The Mystery of Metador | An Unattributed Threat Hiding in Telcos, ISPs, and Universities

sentinelone.com/labs/the-mystery-of-metador-an-unattributed-threat-hiding-in-telcos-isps-and-universities/

Juan Andrés Guerrero-Saade

By Juan Andres Guerrero-Saade, Amitai Ben Shushan Ehrlich, and Aleksandar Milenkoski

Executive Summary

- SentinelLabs researchers uncovered a never-before-seen advanced threat actor we've dubbed 'Metador'.
- Metador primarily targets telecommunications, internet service providers, and universities in several countries in the Middle East and Africa.
- The operators are highly aware of operations security, managing carefully segmented infrastructure per victim, and quickly deploying intricate countermeasures in the presence of security solutions.
- Metador's attack chains are designed to bypass native security solutions while deploying malware platforms directly into memory. SentinelLabs researchers discovered variants of two long-standing Windows malware platforms, and indications of an additional Linux implant.
- At this time, there's no clear, reliable sense of attribution. Traces point to multiple developers and operators that speak both English and Spanish, alongside varied cultural references including British pop punk lyrics and Argentinian political cartoons.
- While Metador appears primarily focused on enabling collection operations aligned with state interests, we'd point to the possibility of a high-end contractor arrangement not tied to a specific country.
- This release is a call to action for threat intelligence researchers, service providers, and defenders to collaborate on tracking an elusive adversary acting with impunity.

Read the Full Report

Introduction

The term 'Magnet of Threats' is used to describe targets so desirable that multiple threat actors regularly cohabitate on the same victim machine in the course of their collection. In the process of responding to a series of tangled intrusions at one of these Magnets of Threats, SentinelLabs researchers encountered an entirely new threat actor. We dubbed this threat actor 'Metador' in reference to the string "I am meta" in one of their malware samples and the expectation of Spanish-language responses from the command-and-control servers.

Our research on Metador was presented at the inaugural LABScon in Arizona. In this post, we offer a short summary of our full findings, which include a detailed report, threat indicators, and an extensive Technical Appendix.

Metador | Hiding in a Magnet of Threats

The Magnet of Threats in question contained a redundant layering of nearly ten (10) known threat actors of Chinese and Iranian origin, including Moshen Dragon and MuddyWater. Among them, we noticed the use of an unusual LOLbin, the Microsoft Console Debugger `cdb.exe`. CDB was the root of an intricate infection chain that would yield two in-memory malware platforms and indications of additional Linux implants.

The intrusions we uncovered were located primarily in telcos, ISPs, and universities in the Middle East and Africa. We believe that we've only seen a small portion of the operations of what's clearly a long-running threat actor of unknown origin.

Throughout our analysis, we retrieved and analyzed examples of two different malware platforms used by Metador–'metaMain' and 'Mafalda'. These Windows-based platforms are intended to operate entirely in-memory and never touch disk in an unencrypted fashion, eluding native security products and standard Windows configurations with relative ease. The internal versioning of Mafalda suggests that this platform has been in use for some time, and its adaptability during our engagement alone highlights active and continuing development.

We also found indications of additional implant(s):

- 'Cryshell'– a custom implant used for bouncing connections in an internal network to external command-and-control servers, with support for custom port knocking sequences.
- Unknown Linux malware used to pilfer materials from other machines in the target environment and route their collection back to Mafalda.

Part of the difficulty in tracking the breadth of Metador's operations involves their strict adherence to infrastructure segmentation. The attackers use a single IP per victim and build.

Attributing Metador remains a garbled mystery. We encountered multiple languages, with diverse idiosyncrasies indicative of multiple developers. There are indications of a separation between developers and operators, and despite a lack of samples, the version history for at least one of the platforms suggests a history of development that extends far beyond the intrusions we've uncovered.

