WildPressure targets industrial-related entities in the Middle East

securelist.com/wildpressure-targets-industrial-in-the-middle-east/96360

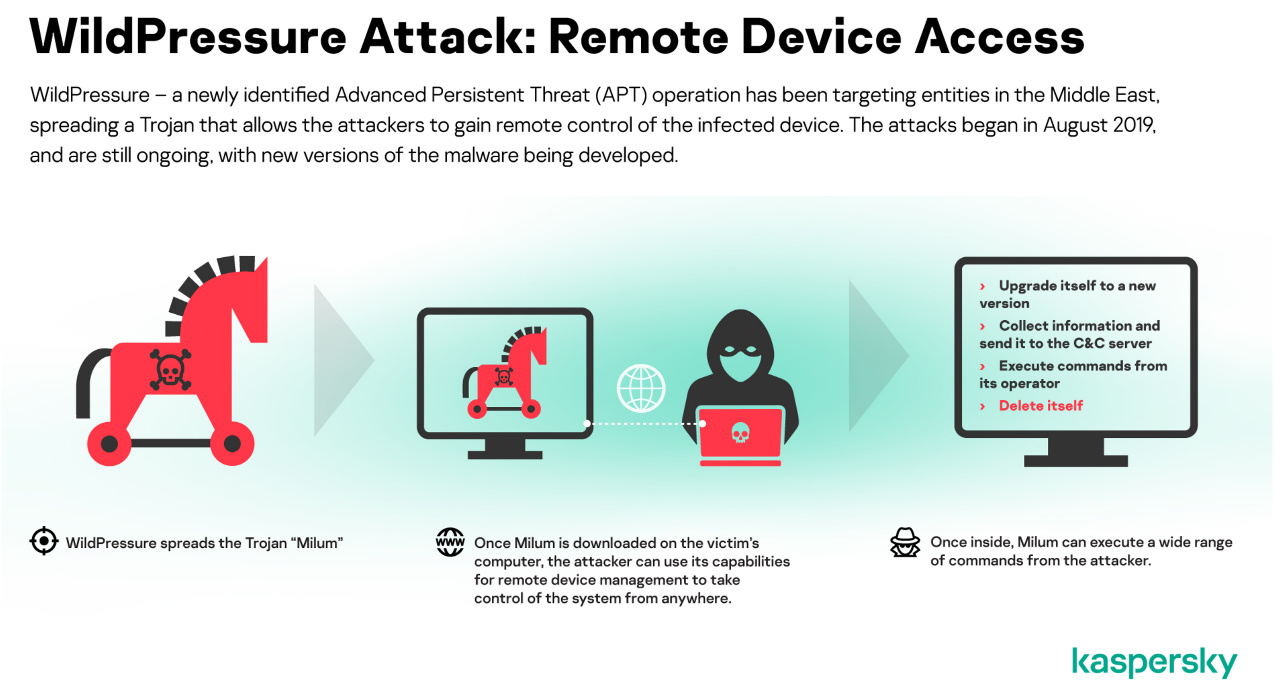
By Denis Legezo

In August 2019, Kaspersky discovered a malicious campaign distributing a fully fledged C++ Trojan that we call Milum. All the victims we registered were organizations from the Middle East. At least some of them are related to industrial sector. Our Kaspersky Threat Attribution Engine (KTAE) doesn't show any code similarities with known campaigns. Nor have we seen any target intersections. In fact, we found just three almost unique samples, all in one country. So we consider the attacks to be targeted and have currently named this operation WildPressure.

The compilation timestamps for all these files is the same – March 2019. This is consistent with the fact that we registered no infections before May 31, 2019, so the compilation dates don't seem to be spoofed. For their campaign infrastructure, the operators used rented OVH and Netzbetrieb virtual private servers (VPS) and a domain registered with the Domains by Proxy anonymization service.

The malware uses the JSON format for configuration data and as a C2 communication protocol over HTTP as well. Inside the encrypted communications within the HTTP POST requests, we found several interesting fields. One of them shows the malware version – 1.0.1. A version number like this indicates an early stage of development. Other fields suggest the existence of, at the very least, plans for non-C++ versions.

