Mapping the Infrastructure and Malware Ecosystem of MuddyWater

Introduction

Since early 2025, Group-IB analysts have observed that MuddyWater, known as an Iranian state-sponsored Advanced Persistent Threat (APT) group, remains active across the Middle East and Europe, with a notable surge in activity within the European region.

Our latest analysis of the group's activities has revealed new intelligence regarding recent shifts in their operational characteristics and arsenal.

The group has significantly reduced its widespread Remote Monitoring and Management based intrusions (RMM), reverting to a more targeted operational approach. Although RMM software continues to be employed, the group has increasingly relied on custom-developed backdoors such as Phoenix and StealthCache in addition to PowerShell-based backdoors.

Recent activity shows that they still rely on phishing for delivery, leveraging maldocs with malicious macros for infection. Infrastructure analysis has revealed active use of Amazon Web Services (AWS) for hosting malicious assets, and Cloudflare services have been leveraged to hide infrastructure fingerprints and impede analysis. Group-IB analysts also identified MuddyWater infrastructure hosted across multiple commercial providers including M247, SEDO, DigitalOcean, OVH and bulletproof providers (like Stark Industries), suggesting a deliberate mix of mainstream and resilient infrastructure.

These findings collectively indicate that MuddyWater remains highly active and is demonstrating increased operational sophistication.

The blog provides an in-depth look at MuddyWater's evolution in tooling, targeting, and infrastructure management, suggesting a more mature and capable advanced persistent threat within the META region.

Key discoveries

- Muddywater significantly reduced opportunistic RMM campaigns in favor of targeted spearphishing and custom malware.
- Multiple new malware variants and tools have been observed to be weaponised by MuddyWater: StealthCache, Phoenix, Fooder, LiteInject, and others.
- MuddyWater continues to be active in the Middle East with increased activity in Europe and the United States.
- The group continues to rely on phishing and maldocs for initial access.
- MuddyWater are weaponising open-source golang projects in their operations.
- Network Infrastructure variation; AWS, Cloudflare, M247, OVH, and bulletproof hosting like Stark Industries.
- Exposing MuddyWater activity by tracking their footprints across infrastructure and open source intelligence.

Who may find this blog interesting

- Cyber Threat Intelligence and Threat Hunting Specialists.
- Cybersecurity Analysts and Corporate Security Teams.
- National Cybersecurity Centers and Intelligence Agencies.
- Computer Emergency Response Teams (CERT).
- Malware Analysts.

Group-IB Threat Intelligence Portal: MuddyWater

Group-IB customers can access our Threat Intelligence portal for more information about MuddyWater and Malware profiles.

Malware Breakdown Used by the Threat Actor:

Malware Profile

GROUP-IB

BugSleep

Type
Backdoor

Platform
Windows

First seen
2024-05-07

Last Discovered
[If available/applicable.
If none, indicate N/A.]

ABOUT

BugSleep is a custom backdoor developed by the Iranian threat group MuddyWater, affiliated with the Ministry of Intelligence and Security (MOIS). First reported by ClearSky Cyber Security on June 9, 2024, the malware has been deployed in phishing campaigns targeting organizations in the Middle East. BugSleep is designed to execute commands and facilitate file transfers between compromised machines and its command-and-control (C&C) server. To evade detection, it employs repeated Sleep API calls and encrypted configurations to hide its C&C details. Continuous updates and bug fixes suggest ongoing development by the threat actors.

FEATURES

- Written in Python, often converted to standalone executables with PyInstaller.
- Communicates with C2 servers over HTTP(S) using obfuscated requests.
- Supports basic backdoor functions: file download, file upload, command execution.
- Uses Base64 encoding for data exfiltration and communication.
- Employs simple string obfuscation to evade static detection.
- Persistence achieved via registry Run keys or scheduled tasks.
- Typically delivered through spear-phishing documents with malicious macros.
- Part of MuddyWater's evolving Python-based malware toolkit.
- Used to establish initial foothold and remote control before deploying additional tools.

FORMATS

EXE

THREAT ACTORS

MuddyWater

TOP TACTICS

T1559.001 - Inter-Process Communication -> Component Object Model
T1106 - Native API
T1053.005 - Scheduled Task/Job -> Scheduled Task
T1059.003 - Command and Scripting Interpreter -> Windows Command Shell
T1055.002 - Process Injection -> Portable Executable Injection
T1140 - Deobfuscate/Decode Files or Information
T1027.007 - Obfuscated Files or Information -> Dynamic API Resolution
T1620 - Reflective Code Loading
T1497.003 - Virtualization/Sandbox Evasion -> Time Based Evasion
T1057 - Process Discovery
T1082 - System Information Discovery
T1105 - Ingress Tool Transfer
T1132.002 - Data Encoding -> Non-Standard Encoding
T1573.001 - Encrypted Channel -> Symmetric Cryptography
T1071.001 - Application Layer Protocol -> Web Protocols
T1041 - Exfiltration Over C2 Channel

Malware Profile

GROUP-IB

LiteInject

Type
Injector

Platform
Windows

First seen
2025-02-19

Last Discovered
N/A

ABOUT

This malware is a PE injector that accepts two command-line arguments: the target process ID and the path to a PE file to inject. It performs manual mapping by writing its own shellcode and the provided module into the target process's memory. This malware is attributed to MuddyWater with moderate confidence.

FEATURES

- Reads the specified PE file from disk (the payload).
- Opens a handle to the target process using the provided PID.
- Injects its own module into the target process's memory.
- Writes the provided PE module into the same process.
- Executes a shellcode function from its own module in the target process using ZwCreateThreadEx.

FORMATS

EXE

THREAT ACTORS

MuddyWater

TOP TACTICS

T1055.003 - Thread Execution Hijacking
T1036.005 - Match Legitimate Resource Name or Location
T1027.007 - Dynamic API Resolution
T1055.003 - Thread Execution Hijacking

Malware Profile

GROUP-IB

Stealth Cache

Type
Backdoor

Platform
Windows

First seen
2025-03-09

Last Discovered
N/A

ABOUT

StealthCache is a more advanced backdoor compared to BugSleep. While it has some minor similarities, it is mostly different malware. It has a rich feature set, and extended C2 communication.

FEATURES

- Read files and exfiltrate
- Write files received from C2
- Kill self
- Restart self
- Delete own executable using Alternate Data Stream (ADS)
- Copy self to hard-coded location
- Check for specific processes (EDR, anti-virus, and malware analysis tools)
- Create and show a Windows credential prompt to steal Windows credentials
- Execute stealer and exfiltrate files
- Exfiltrate local domain host name
- Only works on a device with a specific device name and username

FORMATS

EXE

THREAT ACTORS

MuddyWater

TOP TACTICS

T1059.001 - Command and Scripting Interpreter -> PowerShell
T1574.002 - Hijack Execution Flow -> DLL Side-Loading
T1140 - Deobfuscate/Decode Files or Information
T1070.004 - Indicator Removal -> File Deletion
T1027.002 - Obfuscated Files or Information -> Software Packing
T1497.003 - Virtualization/Sandbox Evasion -> Time Based Evasion
T1622 - Debugger Evasion
T1057 - Process Discovery
T1560 - Archive Collected Data
T1005 - Data from Local System
T1132.002 - Data Encoding -> Non-Standard Encoding
T1071.001 - Application Layer Protocol -> Web Protocols
T1041 - Exfiltration Over C2 Channel