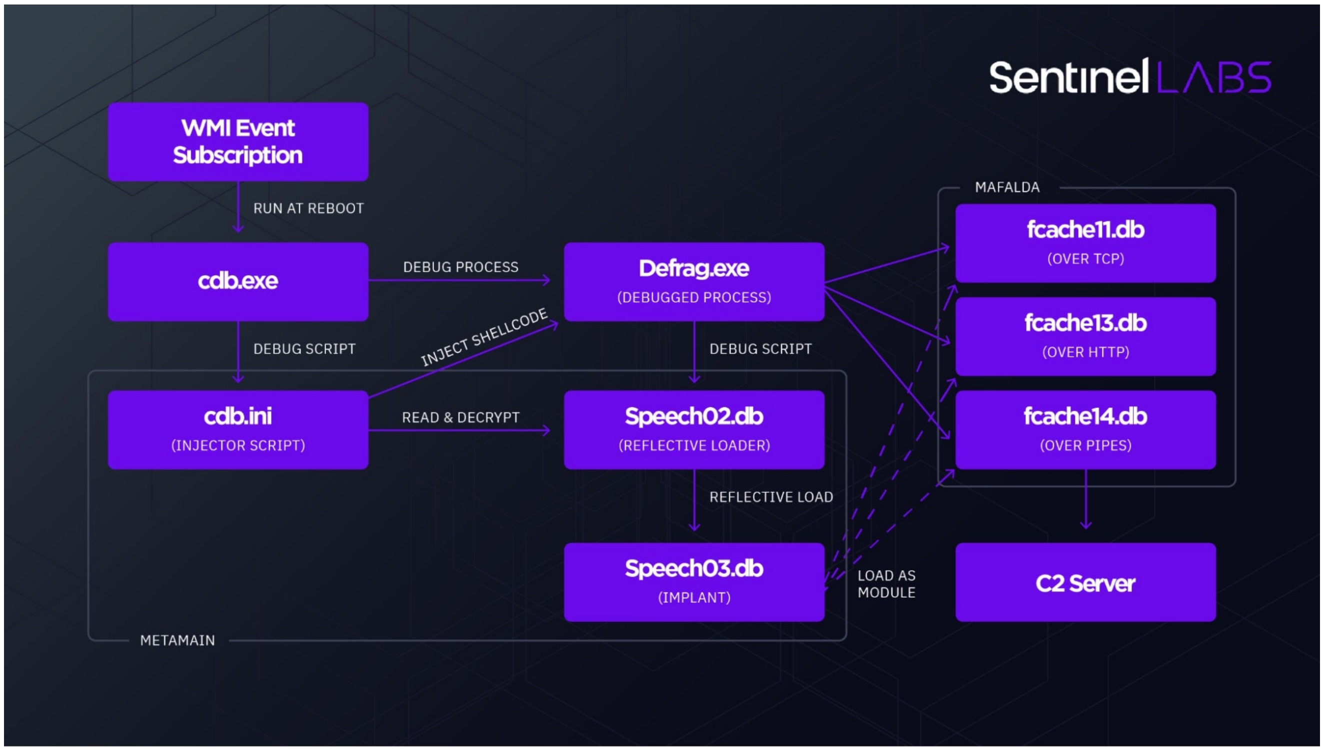
Technical Overview

The Magnet of Threats in question deployed our XDR solution after they'd been infected by Metador for several months. As such, we have no indication of the original infection vector employed in this or other infections.

Once on the target, the Metador operators can choose between multiple execution flows to load one or more of their modular frameworks. The execution flow used on our Magnet of Threats combines a WMI persistence mechanism with an unusual LOLbin in order to kick off the decryption of a multi-mode, in-memory implant we named 'metaMain'.

metaMain is a feature-rich backdoor, but in this case the Metador operators used the metaMain implant to decrypt a subsequent modular framework called 'Mafalda' into memory.

Mafalda is a flexible interactive implant, supporting over 60 commands. It appears to be a highly-valuable asset to the Metador operators, with newer variants exhibiting intense obfuscation making them challenging to analyze.



Metador's multi-framework execution flow

The Many Supported Execution Flows of metaMain

metaMain is an implant framework used to maintain long-term access to compromised machines. It provides operators with extensive functionality, like keyboard and mouse event logging, screenshot theft, file download and upload, and the ability to execute arbitrary shellcode.

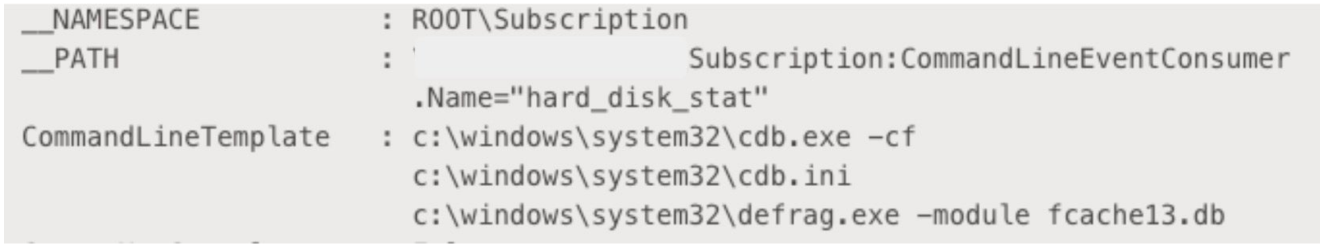
The backdoor is keenly aware of its own execution context and runs in one of two modes as a result. The developers designate these modes by writing out either “I am meta” or “I am main” to a log. We chose to name the platform ‘metaMain’ in reference to these two modes.

metaMain supports multiple start_method’s (i.e., execution flows), with the backdoor’s operations differing slightly per method. The methods supported are CDB_DEBUGGER, HKCMD_SIDELOADING, and KL_INJECTED. We briefly describe CDB_DEBUGGER below, the execution flow seen on our Magnet of Threats. A fuller description of this and additional start_methods, configuration artifacts, and supported commands are available here.

The CDB_DEBUGGER start_method

As the name suggests, this execution scheme relies on CDB, the Microsoft Console Debugger, to carry out the execution process. Within this method, there are two possible variations based on whether the implant is invoked in meta- or main-mode. We witnessed its use in meta-mode, turning the metaMain implant into a glorified loader for a Mafalda implant.