The only encryption implemented is the RC4 algorithm with different 64-byte keys for different victims. Also, the developers were kind enough to leave RTTI data inside the files. Kaspersky products detect this malware as Backdoor.Win32.Agent. For more information, please contact: intelreports@kaspersky.com



WildPressure Attack: Remote Device Access

WildPressure – a newly identified Advanced Persistent Threat (APT) operation has been targeting entities in the Middle East, spreading a Trojan that allows the attackers to gain remote control of the infected device. The attacks began in August 2019, and are still ongoing, with new versions of the malware being developed.

- Upgrade itself to a new version
- Collect information and send it to the C&C server
- Execute commands from its operator
- Delete itself

WildPressure spreads the Trojan "Milum"

Once Milum is downloaded on the victim's computer, the attacker can use its capabilities for remote device management to take control of the system from anywhere.

Once inside, Milum can execute a wide range of commands from the attacker.

Why we call it Milum and why it's of interest

All the aforementioned C++ Trojans are compiled as standalone PE files, originally named Milum46_Win32.exe. The word 'milum' is used in the C++ class names inside the malware, so we named the Trojan after it.

Another distinctive characteristic is that the malware exports lots of zlib compression functions, such as zlibVersion(), inflate() or deflate(). This compression is needed for C2 communication, but in reality there is no need to export them in the case of a standalone application.

The JSON configuration fields are not limited to just the version and programming language; the campaign operators also use target IDs that are found in the samples. Among them, we found HatLandM30 and HatLandid3 – neither of which we are familiar with. The following table provides Milum samples that have similar PE header compilation timestamps but different target IDs:

Milum46_Win32.exe sample MD5 hash	Timestamp (GMT)	clientid
0C5B15D89FDA9BAF446B286C6F97F535	2019.03.09 06:17:19	839ttttttt
17B1A05FC367E52AADA7BDE07714666B	2019.03.09 06:17:19	HatLandid3
A76991F15D6B4F43FBA419ECA1A8E741	2019.03.09 06:17:19	HatLandM30

Rather than describing all the configuration fields one by one, we have gathered them together in the following table, with all the main characteristics for this malware family:

Programming language	C++ with STL functions used mostly to parse JSON data and exception handling.
Configuration data	Base64-encoded JSON data in PE resources. Includes timeouts, C2 URLs and keys for communication, including RC4 64-byte key.
Network protocol	Trojan transmits compressed JSON data in HTTP POST requests with gzip, base64-encoded and RC4 encrypted.
Beacon data	Encrypted JSON contains the malware version "1.0.1", Epoch timestamp and client id. It also has specific fields such as "vt" and "ext" that correspond to programming language "c++" and file extension "exe". If our hypothesis is correct, this suggests that non-C++ Trojan versions may be planned, if not already implemented.
Persistence	HKCU autorun system registry keys Run and RunOnce.