Malware Profile

GROUP-IB

Fooder

Type
Loader

Platform
Windows

First seen
2025-03-09

Last Discovered
N/A

ABOUT

Fooder is a loader known to be used to load, decrypt, and run an encrypted and packed malware payload in memory. Some of the observed samples come as a DLL intended to be side-loaded by a legitimate application. The decryption of the packed payload occurs via a hard-coded decryption key and standard Windows encryption APIs. One key characteristic of the dropper is that it contains log messages that are printed to the console at runtime like the following:

```
--------------------------------------------------
*********          FINISH          *********
--------------------------------------------------
--------------------------------------------------
*********          GOODBYE         *********
--------------------------------------------------
```

FEATURES

- Contains packed encrypted payload
- Runs packed payload in memory
- Prints log messages to the console

FORMATS

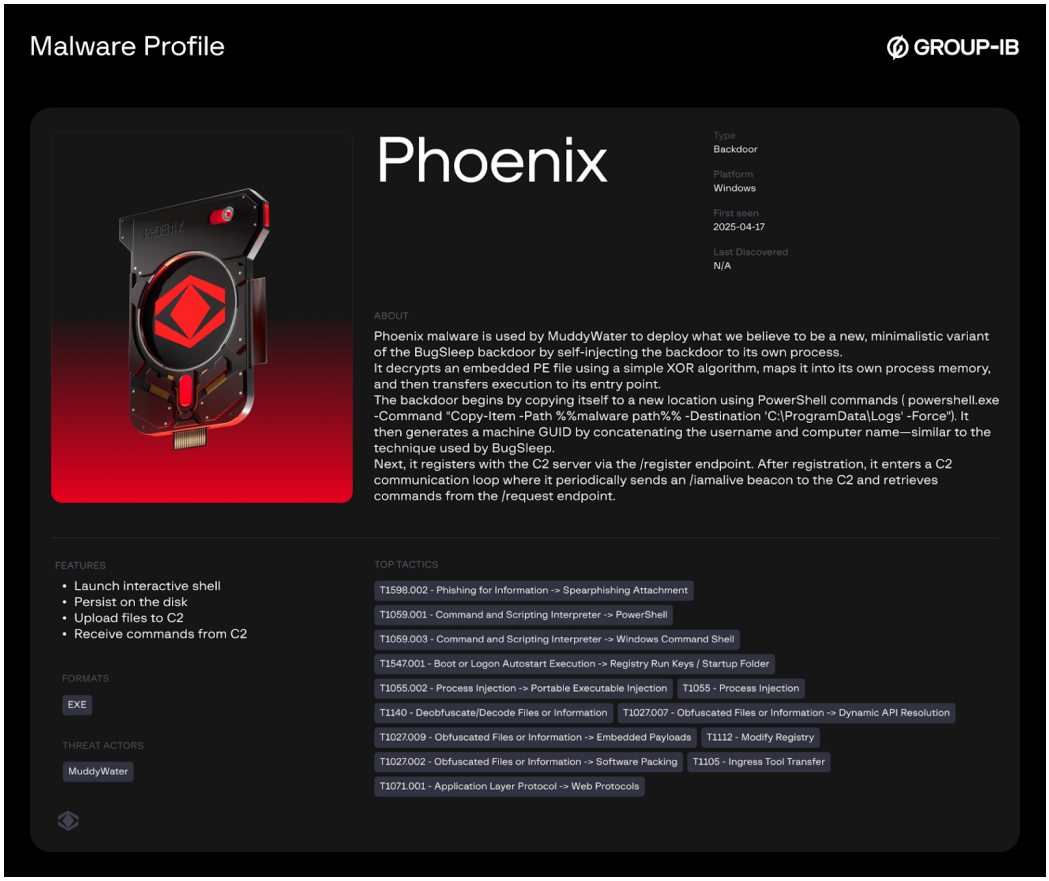
DLL
EXE

THREAT ACTORS

MuddyWater

TOP TACTICS

T1574 - Hijack Execution Flow
T1140 - Deobfuscate/Decode Files or Information
T1036.005 - Match Legitimate Resource Name or Location
T1027.002 - Software Packing



Strategic Context

MuddyWater represents a sophisticated Advanced Persistent Threat (APT) group. It is believed to be operating under Iran's Ministry of Intelligence and Security (MOIS), functioning as a critical component of Tehran's offensive cyber capabilities. Active since at least 2017, this state-sponsored actor executes strategic espionage and disruptive campaigns aligned with Iranian geopolitical objectives.

Their operations often involve long-term persistence, prioritizing stealth and extended access. They regularly update their tools and techniques to stay ahead of detection and attribution, using both custom and off-the-shelf tools. Their counter-attribution methodology includes deliberate insertion of false flags and misleading artifacts within malware samples and infrastructure, creating analytical complexity for threat intelligence teams.

Operational Scope and Targeting

MuddyWater is widely tracked as an Iran-nexus threat group. Open-source and government reporting indicate activity that extends beyond traditional cyber espionage and may support broader asymmetric cyber objectives. Its campaigns facilitate intelligence collection, and may enable potential disruption capabilities against adversarial infrastructure, with activity reported across strategically significant regions.

Consistent with publicly reported MOIS priorities, the group appears to advance regional strategic objectives through cyberspace. The threat actor maintains a primary operational focus on Middle Eastern targets while demonstrating expanded geographical reach encompassing the United States, and multiple European and Asian nations.

MuddyWater has been observed systematically targeting high-value sectors, critical to national security and economic stability, including:

- Telecommunications infrastructure.
- Government entities.
- Energy sector organizations.
- Defense industrial base.
- Critical infrastructure operators.

MuddyWater's persistent campaigns underscore its role in supporting Iranian intelligence requirements while maintaining plausible deniability for state-directed cyber operations against both regional competitors and Western targets.

Historical vs Current Operational Activity

Historically, MuddyWater has relied heavily on phishing emails as a delivery method to infect victims. The group frequently used compromised or spoofed email accounts to impersonate government or academic entities and trick recipients into opening malicious attachments links. Campaigns generally involved mass distribution with lures designed to appeal to a diverse set of organizations and individuals, with a noticeable focus on particular industries or sectors that align with the group's interests.

Over time, however, the nature of the lures has become significantly less sophisticated and more generic. Where earlier operations featured highly tailored content, more recent campaigns adopted generic themes such as online courses or webinars, often in the English language. This allowed the group to scale operations for higher volume rather than focusing on individual targets.

However, since the beginning of 2025 there was a noticeable shift with this tactic, as they shifted to a more targeted approach. At the same time, we also noticed that RMM campaigns have significantly decreased.

A defining element of MuddyWater's tradecraft has been its use of Remote Monitoring and Management (RMM) tools. 360 Threat Intelligence Center report indicates that MuddyWater has utilized RMMs since 2020. For the past couple of years, compromised emails were also used to register accounts for the RMM tools. The link to download the RMM from a filesharing service is delivered primarily via a phishing email, it appears either in the email itself or within an attachment in the email as illustrated in Figure 1.
Hundreds of such campaigns were detected by Group-IB, with activity reaching its highest level during 2024.