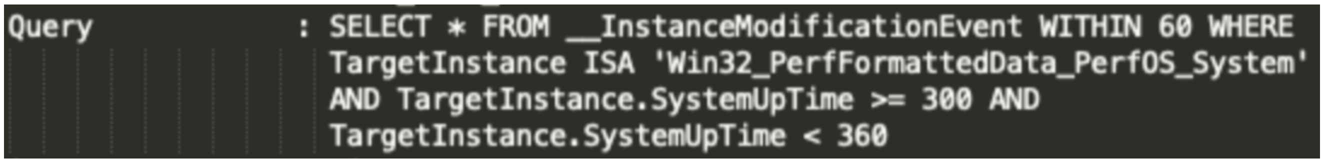
In this case, metaMain’s persistence relies on the abuse of WMI Event Subscriptions. The operators register an event consumer named `hard_disk_stat`.



```
__NAMESPACE            : ROOT\Subscription
__PATH                 : [redacted] Subscription:CommandLineEventConsumer
                         .Name="hard_disk_stat"
CommandLineTemplate    : c:\windows\system32\cdb.exe -cf
                         c:\windows\system32\cdb.ini
                         c:\windows\system32\defrag.exe -module fcache13.db
```

The *hard_disk_stat* event consumer

Five to six minutes after booting up, the event triggers the execution of a LOLbin, `cdb.exe`.

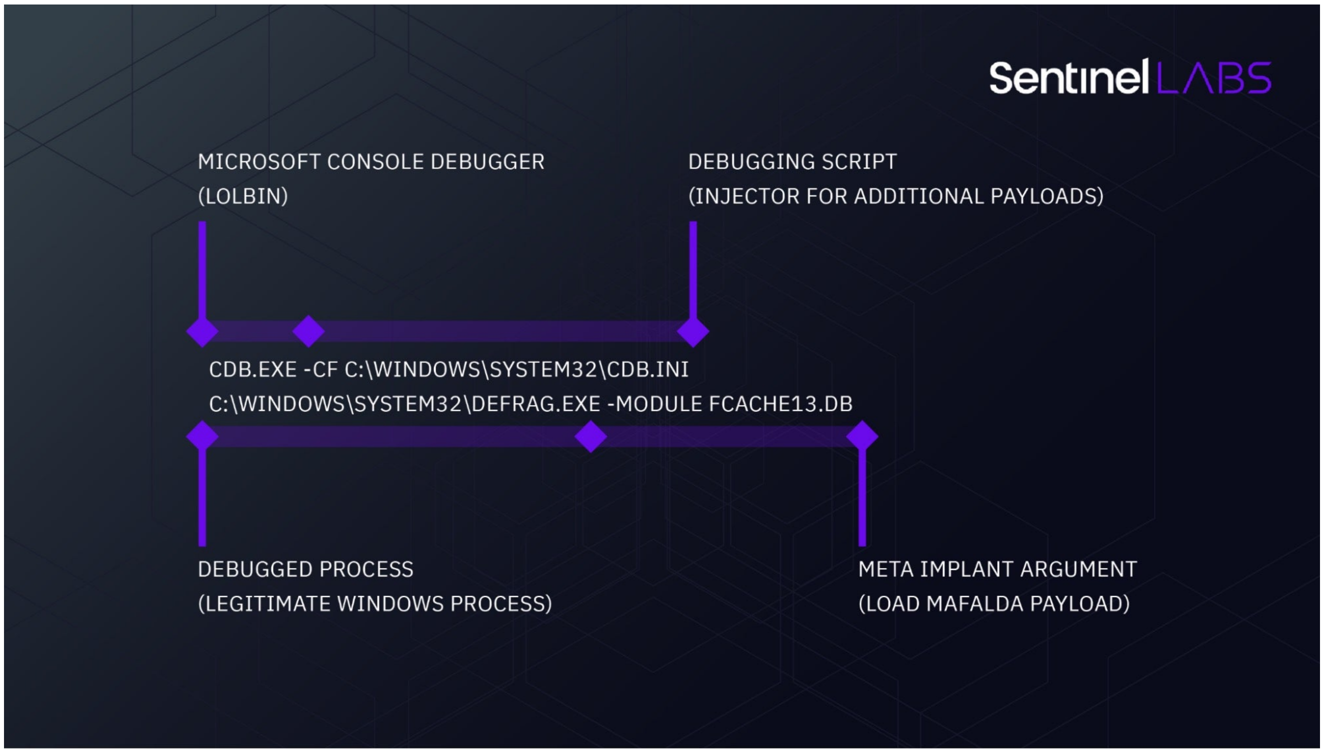


```
Query               : SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE
                      TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System'
                      AND TargetInstance.SystemUpTime >= 300 AND
                      TargetInstance.SystemUpTime < 360
```

