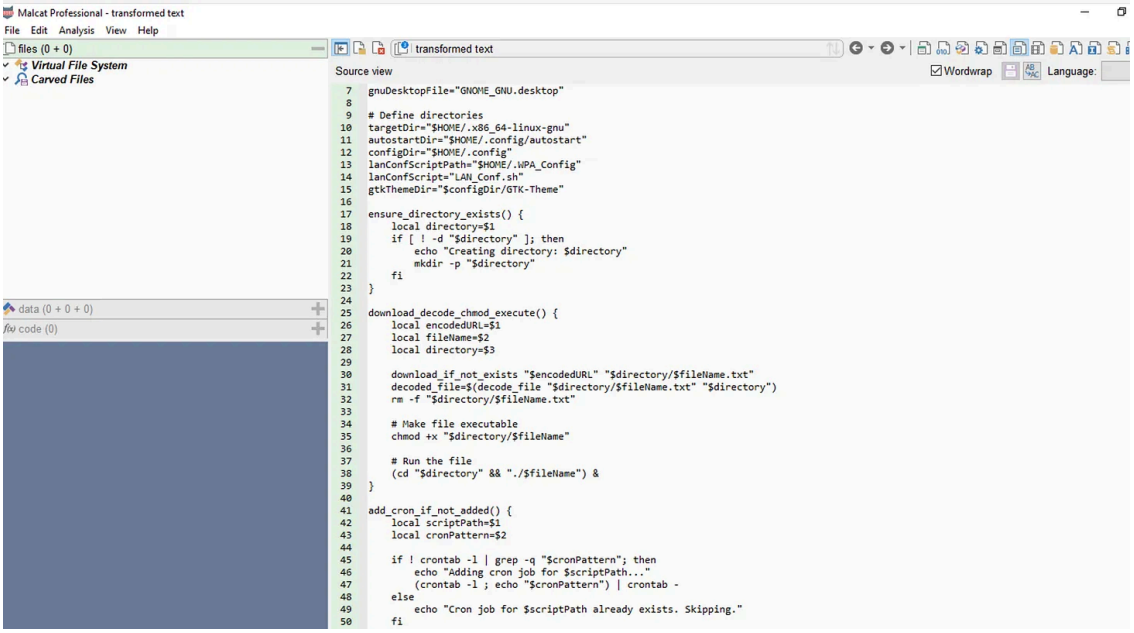
Screenshot 8: BZip2 header revealed

Now, we can see that after decoding the base64, the data is still encoded, but it is now bzip2 compressed data, as the header starts with "BZh9". The next step would be decompressing the data with another Transform in Malcat.



Screenshot 9: The final bash payload is displayed in Malcat

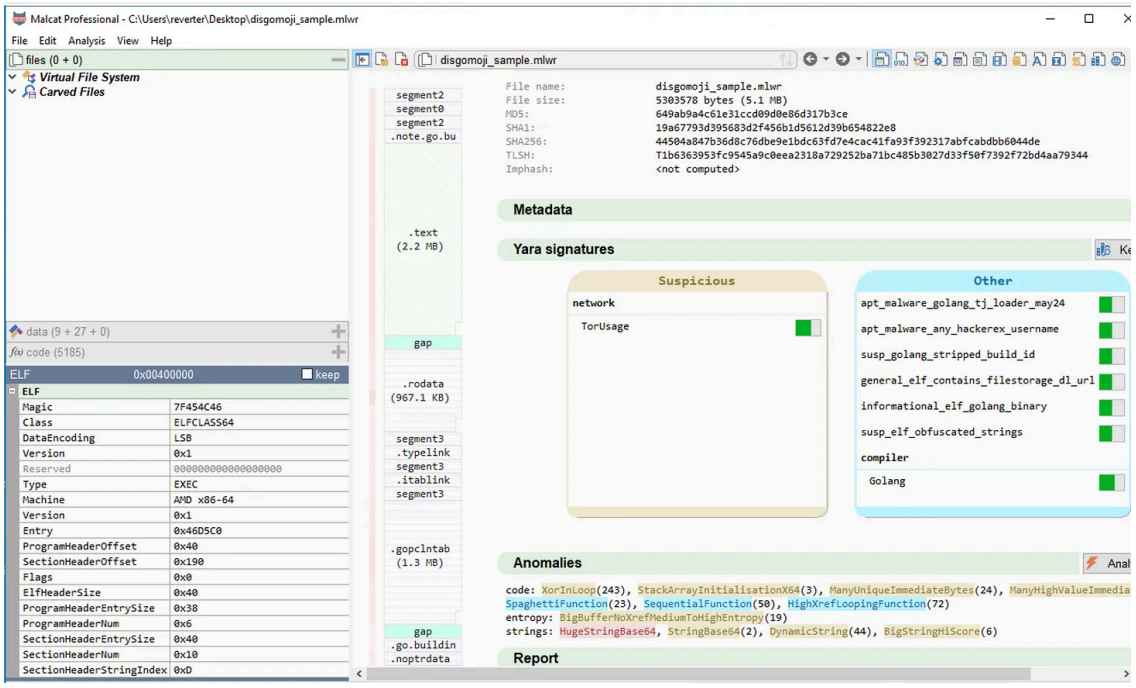
As you can see in the screenshot above, you can stack decoding recipes just like in CyberChef. Malcat also provides Transform functionality for AES and RC4 decoding, for example. The next step is to display the entire configuration of the DISGOMOJI malware.