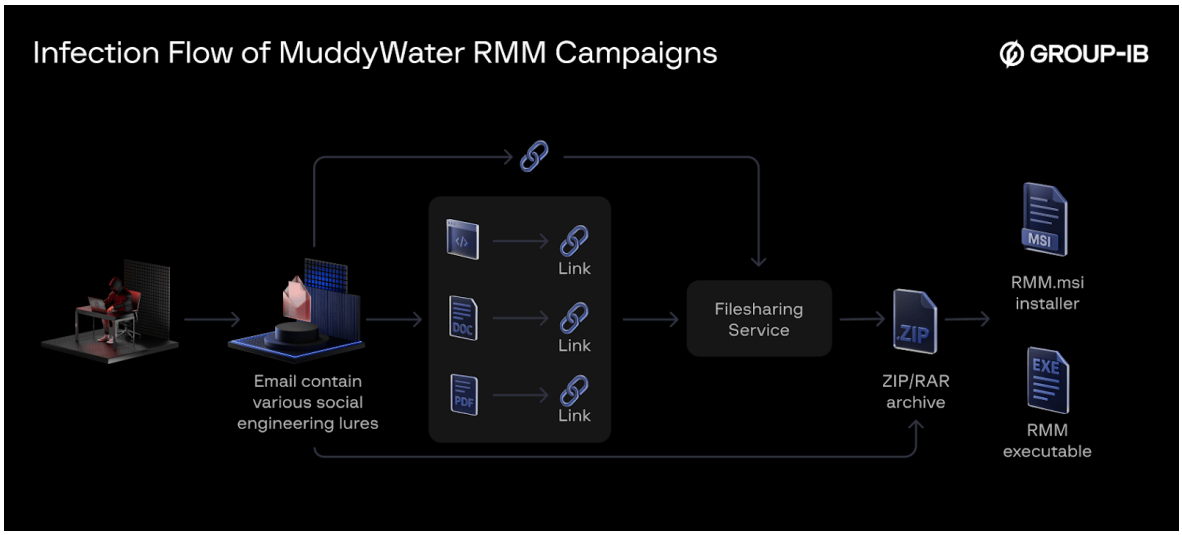


Figure 1. General Infection Chain for MuddyWater RMM campaigns

Below table shows a non-comprehensive list of RMMs the group utilized, and a list of filesharing services used for delivery:

RMM tools	FileSharing Services
• SimpleHelp • Remote Utilities Software • eHorus • N-Able • ScreenConnect • Atera Agent • Syncro • PDQ • Level	• OneHub • FileTransfer.io • Dropbox • Egnyte • Storyblok • OneDrive • Internxt • Sync.com • Mega.nz • Mediafire • Freeupload

Additionally, current observation of operational activity indicates that MuddyWater continues to leverage open-source offensive tools in their operations. PowerShell-based tools and scripts remain central to their toolkit as well, with consistent usage observed across recent campaigns.

The group has weaponized maldocs in their recent infection chains, reverting to tactics previously observed in their historical campaign data. This represents a notable return to established Tactics, Techniques, and Procedures (TTPs)

after a period of tactical deviation.

Recent reporting suggests potential MuddyWater involvement in mobile targeting operations utilizing the DCHSpy Android malware. Despite the infrastructure overlap as mentioned in the report, the campaign's operational characteristics and targeting demonstrate significant deviation from the group's documented tradecraft, warranting further investigation and monitoring for definitive attribution.

Malware & Tools

BugSleep – Backdoor

BugSleep is a custom backdoor developed using C/C++ by MuddyWater and first discovered in mid-2024. It was deployed extensively during the second half of 2024 in campaigns targeting organizations in the Middle East, Turkey, Azerbaijan, and several European countries. Designed to execute commands and facilitate file transfers between compromised machines and its command-and-control (C&C) server, BugSleep quickly became a cornerstone of MuddyWater's toolkit.

Multiple versions of the malware were discovered with each version showing improvements and bug fixes — showing that the malware was under ongoing development by the threat actors during operations.

Technical Highlights include:

- BugSleep comes in two variations, either as a standalone executable file, or embedded inside a loader, which injects the BugSleep backdoor into legitimate processes.
- The backdoor creates a mutex named DocumentUpdater, and the loader creates PackageManager.
- When delivered via a loader, the loader injects BugSleep into one of common processes such as browsers and windows processes.
- All the configurations, strings, and C2 communications are encrypted in the same way, adding/subtracting a static value from every byte modulo 256.
- Each infected machine generates a unique identifier in the format: Computer Name/User Name.
- C2 communications use a custom pseudo-TLV based protocol (structure: [size_of_data][data]) implemented using plain TCP sockets where the first byte in the data is command number, followed by argument data corresponding to the command.

Observed Command Set:

Command Number	Argument	Description
1	File Name to upload	Upload file to C2 server
2	File Name to download	Download data from C2 server
3	N/A	Open an interactive shell until Exit /or Terminate is received
4	Integer value for timeout value	Update timeout value using setsockopt
6	N/A	Stop communication
9	N/A	Delete persistence task
10	N/A	Get persistence task status
11	N/A	Create persistence task
97	Sleep time	Update sleep time (added in updated versions)
98	Timeout Value	Update the receive timeout (added in updated versions)
99	N/A	Sends the same value back

StealthCache – Backdoor

StealthCache is a more advanced backdoor compared to BugSleep. While it has some minor similarities, it is mostly different malware. It has a rich feature set, and extended C2 communication.

The first public sample was observed to be uploaded from Israel (wtsapi.dll – 5f22f4c4fdb36c4f0ea3248abb00521e39008c1fb4c97e1b4a9c7b9ef0b691c2) and it also comes with the Fooder loader which will be described later, and its C2 domain is netivtech[.]org which we will examine in greater detail within the upcoming section (*Hunting MuddyWater in the wild*).

C2 Communication

The malware works by sending HTTP(S) requests to the C2 endpoint /aq36 at regular intervals. The server's response contains a command code that determines which action is executed (see the table below). Errors are

always sent to /q2qq32. We note that they are only mentioned in the table if the command explicitly triggers such reporting. Some commands also send log messages to other endpoints, which are not listed here as well.

Observed Commands Set:

Code	Description	Parameters	Output Endpoint
207	Executes stealer and sends results to C2	–	/mq65
350/360	Sends error to C2	–	/q2qq32
361	Send local domain host name to C2	–	/dfa65
400	Sends error to C2	–	/q2qq32
401	Sends file contents to C2	File path	/dadw
500	Sends error to C2	–	/q2qq32
501	Create file on disk	File path	–
700	Set sleep time	Sleep time	–
800	Sends EDR/Anti-Virus processes running on system from list to C2	–	/rq13
805	Show Windows credential prompt and send entered credentials to C2	–	/rq13
806	Copy self to hard-coded path	–	–
900	Deletes own executable	–	–
905	Shuts down malware	–	–
906	Restart malware	–	–

The stealer functionality in command 207 stores all stolen files in the directory results\. This directory is then compressed into CacheDump.zip via PowerShell and sent to the C2. When command code 800 is received, the malware enumerates running processes and compares them against a list of known Endpoint Detection and Response (EDR) and antivirus solutions, the list includes about 250 names. The malware does not terminate itself if such a process is found. Instead, it sends a message to the C2, informing the operators of the detected security product and continues execution.
For the command 806, the hardcoded path was

C:\Users\<username>\AppData\Local\Microsoft\Windows\PPBCompatCache\ManagerCache\WinCache.exe in the sample above.
For the command 900, the malware uses Alternate Data Streams (ADS) to delete its own executable. The ADS used is the hard-coded value :wtfbbq. This ADS is added to the executable, which allows for the executable itself to be deleted by the malware.

The malware will also check each cycle for the presence of analysis tools running on the system. It checks against a hard-coded list of processes. If any of these processes are found, the malware does not terminate itself but instead, a message is sent to the C2 at the endpoint /rq13, notifying the threat actor analysis tools that are running on the infected host.