WMI Event Subscription

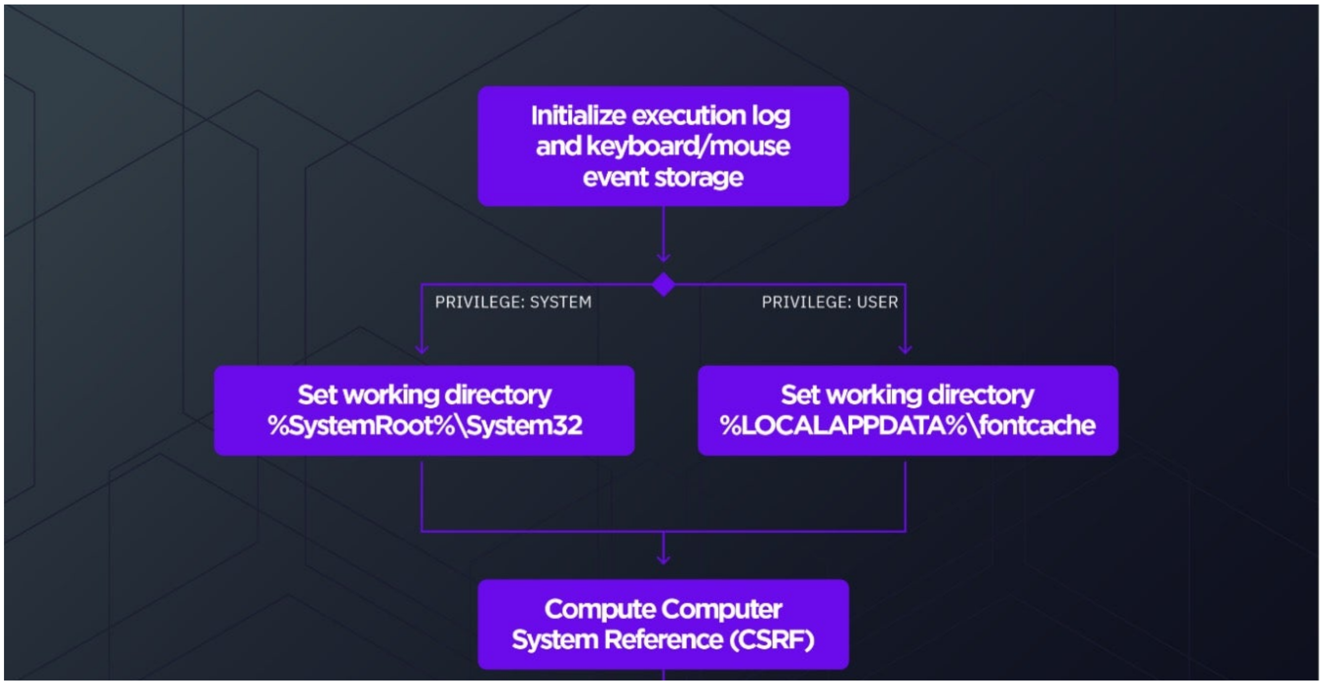
The attackers used the following command line:

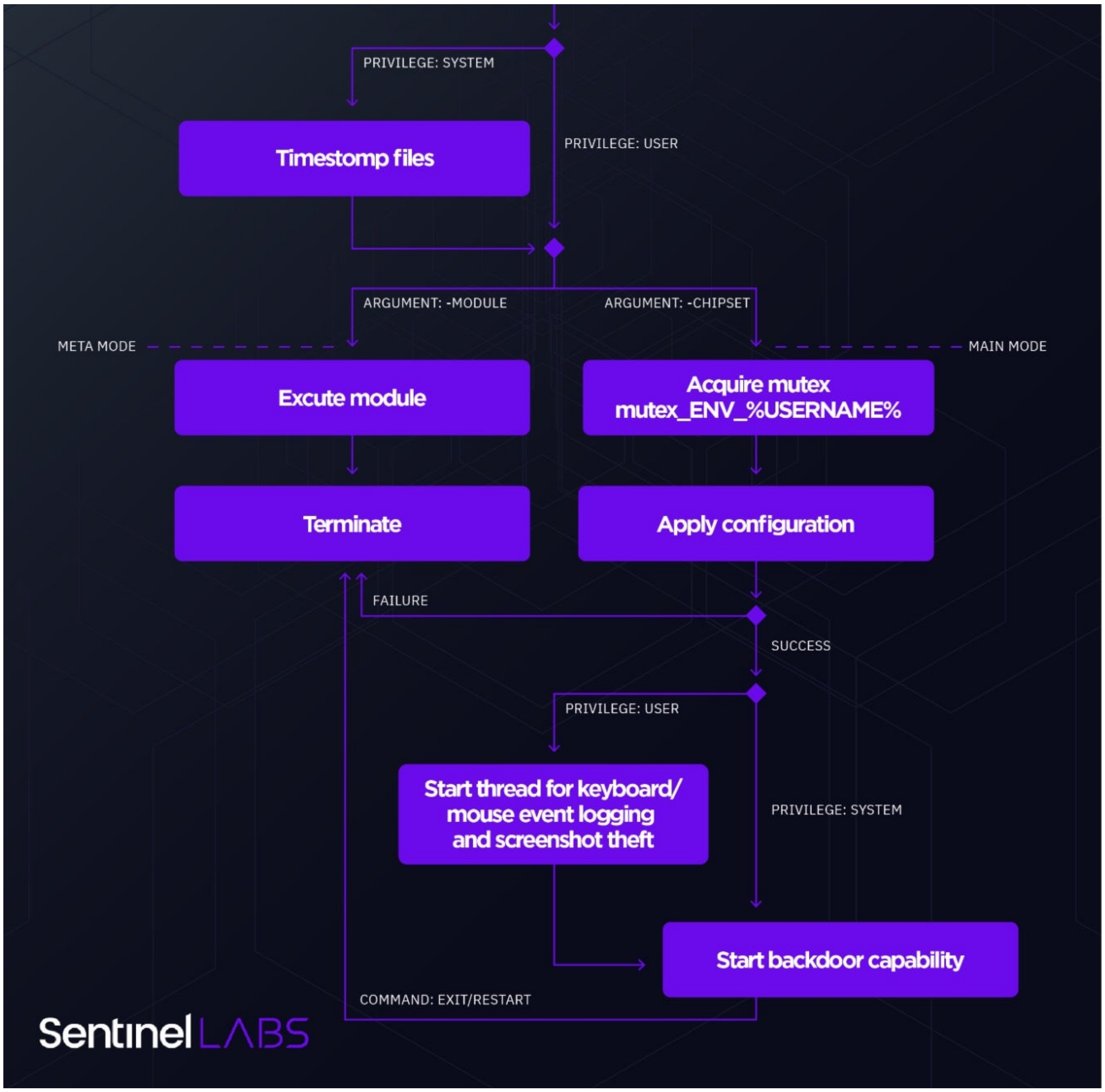
```
cdb.exe -cf c:\windows\system32\cdb.ini c:\windows\system32\defrag.exe -module
fcache13.db
```