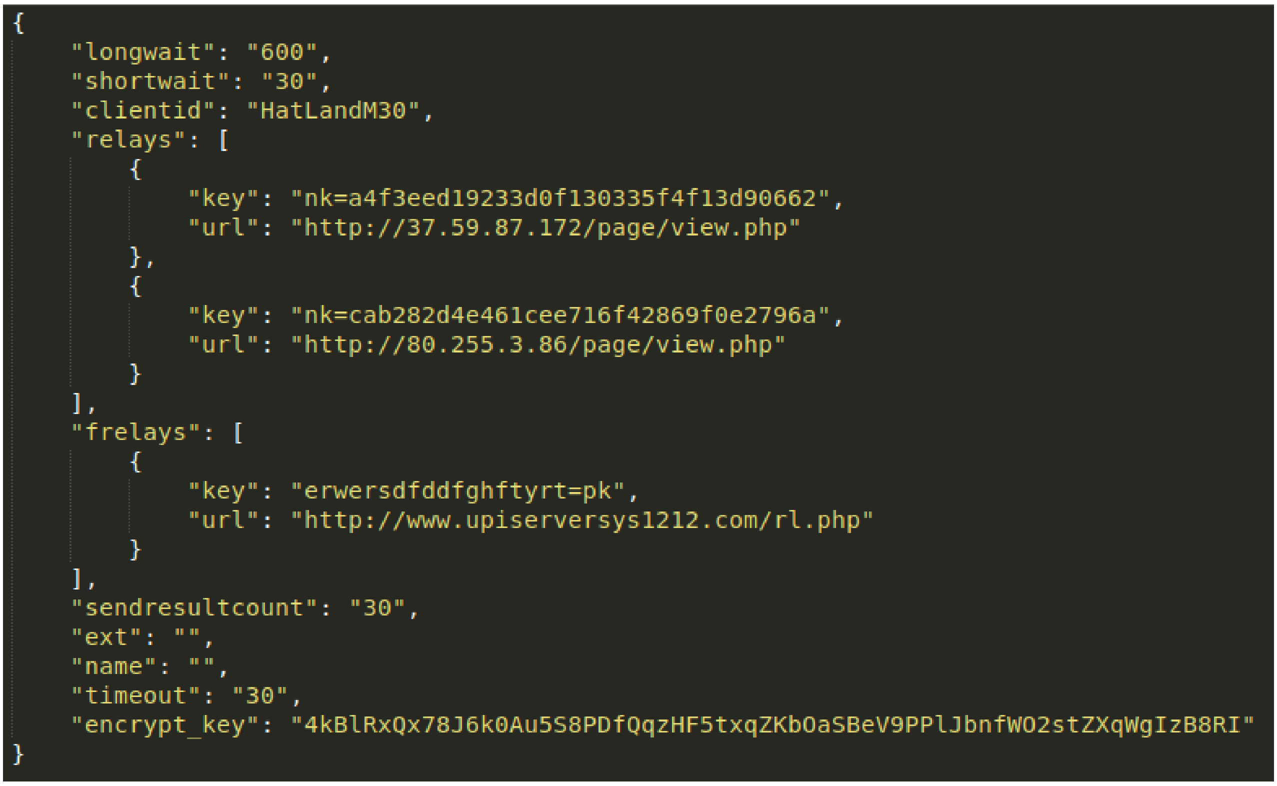
Encryption	The communication encryption used is RC4 with the 64-byte key stored in the configuration data.
Compression	For compression the Trojan uses an embedded gzip code. For some reason gzip functions are exported from PE, although the samples are standalone executables, not DLLs.

Let’s dig a little deeper inside

The most popular sample in our telemetry was:

MD5	0c5b15d89fda9baf446b286c6f97f535
Compiled	2019.03.09 06:17:19 (GMT)
Size	520704
Internal name	Milum46_Win32.exe

This application exists as an invisible toolbar window. The main malicious functions are implemented in a separate thread. Milum decodes its configuration data and, besides timeouts, it gets the parameters “clientid” and “encrypt_key” to use in RC4 encryption.



```
{
    "longwait": "600",
    "shortwait": "30",
    "clientid": "HatLandM30",
    "relays": [
        {
            "key": "nk=a4f3eed19233d0f130335f4f13d90662",
            "url": "http://37.59.87.172/page/view.php"
        },
        {
            "key": "nk=cab282d4e461cee716f42869f0e2796a",
            "url": "http://80.255.3.86/page/view.php"
        }
    ],
    "frelays": [
        {
            "key": "erwersdfddfghftyrt=pk",
            "url": "http://www.upiserversys1212.com/rl.php"
        }
    ],
    "sendresultcount": "30",
    "ext": "",
    "name": "",
    "timeout": "30",
    "encrypt_key": "4kBlRxQx78J6k0Au5S8PDfQqzHF5txqZKbOaSBeV9PPlJbnfWO2stZXqWgIzB8RI"
}
```

Example of the decoded and beautified configuration data. The “clientid” field differs in every sample observed

The following table describes the different configuration parameters:

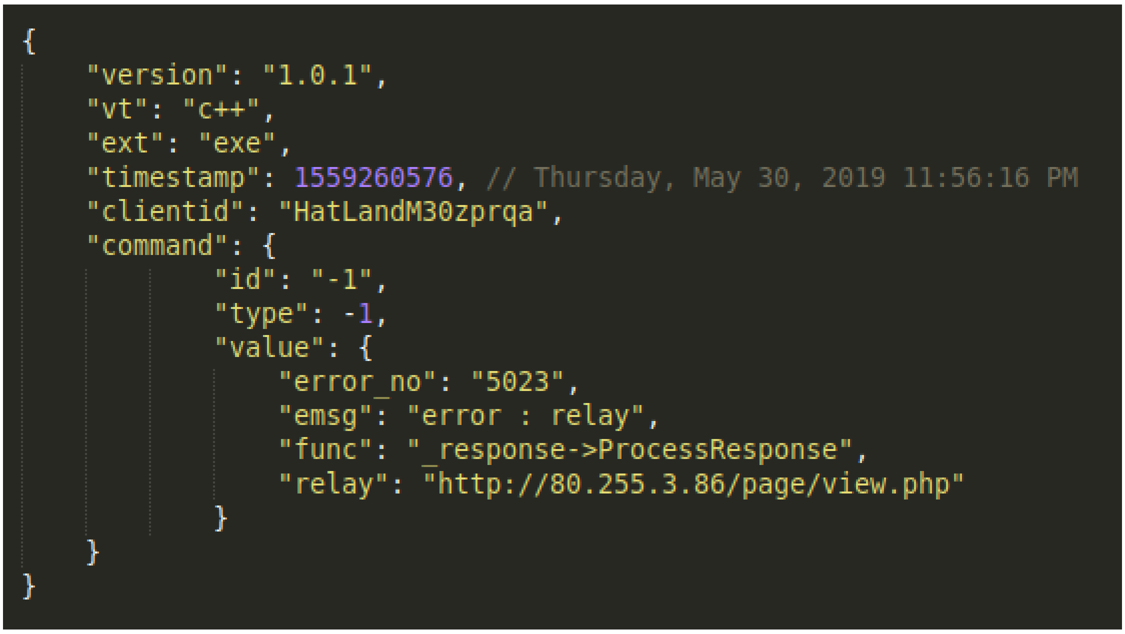
Config parameter	Parameter features
shortwait	Pause in milliseconds between C2 communication working cycles
clientid	Unique ASCII target name
encrypt_key	RC4 encryption key for JSON-based C2 communications
relays – url	Full URL to send HTTP POST beacon and GET commands
relays – key	Unique ASCII key for each C2 to communicate with it

The operators can run the Trojan using the key (“b” or “B”) as the first argument and the file name as the second. In this case, Milum will delete the file sent as a parameter. Then the Trojan will create the C:\ProgramData\Micapp\Windows\ directory and parse its configuration data to form the beacon to send to its C2.

To send the beacon, Milum uses the HTTP POST request with three parameters as enumerated in the table below.

Beacon parameter	Parameter values
md	Clientid from config, with prefix 01011 and random five-character ASCII suffix
nk	Key from config to communicate with C2, differs for each server
val	Compressed, encrypted and encoded command JSON data

The first two parameters are taken from the configuration data. The third one is encrypted and after decryption, decompression, decoding and beautifying, it looks like this:



```
{
    "version": "1.0.1",
    "vt": "c++",
    "ext": "exe",
    "timestamp": 1559260576, // Thursday, May 30, 2019 11:56:16 PM
    "clientid": "HatLandM30zprqa",
    "command": {
            "id": "-1",
            "type": -1,
            "value": {
                "error_no": "5023",
                "emsg": "error : relay",
                "func": "_response->ProcessResponse",
                "relay": "http://80.255.3.86/page/view.php"
            }
    }
}
```

Decoded and beautified JSON beacon to C2. In this case, the connection to the first server was unsuccessful