The List of checked tools :

```
wireshark.exe
Wireshark.exe
dumpcap.exe
procmon64.exe
procmon.exe
procexp.exe
procexp64.exe
autoruns.exe
autoruns64.exe
ida.exe
DbgX.Shell.exe
WinDbg.exe
x32dbg.exe
x64dbg.exe
```

It is worth noting that the malware is designed to run on systems with a specific device name and username.

```
enc_buff = VirtualAlloc_0(0LL, ::dwSize, 0x1000u, 4u);
dec_buff = enc_buff;
if ( enc_buff )
{
```

```
 *enc_buff = v49;
 *(enc_buff + 2) = v50;
 xor_key = ::aDeviceAndComputerName;
 key_size = ::dwSize;
 *enc_buff ^= *::aDeviceAndComputerName;
 enc_buff[1] ^= xor_key[1 % key_size];
 enc_buff[2] ^= xor_key[2 % key_size];
 enc_buff[3] ^= xor_key[3 % key_size];
 enc_buff[4] ^= xor_key[4 % key_size];
 enc_buff[5] ^= xor_key[5 % key_size];
 enc_buff[6] ^= xor_key[6 % key_size];
 enc_buff[7] ^= xor_key[7 % key_size];
 enc_buff[8] ^= xor_key[8 % key_size];
 enc_buff[9] ^= xor_key[9 % key_size];
 enc_buff[10] ^= xor_key[0xA % key_size];
 enc_buff[11] ^= xor_key[0xB % key_size];
}
else
{
 dec_buff = 0LL;
}
hKernel32 = LoadLibraryW(aKernel32Dll_0);
func_ptr = GetProcAddress_0(hKernel32, dec_buff);
hAlgorithm = 0LL;
aC2Response = 0LL;
v12 = 0LL;
alloc_mem = (func_ptr)(0LL, 1LL, 0x1000LL, 4LL);
```

This code allocates memory, decrypts a function name using device/computer name as XOR key, then dynamically resolves and calls that function. The decryption will only produce the correct function name if the device name matches what the malware expects, otherwise it will crash.

Phoenix – Backdoor

Phoenix is a malware with minimalistic backdoor functionality that was attributed to MuddyWater. The malware decrypts an embedded PE file using a simple XOR algorithm, maps it into its own process memory, and transfers execution to its entry point.

```
MZ_Data = (void *)((__int64 (__fastcall *)(_QWORD, size_t, __int64, __int64))VirtualAlloc)(
                     0LL,
                     MZ_Size,
                     0x3000LL,
                     64LL);
memcpy(MZ_Data, &Encrypted_MZ_data, MZ_Size);
XOR_Decrypt((__int64)MZ_Data, MZ_Size, (__int64)Key);// "edwfwergrtgtrgt5hyhy6hghtbgbt5by6hy6nyhnhynuynunyntntyntyntntntyn"
sub_7FF68BAA1500((__int64)MZ_Data, MZ_Size, &MZ_Data_1, v10);
if ( !MZ_Data_1 )
{
  MZ_Data_1 = MZ_Data;
  v10[0] = MZ_Size;
}
SecondStage_EntryPoint = (__int64)Get_EntryPoint((__int64)MZ_Data_1);
if ( SecondStage_EntryPoint )
{
  v13 = ((__int64 (__fastcall *)(__int64))GetCurrentProcessId_0)(v6);
  PHandle = ((__int64 (__fastcall *)(__int64, _QWORD, _QWORD))OpenProcess)(1040LL, 0LL, v13);
  hHandle = (HANDLE)((__int64 (__fastcall *)(__int64, _QWORD, _QWORD, __int64, _QWORD, _DWORD, _QWORD))CreateRemoteThread)(
                       PHandle,
                       0LL,
                       0LL,
                       SecondStage_EntryPoint,
                       0LL,
                       0,
                       0LL);
  WaitForSingleObject(hHandle, 0xFFFFFFFF);
}
```

The backdoor begins by copying itself to a new location using PowerShell commands (powershell.exe -Command “Copy-Item -Path %%malware path%% -Destination ‘C:\ProgramData\Logs’ -Force”). It then generates a machine GUID by concatenating the username and computer name—similar to the technique used by BugSleep backdoor.

Next, it registers with the C2 server via the /register endpoint. After registration, it enters a C2 communication loop where it periodically sends an /iamalive beacon to the C2 and retrieves commands from the /request endpoint.

The backdoor communicates with the C2 server over HTTP and supports the following commands:

Command	Description
Cmd	Launch an interactive shell
Persisit	Install the malware using “Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders\StartUp”
timeout	Adjust the connection timeout
Upload	Write file to disk

Fooder – Loader

Fooder is a loader known to be used to load, decrypt, and run an encrypted and packed malware payload in memory. Some of the observed samples come as a DLL intended to be side-loaded by a legitimate application.

DLL Side-Loading & Multi-Threading

Upon loading the DLL version of this loader, the DllEntryPoint function is executed. The DLL then creates a new thread to load, decrypt, and run the actual payload.

Automated analysis platforms often use RunDLL to execute DLLs for analysis, but because of the multi-threading approach used by Fooder, the process will terminate prematurely, which complicates analysis and evades detection. The DLL is designed to be loaded by an application that remains active after loading the DLL, and allows for the malicious thread to live.

Payload Decryption

The loader contains an encrypted payload that is decrypted at runtime using a hard-coded key and standard Windows encryption APIs, specifically the “Microsoft Enhanced RSA and AES Cryptographic Provider.”

The decryption process involves three steps:

1. The hard-coded decryption key is hashed using CryptHashData.
2. The resulting hash is used with CryptDeriveKey to obtain the true decryption key.
3. CryptDecrypt is then used to decrypt the payload.

```
qmemcpy(szProvider, L"Microsoft Enhanced RSA and AES Cryptographic Provider",
0x6CuLL);
if ( !CryptAcquireContextW(phProv, 0LL, szProvider, 0x18u, 0xF0000000)
    || !CryptCreateHash(phProv, 0x800Cu, 0LL, 0, phHash) )
{
    goto LABEL_10;
}
if ( !CryptHashData(phHash, decryptionKey, a4, 0) )
    return GetLastError();
    if ( CryptDeriveKey(phProv, 0x660Eu, phHash, 0, phKey) )
```

Logging

A key characteristic of the loader is its use of log messages, which are printed to the console at runtime. While these messages are not visible to the user when the malicious DLL is side-loaded by its intended target executable (as it typically lacks a console window), they become visible if a custom application with an open console window is used to load the DLL.

Examples of log messages:

```
mal_StdioPrint("------------------------------------------------------");
mal_StdioPrint("********         License C**2023**C           ********");
mal_StdioPrint("------------------------------------------------------");
if ( sub_7FF8C115EA21(87LL) == 0.0 )
{
    MessageBoxA(0LL, 0LL, 0LL, 0);
    MessageBoxA(0LL, 0LL, 0LL, 0);
    MessageBoxA(0LL, 0LL, 0LL, 0);
}
sub_7FF8C115EA44(10LL);
mal_StdioPrint("------------------------------------------------------");
mal_StdioPrint("********            DLL Config               ********");
mal_StdioPrint("------------------------------------------------------");
mal_StdioPrint("------------------------------------------------------");
```

```
mal_StdioPrint("********            DLL Config               ********");
mal_StdioPrint("-------------------------------------------------------");
mal_StdioPrint("-------------------------------------------------------");
mal_StdioPrint("********            DLL Config               ********");
mal_StdioPrint("-------------------------------------------------------");
```

Additional log strings, along with the unpacking and running of the payload, were also observed:

```
if ( sub_7FF8C115C05A(89) == 0.0 )
    MessageBoxA(0LL, "Run", 0LL, 0);
v10 = j_mal_UnpackPEAndRun(v7);
Sleep(10000u);
sub_7FF8C115EA44(50LL);
sub_7FF8C115EDFA(45LL);
sub_7FF8C115EDFA(89LL);
sub_7FF8C115EA21(52LL);
mal_StdioPrint("-------------------------------------------------------");
mal_StdioPrint("********               FINISH               ********");
mal_StdioPrint("-------------------------------------------------------");
mal_StdioPrint("-------------------------------------------------------");
mal_StdioPrint("********               GOODBYE               ********");
mal_StdioPrint("-------------------------------------------------------");
```

Sleep

The loader extensively uses the Sleep function to introduce delays, primarily hindering automated analysis solutions. These sleep times are generally 1000ms (1 second) in duration, with multiple instances present throughout the malware.