SentinelLABS
MICROSOFT CONSOLE DEBUGGER (LOLBIN)
DEBUGGING SCRIPT (INJECTOR FOR ADDITIONAL PAYLOADS)
CDB.EXE -CF C:\WINDOWS\SYSTEM32\CDB.INI C:\WINDOWS\SYSTEM32\DEFRAG.EXE -MODULE FCACHE13.DB
DEBUGGED PROCESS (LEGITIMATE WINDOWS PROCESS)
META IMPLANT ARGUMENT (LOAD MAFALDA PAYLOAD)
The *cdb.exe* command line

A debugging script, `cdb.ini`, is used to inject a small amount of shellcode into the debugged process in order to load metaMain. The shellcode reads, decrypts, and executes metaMain's reflective DLL Loader from `c:\windows\system32\Speech\Speech02.db`. The DLL's sole purpose is to then read, decrypt, and load the metaMain orchestrator, stored as `Speech03.db`.

When invoked in meta-mode, metaMain serves as a loader for the payload provided as an argument following `-module`. In our observed case, the executed module was `fcache13.db`, an encrypted Mafalda payload.


Initialize execution log and keyboard/mouse event storage
PRIVILEGE: SYSTEM
PRIVILEGE: USER
Set working directory %SystemRoot%\System32
Set working directory %LOCALAPPDATA%\fontcache
Compute Computer System Reference (CSRF)



main and meta execution modes

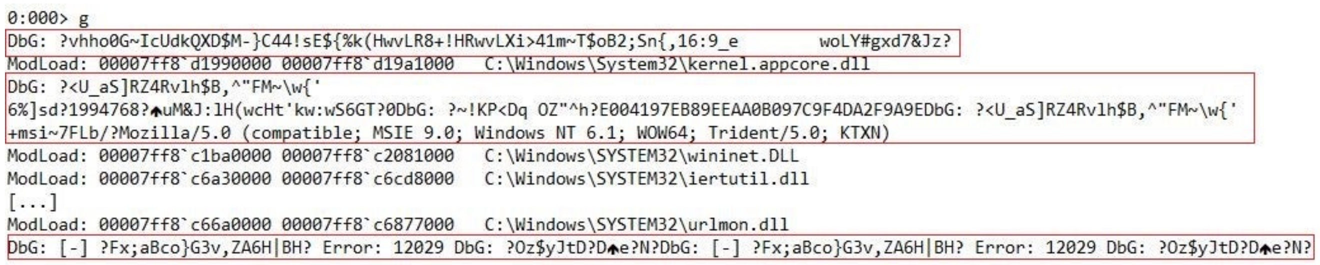
Mafalda Backdoor Overview

The Mafalda implant extends the backdoor functionalities that metaMain provides and is an actively maintained, ongoing project. We observed two variants of the Mafalda backdoor::

- ‘Mafalda Clear Build 144’ – compiled with a timestamp of April 2021
- ‘Obfuscated Mafalda variant’ – compiled with a timestamp of December 2021

The newer Obfuscated Mafalda variant extends the number of supported commands from 54 to 67 and is rife with anti-analysis techniques that make analysis extremely challenging.

Interestingly, we noted that Mafalda prints encrypted debugger messages if the name of the host is WIN-K4C3EKBSMMI, possibly indicating the name of the computer used by the developers.