Screenshot 10: The final payload configuration of DISGOMOJI

With a few simple actions, Malcat transforms data to display the final configuration of the DISGOMOJI malware. Let’s briefly zoom in on some of the features if you customize Malcat with your Yara rule set.

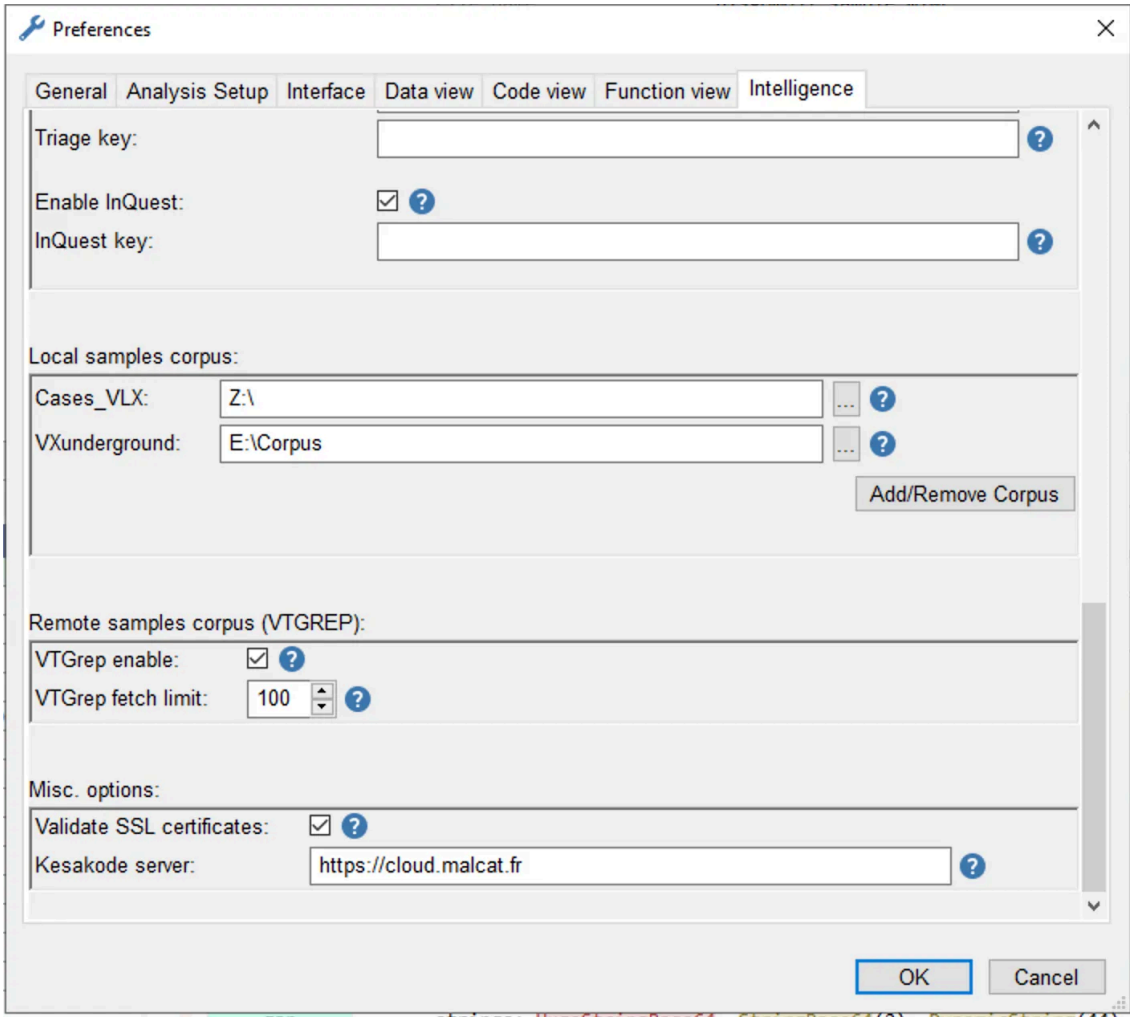
Malcat's documentation clearly describes how you can leverage your own Yara rule set. I have configured the rule set on my host system to update it with new Yara rules.