Other Malware and Utilities

In addition to the aforementioned malware families, GROUP-IB has identified and attributed several other malware and tools to MuddyWater, which will be briefly mentioned in this section.

CannonRat – Remote Access Trojan
The CannonRat Remote Access Trojan (RAT) is a malicious tool designed for remote control of compromised systems. It establishes communication with a Command and Control (C2) server using the HTTP protocol, allowing attackers to execute commands, upload and download files, and dynamically load malicious DLLs. CannonRat consists of two components:

1. Primary Component – Handles communication with the C2 server and executes commands.
2. Persistence Component – Ensures the malware's persistence by executing the primary component upon system restart.

LiteInject – PE injector
LiteInject is a Portable Executable (PE) injector that accepts two command-line arguments: the target process ID and the file path of the PE file to be injected. It performs manual mapping by writing both its own shellcode and the provided module into the target process's memory.
Analysis of the sample revealed an embedded PDB path:
C:\Users\pc\Desktop\main\My_Project\Injector\x64\Debug\Injector.pdb.
Based on code characteristics and campaign context, this malware is attributed to MuddyWater with **moderate confidence**.

UDPGangster – Backdoor
UDPGangster is a basic backdoor that communicates with its command-and-control (C2) server over the UDP protocol. The malware attempts to read the C2 address from a file on disk; if the file is not found, it uses a hardcoded C2 server.
Analysis of the sample revealed the following embedded PDB path:
C:\Users\gangster\source\repos\udp_3.0 – Copy – Copy\x64\release_86\udp_3.0.pdb.

HackBrowserData
HackBrowserData is an open source project written in golang, it is a command-line tool for decrypting and exporting browser data (passwords, history, cookies, bookmarks, credit cards, download history, localStorage and extensions)

from the browser. The tool supports all major browsers on the market and is compatible with Windows, macOS and Linux.

go-socks5
go-socks5 is an open source project written in golang, it provides the socks5 package that implements a SOCKS5 server which is used to route traffic between a client and server through an intermediate proxy layer.

Yamux
Yamux is an open source project written in golang, it is a library created by HashiCorp that provides stream-oriented multiplexing, enabling multiple independent logical streams to share a single, reliable connection, such as TCP. It facilitates bidirectional stream creation, manages flow control with back-pressure, and incorporates keep-alive messages for persistent sessions. This makes it highly efficient for applications like tunneling, RPC, and proxies, where numerous streams need to coexist without the overhead of establishing multiple physical connections.

Network Infrastructure

Understanding MuddyWater's network infrastructure is key to grasping their modus operandi. By analyzing their C2 ecosystem, domain registrations, hosting providers, and related assets, we can proactively track and uncover their infrastructure before an attack. This analysis also helps prevent them from remaining undetected post-attack and aids in attribution by revealing overlaps with previous campaigns.

MuddyWater exhibits enhanced operational sophistication through its diverse infrastructure, employing both mainstream and resilient technologies and service providers. This calculated approach suggests a deliberate effort to maximize operational flexibility and evade detection. Additionally, in some operations, the group intentionally limits the C2 (command and control) server's uptime to a few days. This tactic further conceals their infrastructure and hinders efforts to trace their activities.

This section details the characteristics of the infrastructure identified since the beginning of 2025. It offers an overview of MuddyWater's known servers, service providers, and domain usage, offering insights into the technical backbone enabling their malicious campaigns.

C2 Backend and Traffic Characteristics
MuddyWater's C2 infrastructure heavily utilizes Python-based technologies, with werkzeug and uvicorn C2 handlers being the most prevalent. This suggests a strong reliance on Python in their server-side operations. Apache deployments were also observed, typically returning a 503 status code. These are believed to be decoys, not actively handling C2 clients. Werkzeug is thought to manage the StealthCache backdoor, while uvicorn is associated with the Phoenix backdoor family.

While HTTP(S) is the primary method for C2 communication for most of their custom backdoors, raw TCP and UDP backdoors have also been detected. When TCP/UDP is used, the traffic is secured with robust encryption algorithms like AES. Common HTTP ports include 80, 443, and 8080, although other less common ports have also been noted.

Infrastructure Characteristics
According to our observations, MuddyWater utilized a diverse array of hosting providers, including cloud and bulletproof hosting services such as BlueVPS, AS-COLOCROSSING, BLNWX, SEDO, HosterDaddy, Stark Industries, DIGITALOCEAN, Strato AG, AWS, OVH SAS, Scalaxy, M247, and Clouvider Limited. The diverse nature of the hosting infrastructure makes detection and attribution challenging, as there's no distinguishable pattern or preference.

Cloudflare protection was almost always observed on domains linked to MuddyWater. Domain registrations were predominantly carried out through Namecheap. For TLS certificates, Let's Encrypt and Google Trust Services were most frequently used, with DigiCert appearing in some cases.

OPSEC Practices
Despite being a prolific APT, MuddyWater often makes OPSEC mistakes that allow researchers to track and attribute their infrastructure. Reuse of domains, certificates, servers is still commonly observed when analyzing their network infrastructure.

Despite that, MuddyWater is attempting to cover their tracks, they do not keep the C2 active for too long and it is turned off when the operation is complete, we believe they use second stage network infrastructure that is used for subsequent operations on initially infected hosts.
They were observed using VPN providers such as NordVPN to bypass geofensing and detection and hide their tracks.

Pivoting Methods
To identify additional infrastructure associated with a threat actor, threat intelligence analysts commonly employ

several pivoting methods. These methods leverage shared attributes or connections between known malicious indicators to uncover previously unknown infrastructure.

Key overlaps include:

- **Shared IP Address:** If multiple malicious domains or samples resolve to the same IP address, it suggests they are part of the same infrastructure. Pivoting on this IP address can reveal other domains hosted on it or services running on that IP that might also be malicious.
- **Unique String or Pattern in HTML/Web Server Banners:** Threat actors often reuse specific code snippets, unique phrases, or custom server configurations (specific HTTP headers, server banners like “Werkzeug” or “Uvicorn” as observed with MuddyWater). Searching for these unique strings or patterns in public scanning databases (like FOFA, Shodan, Censys) can lead to discovery of additional associated infrastructure.
- **TLS Certificate Information:** Threat actors sometimes reuse TLS certificates across multiple C2 servers or domains. Pivoting on common TLS certificate attributes such as the Subject Common Name (CN), Issuer, Serial Number, or Subject Alternative Names (SANs) can expose other domains or IP addresses using the same certificate. This method is particularly effective.
- **Registrar and WHOIS Information:** While often anonymized, sometimes threat actors make mistakes or use specific registrars and registration patterns. Analyzing WHOIS data for known malicious domains (even if redacted) can sometimes reveal shared email addresses, names, or organizational patterns that lead to other related domains. Other details such as DNS servers can prove useful as well.
- **Autonomous System Numbers (ASNs):** Identifying the ASN associated with a malicious IP address can help map out the broader network space controlled by the hosting provider. This allows for searches within the same ASN for other suspicious activity or infrastructure. This attribute is useful for narrowing down the search pool.
- **File Hashes and Unique File Characteristics:** If a specific malware sample is found, its hash (SHA256, MD5) can be searched on platforms like VirusTotal to find related samples, C2 addresses, or file names. Furthermore, unique characteristics within the malware (like PDB paths, mutexes, encryption keys, or specific code patterns) can be used as pivot points to discover other related malware and their associated infrastructure.
- **Infrastructure Hosting Providers:** Observing the specific hosting providers used by an APT group (e.g., AWS, DigitalOcean, bulletproof hosting) can inform searches for other potentially malicious infrastructure hosted on the same networks, but this needs to be combined with other indicators.