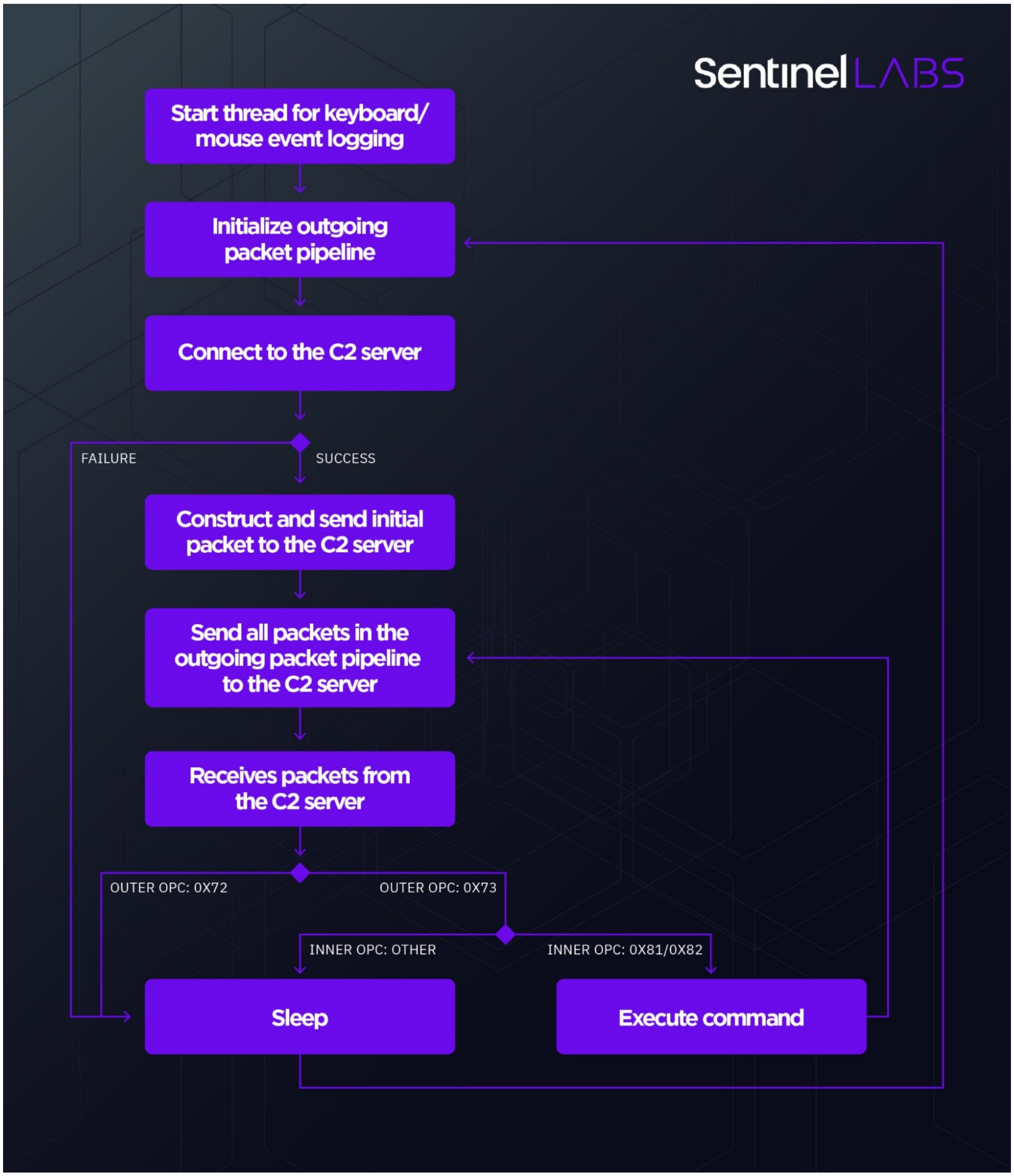
```
0:000> g
DbG: ?vhho0G~IcUdkQXD$M-}C44!sE${%k(HwvLR8+!HRwvLXi>41m~T$oB2;Sn{,16:9_e        woLY#gxd7&Jz?
ModLoad: 00007ff8`d1990000 00007ff8`d19a1000   C:\Windows\System32\kernel.appcore.dll
DbG: ?<U_aS]RZ4Rvlh$B,^"FM~\w{'
6%]sd?1994768?▲uM&J:lH(wcHt'kw:wS6GT?0DbG: ?~!KP<Dq OZ"^h?E004197EB89EEAA0B097C9F4DA2F9A9EDbG: ?<U_aS]RZ4Rvlh$B,^"FM~\w{'
+msi~7FLb/?Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; KTXN)
ModLoad: 00007ff8`c1ba0000 00007ff8`c2081000   C:\Windows\SYSTEM32\wininet.DLL
ModLoad: 00007ff8`c6a30000 00007ff8`c6cd8000   C:\Windows\SYSTEM32\iertutil.dll
[...]
ModLoad: 00007ff8`c66a0000 00007ff8`c6877000   C:\Windows\SYSTEM32\urlmon.dll
DbG: [-] ?Fx;aBco}G3v,ZA6H|BH? Error: 12029 DbG: ?Oz$yJtD?D▲e?N?DbG: [-] ?Fx;aBco}G3v,ZA6H|BH? Error: 12029 DbG: ?Oz$yJtD?D▲e?N?
```

Encrypted debugger messages

If Mafalda successfully establishes a connection to the C2 server, it builds and sends an initial packet containing information about the host environment and the version of Mafalda being run. Mafalda then executes in a loop, exchanging packets with the C2 server.