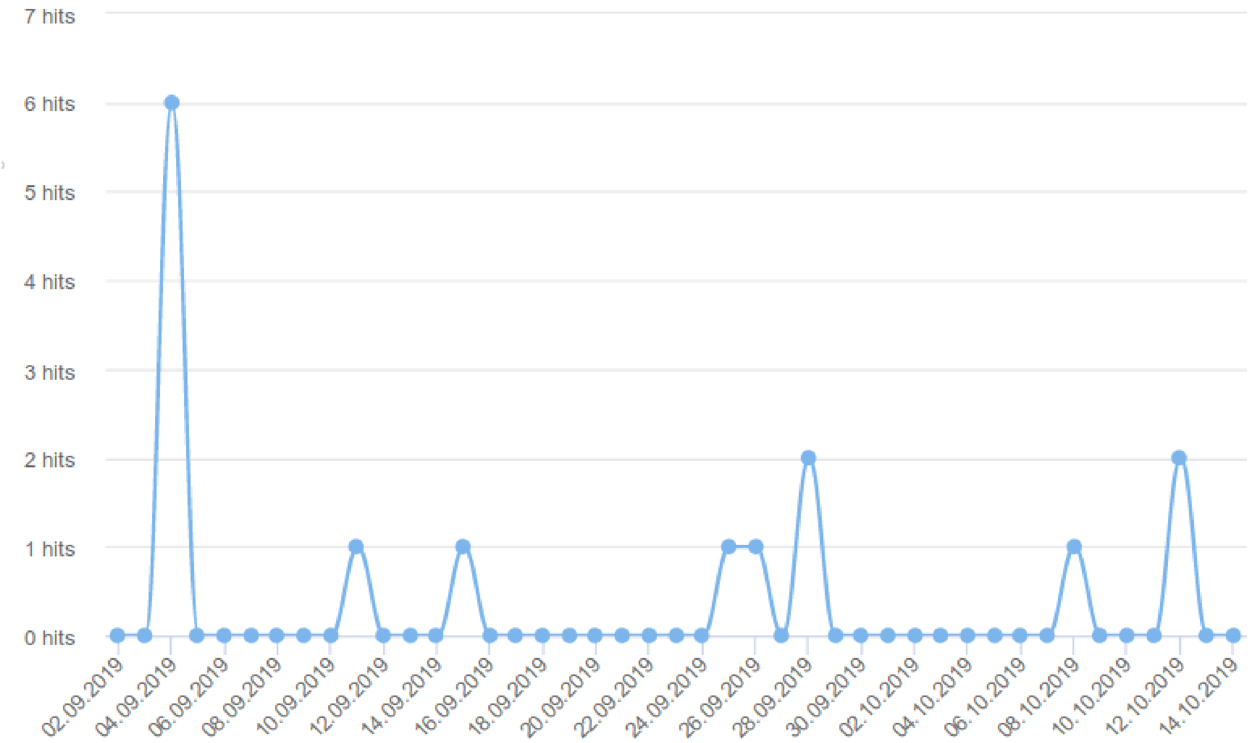
There are several fields worth mentioning here. We referred above to different programming languages besides C++: “vt” seems to reference a programming language and “ext” a file extension. The only reason that we could think of for keeping these is if the attackers have several Trojans, written in different languages, to work with the same control server.

Regarding the “command” field, the control servers were inaccessible at the time of the analysis, so we don’t have commands from them. However, we analyzed the command handlers in Milum’s code as described below:

Code	Meaning	Features
1	Execution	Silently execute received interpreter command and return result through pipe
2	Server to client	Decode received content in “data” JSON field and drop to file mentioned in “path” field
3	Client to server	Encode file mentioned in received command “path” field to send it
4	File info	Get file attributes: hidden, read only, archive, system or executable
5	Cleanup	Generate and run batch script to delete itself
6	Command result	Get command execution status
7	System information	Validate target with Windows version, architecture (32- or 64-bit), host and user name, installed security products (with WQL request “Select From AntiVirusProduct WHERE displayName <>’Windows Defender'”)
8	Directory list	Get info about files in directory: hidden, read only, archive, system or executable
9	Update	Get the new version and remove the old one