Hunting MuddyWater in the Wild

This section focuses on how threat intelligence analysts can utilize publicly available tools and information for enrichment and attribution to produce actionable intelligence that can aid in defending against prolific APT groups such as MuddyWater.

We will start the hunt by taking a public report that we validated internally as a starting point. The report is made by ClearSky Cyber Security on X claiming that a suspected new malware variant linked to the MuddyWater Iranian APT group has been discovered. Based on the PDB path, it has been named Phoniex. Phoniex was used in attacks impersonating the Hungary government and Netivtech, an Israeli company.

← Post

ClearSky Cyber Security ✓
@ClearskySec

A suspected new malware variant linked to the MuddyWater Iranian APT group has been discovered. Based on the PDB path, we named it Phoniex. Phoniex was used in attacks impersonating the Hungary government and Netivtech, an Israeli company

IoCs:
Sha256:
376f96a5363f7d1d0bf269f2a35f6fa22bce52ceb6f3fedb0245534d69cfd5d6
40dead1e1d83107698ff96bce9ea52236803b15b63fb0002e0b55af71a9b5e05
ae4dee53be0ecc4a3b53174b5bf7a47a6155b16645c69c504fc4a3f7972c004b
c3afd5ce1ca50a38438bb5026cca27bfbf2d8e786e03f323adceb8ad17517eca
b99edfb9ef9e1fb4a587e4a4a66d1947739036887dd22e24602c877b0045f070
Network:
netivtech[.]org
46[.]101.36.39

Figure 2. ClearSkySec post on X

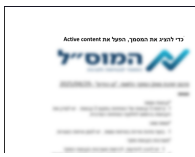
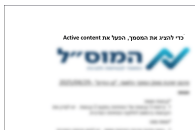
Enrichment and Pivoting

The phoenix backdoor referenced in this report (named by its developer as seen in the PDB path

D:\phonix\phoenixV3\phoenixV3\phoenixV2\x64\Release\phoenix.pdb) is a backdoor with basic functionality and its size is relatively small (around 500kb). Upon execution it creates a mutex rfgcn. Using the PDB path or the mutex value, multiple samples can be found on VirusTotal. The same approach can be applied by organizations on their internal systems to hunt the malware on internal networks.

Analysis of related files on VirusTotal suggests the infection chain begins with a Microsoft Office document containing malicious macros, and blurred decoy content. While the exact delivery mechanism remains unknown, phishing emails or messages are the most likely distribution method. example document:

SHA256 hash of the file	Screenshot of the file
c3afd5ce1ca50a38438bb5026cca27bfbf2d8e786e03f323dceb8ad17517eca	

40dead1e1d83107698ff96bce9ea52236803b15b63fb0002e0b55af71a9b5e05	
f782dfdc7ce624f98356c149fbb27f7e9b258183640943543bbf561c8af13db0	

All 3 documents appear to be timestomped, with identical creation time of 2024-07-25T23:39:00.
The embedded VBA macros in all these documents follow a similar pattern: it drops the phoenix backdoor under C:\Users\public and changes the extension and then executes it using shell32.dll.
We can see on VirusTotal that the samples communicate with the C2 on 3 endpoints specific to the phoenix backdoor:

netivtech[.]org	c
https://netivtech[.]org/register	http://46[.]101[.]36[.]39:443/imalive
https://netivtech[.]org/imalive	http://46[.]101[.]36[.]39/register
https://netivtech[.]org/request	http://46[.]101[.]36[.]39:443/request

One important observation is the sample (wtsapi32.dll – 5f22f4c4fdb36c4f0ea3248abb00521e39008c1fb4c97e1b4a9c7b9ef0b691c2) which can be seen in the relations of the C2 domain on VirusTotal, which was the newly discovered Fooder loader with embedded StealthCache backdoor that used the domain netivtech[.]org as the C2 short time before Phoenix. This can be seen on sample report on VT which shows that the sample is contacting these endpoints:

```
hxxps://netivtech[.]org/rq13
hxxps://netivtech[.]org/adad
hxxps://netivtech[.]org/aq36
```

The StealthCache backdoor maintains communication by sending periodic HTTP(S) requests to the C2 endpoint /aq36. The server's response is then parsed and contains a command code which determines the command to be executed, as previously outlined in the StealthCache malware report above.

Based on these findings, strong links can be established between Phoenix, Fooder, StealthCache, and the associated domain, all of which are attributed to MuddyWater. Consequently, any related samples and infrastructure directly related to these IOCs within a reasonable timeframe can also be linked to MuddyWater. This attribution forms a foundational basis for our subsequent hunting.

Infrastructure Analysis

Infrastructure analysis can be done using several publicly available tools. This demonstration will focus on FOFA, though it is advisable to employ multiple tools to cross-verify results and uncover additional details, as some tools may have gathered information missed by others.

This exercise is intended to illustrate hunting concepts, not to exhaustively reveal all possible details and links.Therefore we will not recursively investigate newly discovered IP/domains beyond the first pivot.

In this hunt we focus on the domain netivtech[.]org; the IP 46[.]101[.]36[.]39 returned limited contextual data and did not merit further pivoting within the scope of this demonstration.

IP and DNS information

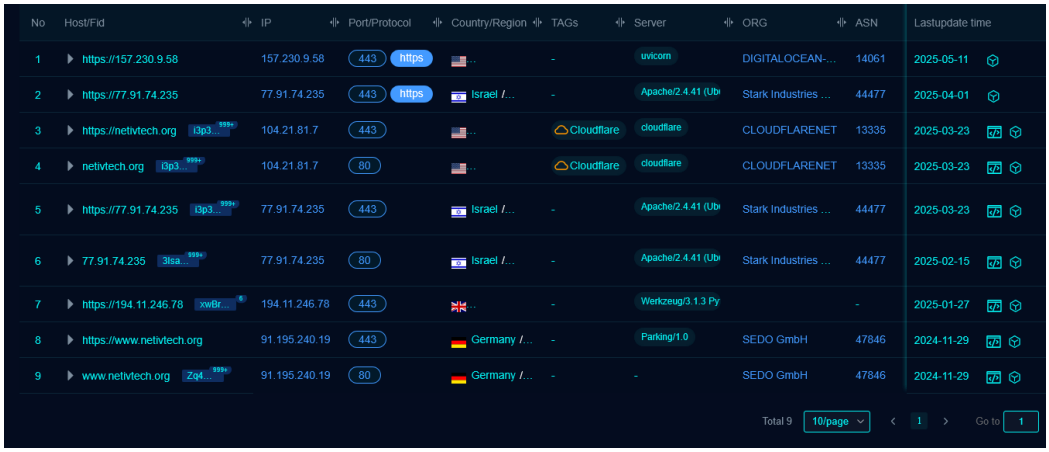
The domain netivtech[.]org was registered via NameCheap on 27 November 2024. Passive DNS records show the name initially pointing to a SEDO hosting server where the actual server is running, and finally Cloudflare protection was added.

Below we summarize the passive-DNS history:

Date (first observed)	Domain	Resolved IP	Hosting Organization
2024-11-28	netivtech[.]org	162[.]255[.]119[.]28	NameCheap
2024-11-29	www.netivtech[.]org	91[.]195[.]240[.]19	SEDO GmbH
2025-01-02	netivtech[.]org	104[.]21[.]81[.]7	Cloudflare

172[.]67[.]136[.]150

A basic keyword search on FOFA already shows useful results. These results appear because the keyword “netivtech[.]org” is present either in the banner, headers, TLS certificate or in metadata.