Each packet is of a given type and subtype, uniquely identified by identification numbers, internally refered to as `outer OPC` and `inner OPC`, respectively:

- Packet of type **0x71** has no impact on the operation of Mafalda.
- Packet of type **0x72** instructs Mafalda to exit the loop and reconnect to the C2 after a sleep period.
- Packet of type **0x73** instructs Mafalda that the packet has a subtype:
  - Subtype **0x81** or **0x82** instructs Mafalda to execute the backdoor command with the command identification number stored in the packet.
  - Any other subtype instructs Mafalda to exit the loop and reconnect to the C2 after a sleep period.


SentinelLABS
Start thread for keyboard/ mouse event logging
Initialize outgoing packet pipeline
Connect to the C2 server
FAILURE
SUCCESS
Construct and send initial packet to the C2 server
Send all packets in the outgoing packet pipeline to the C2 server
Receives packets from the C2 server
OUTER OPC: 0X72
OUTER OPC: 0X73
INNER OPC: OTHER
INNER OPC: 0X81/0X82
Sleep
Execute command
Overview of Mafalda Backdoor operation

Mafalda Backdoor Commands

The Mafalda backdoor has a total of 67 commands, with 13 of these added in the newer variant, indicating that the Mafalda implant is a maintained, ongoing project.

The full unobfuscated list of commands, along with the developer's descriptions, are available from our full report. Some of the more interesting commands only available in the newer Mafalda variant include:

- **Command 55** – Copies a file or directory from an attacker-provided source filesystem location to an attacker-provided destination file system location.
- **Command 60** – Reads the content of

  ```
  %USERPROFILE%\AppData\Local\Google\Chrome\User Data\Local State
  ```

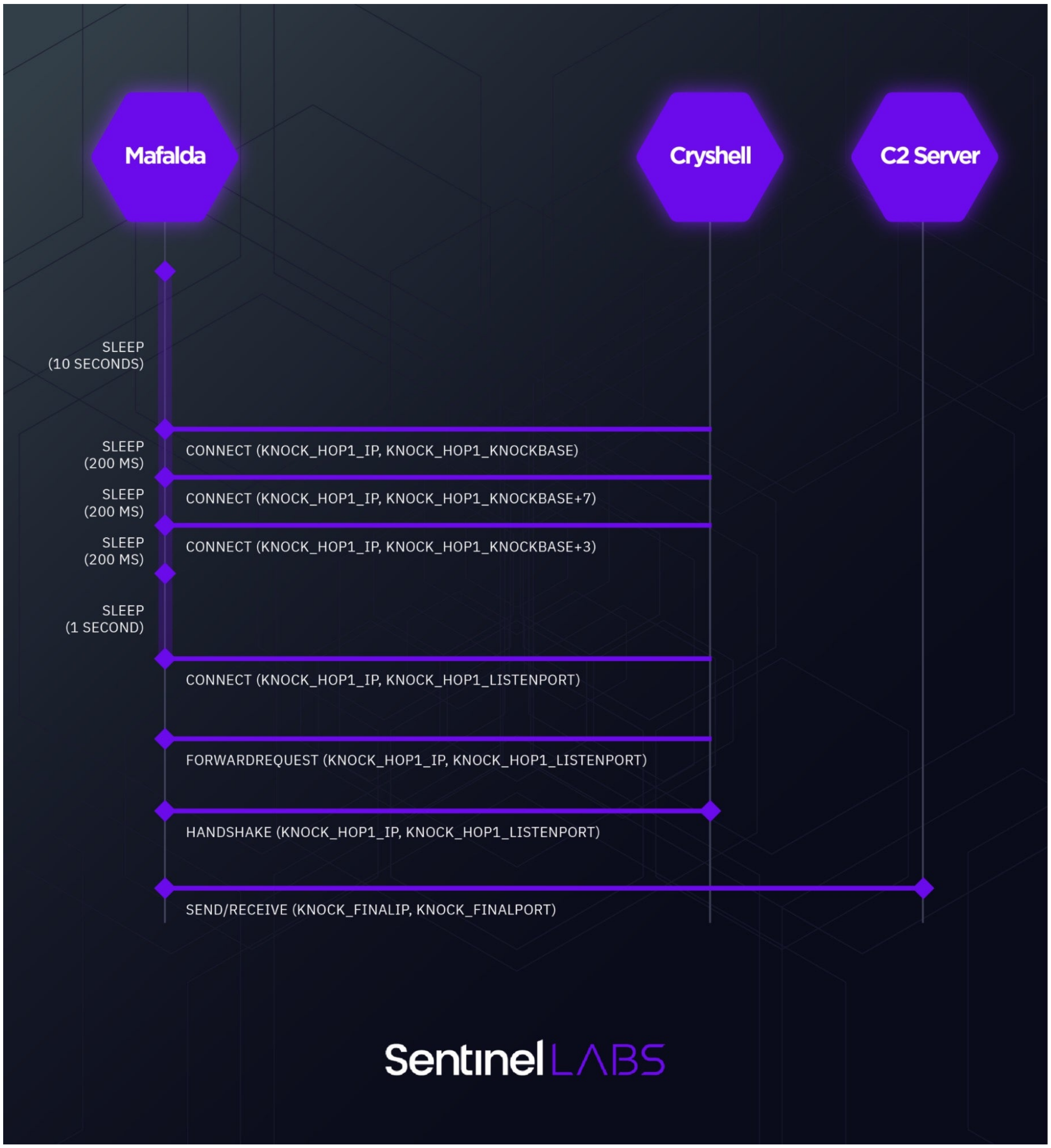
  and sends the content to the C2 with a name prefixed with `loot\`.