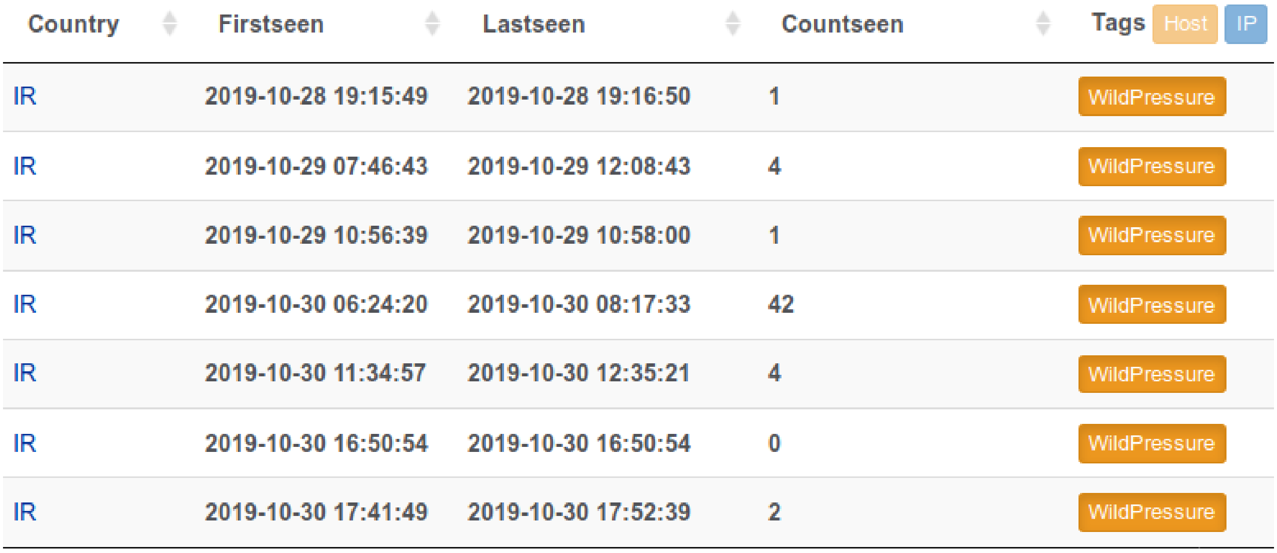
Who was attacked?

According to our telemetry, the Milum Trojan was exclusively used to attack targets in the Middle East from at least the end of May 2019.



Number of detections for one of the samples from September 2019

We were able to sinkhole one of the WildPressure C2 domains (upiserversys1212[.]com) in September 2019. The vast majority of visitor IPs were also from the Middle East, and we believe the rest were network scanners, TOR exit nodes or VPN connections.



Country	Firstseen	Lastseen	Countseen	Tags Host IP
IR	2019-10-28 19:15:49	2019-10-28 19:16:50	1	WildPressure
IR	2019-10-29 07:46:43	2019-10-29 12:08:43	4	WildPressure
IR	2019-10-29 10:56:39	2019-10-29 10:58:00	1	WildPressure
IR	2019-10-30 06:24:20	2019-10-30 08:17:33	42	WildPressure
IR	2019-10-30 11:34:57	2019-10-30 12:35:21	4	WildPressure
IR	2019-10-30 16:50:54	2019-10-30 16:50:54	0	WildPressure
IR	2019-10-30 17:41:49	2019-10-30 17:52:39	2	WildPressure

C2 domain sinkholing also shows active infections mostly from the Middle East

And who’s behind it?

To date we haven’t observed any strong code- or victim-based similarities with any known actor or set of activity. Their C++ code is quite common, regarding configuration data and communication protocol malware uses base64-encoded JSON-formatted configuration data stored in the binary’s resource section and parses it with Standard Template Library (STL) functions. However, these commonalities are not conclusive enough for attribution and our hypothesis is that they are merely coincidence. We will continue to monitor this activity

To sum up

To date, we don't have any data regarding Milum's spreading mechanism. A campaign that is, apparently, exclusively targeting entities in the Middle East (at least some of them are industrial-related) is something that automatically attracts the attention of any analyst. Any similarities should be considered weak in terms of attribution, and may simply be techniques copied from previous well-known cases. Indeed, this "learning from more experienced attackers" cycle has been adopted by some interesting new actors in recent years.

We should also be cautious regarding the true targeting of this new set of activities, as it is probably too soon to jump to conclusions. The targeted nature seems to be clear, but the targeting itself might be limited by our own visibility. The malware is not exclusively designed against any kind of victim in particular and might be reused in other operations.

Indicators of compromise

Files MD5
0C5B15D89FDA9BAF446B286C6F97F535
17B1A05FC367E52AADA7BDE07714666B
A76991F15D6B4F43FBA419ECA1A8E741
Original file names are Milum46_Win32.exe; on the target side they exist as system32.exe

URLs
upiserversys1212[.]com/rl.php
37.59.87[.]172/page/view.php
80.255.3[.]86/page/view.php