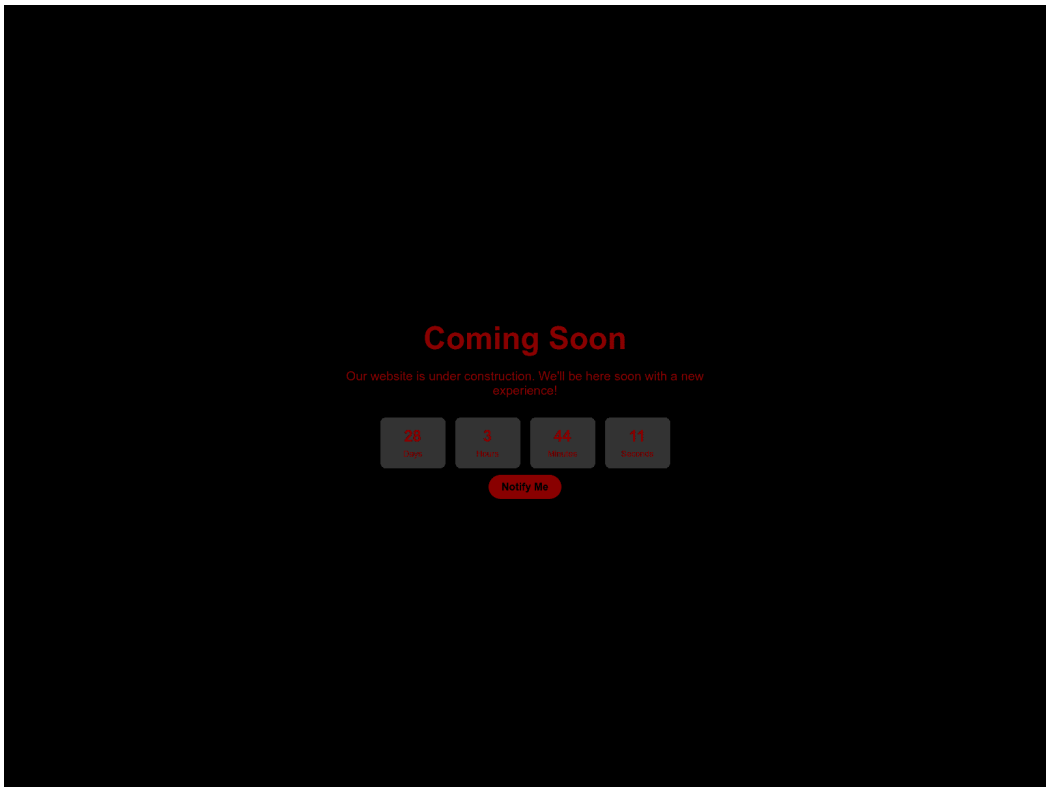


No	Host/Fid	IP	Port/Protocol	Country/Region	TAGs	Server	ORG	ASN	Lastupdate time
1	https://157.230.9.58	157.230.9.58	443 https	…	-	uvicorn	DIGITALOCEAN-…	14061	2025-05-11
2	https://77.91.74.235	77.91.74.235	443 https	Israel /…	-	Apache/2.4.41 (Ubi	Stark Industries …	44477	2025-04-01
3	https://netivtech.org i3p3…	104.21.81.7	443	…	Cloudflare	cloudflare	CLOUDFLARENET	13335	2025-03-23
4	netivtech.org i3p3…	104.21.81.7	80	…	Cloudflare	cloudflare	CLOUDFLARENET	13335	2025-03-23
5	https://77.91.74.235 i3p3…	77.91.74.235	443	Israel /…	-	Apache/2.4.41 (Ubi	Stark Industries …	44477	2025-03-23
6	77.91.74.235 3lsa…	77.91.74.235	80	Israel /…	-	Apache/2.4.41 (Ubi	Stark Industries …	44477	2025-02-15
7	https://194.11.246.78 xwBr…	194.11.246.78	443	…	-	Werkzeug/3.1.3 Py		-	2025-01-27
8	https://www.netivtech.org	91.195.240.19	443	Germany /…	-	Parking/1.0	SEDO GmbH	47846	2024-11-29
9	www.netivtech.org Zq4…	91.195.240.19	80	Germany /…	-	-	SEDO GmbH	47846	2024-11-29

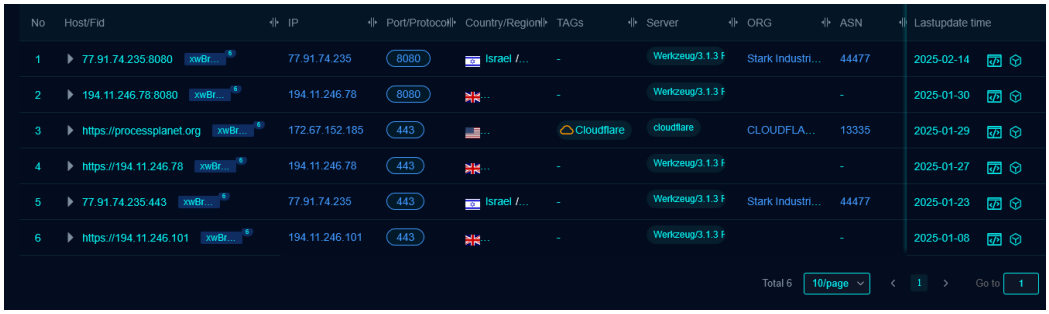
Total 9 10/page 1 Go to 1

From these results, we can see the approximate time when this domain was active. Observing the changes on the domain and running services and their contents provides valuable insight into how MuddyWater operates its infrastructure. For instance, we can see that the domain was originally resolving to an IP address on SEDO hosting, and cloudflare protection was added afterwards, confirming our observation in passive DNS records.

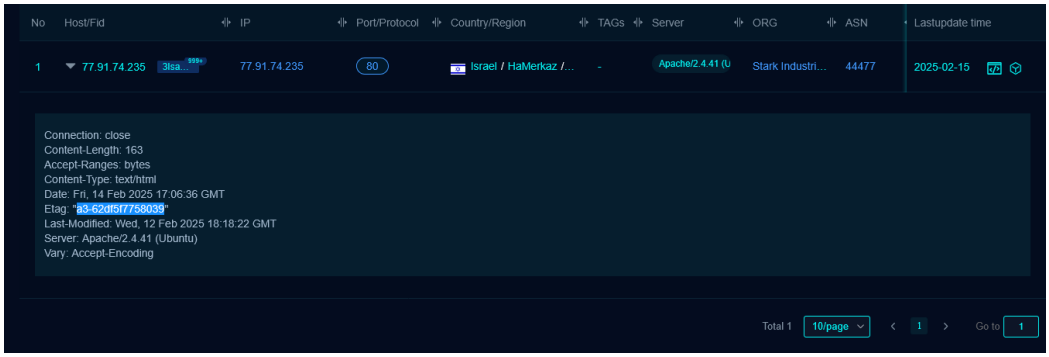
The HTML body content on 194.11.246.78:443 is unique, and has 6 pivots, which reveals additional IPs and domains (body hash: 32d299b913d5b13109d6f6117051910f5e56e74ffc8a539ecdc931d1a558c77b and title: Coming Soon).



We can search either by body hash body_hash=”-764286615” or by the FID fid=”xwBrTd+GCwK7mQQrdUg4lQ==”.