Screenshot 11: Own Yara dataset enabled

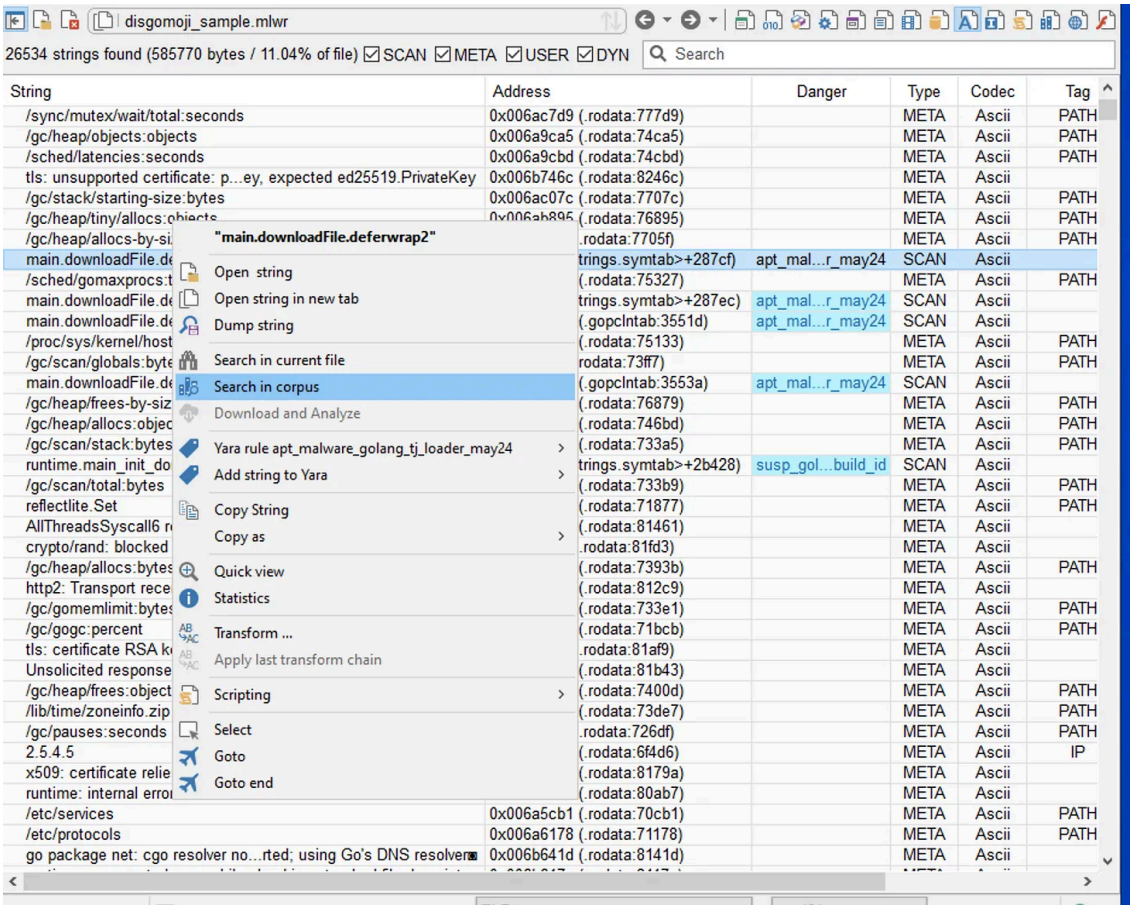
Once I have enabled the custom Yara rule base, all relevant detections will be displayed in the “Other” view under “Yara signatures,” as seen in the screenshot above.

Another feature I would like to share is the ability to look up relevant strings or byte patterns of a new sample against a corpus of malware samples.



Screenshot 12: Configuration of the Malware Corpus

In the screenshot above, you can see that I have enabled the VXUnderground dataset and my own dataset.



Whenever you find interesting strings or byte patterns that you want to check against your configured malware corpora, you can select the strings and select “Search in corpus” via the context menu of Malcat. If a sample is matched, it will be displayed in the search results.

I have highlighted some of Malcat's great functionalities in this anchored narrative based on a known DISGOMOJI sample we encountered in an active breach of the Indian government by Pakistani hackers. If you cannot find your way in IDA or Ghidra, Malcat might be yours. It will also increase the quality of your Yara signature writing as it has some great features I have not yet highlighted. Malcat is a time saver as it drastically improved my static analysis speed.

I will update you on the Bhima Koregaon case in the subsequent anchored narrative. The defendants are still in jail because a hacker group dubbed ModifiedElephant planted manipulated digital evidence on their computers.

A la prochaine!