- **Command 63** – Conducts network and system configuration reconnaissance
- **Command 67** – Retrieves data from another implant that resides in the victim's network and sends the data to the C2

The functionalities of the backdoor commands have a very broad scope and include credential theft, data and information theft, command execution, system registry and file system manipulation, and Mafalda reconfiguration.

Cryshell and Additional Implants

When the TCP KNOCK communication method is enabled, the metaMain and Mafalda implants can establish an indirect connection to the C2 server through another implant. On Windows systems, this implant is internally referred to as 'Cryshell'. metaMain and Mafalda authenticate themselves to Cryshell through a port-knocking and handshake procedure.


Mafalda
Cryshell
C2 Server
SLEEP (10 SECONDS)
SLEEP (200 MS)
CONNECT (KNOCK_HOP1_IP, KNOCK_HOP1_KNOCKBASE)
SLEEP (200 MS)
CONNECT (KNOCK_HOP1_IP, KNOCK_HOP1_KNOCKBASE+7)
SLEEP (200 MS)
CONNECT (KNOCK_HOP1_IP, KNOCK_HOP1_KNOCKBASE+3)
SLEEP (1 SECOND)
CONNECT (KNOCK_HOP1_IP, KNOCK_HOP1_LISTENPORT)
FORWARDREQUEST (KNOCK_HOP1_IP, KNOCK_HOP1_LISTENPORT)
HANDSHAKE (KNOCK_HOP1_IP, KNOCK_HOP1_LISTENPORT)
SEND/RECEIVE (KNOCK_FINALIP, KNOCK_FINALPORT)
SentinelLABS
Mafalda authenticates itself to Cryshell

Mafalda also supports retrieval of data from Linux machines with another implant that sends data to the C2 as part of a packet with a name prefixed with `loot_linux\`. Though it's possible that this unnamed Linux implant and Cryshell are the same, Mafalda authenticates itself to the Linux implant through a different port-knocking and handshake procedure.

Infrastructure

In all Metador intrusions we've observed, the operators use a single external IP address per victim network. That IP is utilized for command-and-control over either HTTP (metaMain, Mafalda) or raw TCP (Mafalda). In all confirmed instances, the servers were hosted on LITESERVER, a Dutch hosting provider.

In addition to HTTP, external Mafalda C2 servers also support raw TCP connections over port 29029. We also observed some of Metador's infrastructure host an SSH server at an unusual port. While SSH is commonly used for remote access to *nix systems, we find it hard to believe that a mature threat actor would expose their infrastructure in such a way. Instead, it's likely those were used to tunnel traffic through Mafalda's internal portfwd commands.

We were able to identify one additional server we believe is operated by Metador actors, also hosted on Liteserver – `5.2.78[.]14`. This IP hosts what appears to be a malicious domain, `networkselfhelp[.]com`, which might have been used as a C2 for Metador intrusions. If so, it's an indication that Metador operators not only utilize IPs for their intrusions, but also domains.

Attribution

The limited number of intrusions and long-term access to targets suggests that the threat actor's primary motive is espionage. Moreover, the technical complexity of the malware and its active development suggest a well-resourced group able to acquire, maintain and extend multiple frameworks.

Metador was observed mainly in Telecoms, Internet Service Providers (ISP), and Universities in the Middle East and Africa, and appears intended to provide long-term access in multiple redundant ways.

Mafalda internal documentation suggests the implant is maintained and developed by a dedicated team, leaving comments for a separate group of operators.

Conclusion

Running into Metador is a daunting reminder that a different class of threat actors continues to operate in the shadows with impunity. Previous threat intelligence discoveries have broadened our understanding of the kind of threats that are out there but so far, our collective ability to track these actors remains inconsistent at best. Developers of security products in particular should take this as an opportunity to proactively engineer their solutions towards monitoring for the most cunning, well-resourced threat actors. High-end threat actors are thriving in a market that primarily rewards compliance and perfunctory detections.

From the perspective of the threat intelligence research community, we are deeply grateful for the contributions of the research teams and service providers who have willingly shared their expertise and telemetry for this research.

We hope that this publication will incentivize further collaboration and provide us with answers to the mystery of Metador, and to that end we urge interested researchers to read the full version of this report, where a list Indicators of Compromise can also be found, and its extended Technical Appendix.

Read the Full Report