These were active in the same timeframe when StealthCache backdoor was observed in the wild, suggesting that the Werkzeug backend server was serving the StealthCache backdoor. The apache server running on 77.91.74.235:443 exposed an ETag value “a3-62df5f7758039”, which could be used to hunt more similar hosts, though in this case no additional matches were identified.



TLS Certificate Information

Inspecting TLS certificates reveals multiple certificates with the same CN “netivtech[.]org” which can be used to pivot to related infrastructure. Also, multiple C2 Backend Servers: Werkzeug, Apache, Uvicorn, each one probably serving a different C2 client.

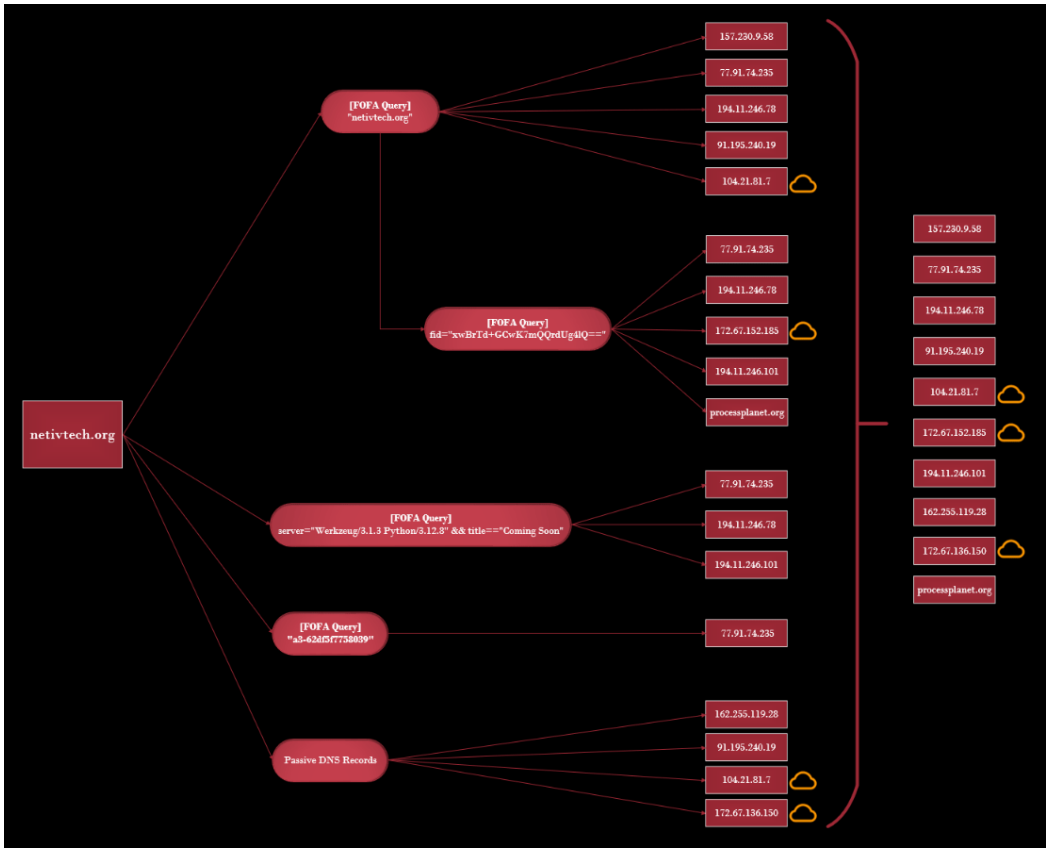
Multiple TLS certificates were found:

Issuer Org	Subject CN	IP	Validity	Serial Number
Let’s Encrypt (E5)	netivtech.org	157.230.9.58 uvicorn	Not Before: 2025-04-22 18:27 UTC Not After : 2025-07-21 18:27 UTC	457783476293808881160790836763050840512767
Let’s Encrypt (R10)	netivtech.org	77.91.74.235 apache	Not Before: 2025-02-12 17:34 UTC Not After : 2025-05-13 17:34 UTC	366936072387825346899777431974131644227331
Google Trust Services (WE1)	netivtech.org	104.21.81.7 apache	Not Before: 2025-01-28 20:53 UTC	268247748217056161689043205070074780198

Let’s Encrypt (R11)	netivtech.org	194.11.246.78 werkzeug	Not After : 2025-04-28 21:50 UTC Not Before: 2024-12-03 18:41 UTC Not After : 2025-03-03 18:41 UTC	281776326511209000864183116510500494996184
DigiCert Inc (Encryption Everywhere DV TLS CA – G2)	www.netivtech.org	91.195.240.19 parking	Not Before: 2024-11-28 00:00 UTC Not After : 2025-11-27 23:59 UTC	145932943371486894480180240417122610 55

On FOFA, no additional IPs were discovered when searching by the serial numbers of these certificates, that’s because our initial keyword search revealed them all. However, these serial numbers can be used on other platforms where additional IPs can be discovered.
So far, 10 IOCs have been collected, three of which are Cloudflare IPs that can be excluded as they are not useful:

```
157.230.9.58
77.91.74.235
194.11.246.78
91.195.240.19
194.11.246.101
162.255.119.28
processplanet.org
104.21.81.7 (CF)
172.67.152.185 (CF)
172.67.136.150 (CF)
```



Each of the resulting IOCs can be taken further for deeper investigation, applying the same concept recursively until reaching all dead ends. It is worth noting that this investigation led to the discovery of StealthCache malware.

Conclusion

MuddyWater remains one of the most significant state-aligned threats in the Middle East, with increasing activity in Europe, and the United States. Its targeting aligns with Iranian geopolitical objectives, focusing on high-value industries including telecommunications, government, energy, defense, and critical infrastructure, often seeking long-term persistence enabling both intelligence collection and potential disruptive operations.

The group has steadily increased its sophistication, shifting TTPs, and weaponizing new malware, and tools. While phishing remains the primary vector for initial access, MuddyWater has diversified its toolkit with custom malware families and operational security. Their recent malware includes custom backdoors (BugSleep, StealthCache, Phoenix), the Fooder loader, and open-source tools. They also continue to use malicious PowerShell scripts.

Infrastructure analysis shows that MuddyWater relies on a variety of services and various commercial and bulletproof hosting providers. This variety complicates detection and attribution, as there is no clear preference or pattern. However, tracking them is not impossible—as was demonstrated in this report. Their constant refinement of tools, infrastructures, and techniques alongside counter-attribution methods, indicates that MuddyWater will remain a **persistent and adaptive threat**, especially in regions and sectors tied to Iranian strategic interests.

Recommendations

Organizations can reduce exposure to MudduWater operations by implementing the following measures:

- **Threat Intelligence:** Stay updated on MuddyWater TTPs and subscribe to reliable Threat Intelligence feeds to receive the latest actionable intelligence.
- **Threat Hunting**: Proactively hunt for MuddyWater infrastructure using the latest TTPs and IOCs highlighted in this or other reports.
- **Phishing Awareness:** Conduct regular employee training on recognizing and reporting phishing and social engineering attempts.
- **Macro Controls** : Disable Office macros by default; Enforce GPO policies to restrict execution from untrusted sources, and only permit digitally signed macros if essential.
- **Endpoint Security:** Deploy and properly configure EDR and Anti-Virus technologies to detect and block MuddyWater malwares on hosts.

- **Implement Multi-Factor Authentication (MFA):** Enforce MFA on all critical accounts to mitigate credential theft risk and unauthorized access.
- **File System Monitoring:** Monitor for suspicious directory creation under the Public user folder.
- **Network Defense:** Inspect and baseline outbound network traffic. Hunt for anomalous communications matching known MuddyWater C2 patterns and domains.