# DirtyMoe: Introduction and General Overview of Modularized Malware

decoded.avast.io/martinchlumecky/dirtymoe-1/

June 16, 2021


by Martin ChlumeckýJune 16, 202121 min read

## Abstract

The rising price of the cryptocurrency has caused a skyrocketing trend of malware samples in the wild. DDoS attacks go hand in hand with the mining of cryptocurrencies to increase the attackers' revenue/profitability. Moreover, the temptation grows if you have thousands of victims at your disposal.

This article presents the result of our recent research on the DirtyMoe malware. We noticed that the NuggetPhantom malware [1] had been the first version of DirtyMoe, and PurpleFox is its exploit kit [2]. We date the first mention of the malware to 2016. The malware has followed a fascinating evolution during the last few years. The first samples were often unstable and produced obvious symptoms. Nonetheless, the current samples are at par with other malware in the use of anti-forensic, anti-tracking, and anti-debugging techniques nowadays.

The DirtyMoe malware uses a simple idea of how to be modularized, undetectable, and untrackable at the same time. The aim of this malware is focused on Cryptojacking and DDoS attacks. DirtyMoe is run as a Windows service under system-level privileges via EternalBlue and at least three other exploits. The particular functionality is controlled remotely by the malware authors who can reconfigure thousands of DirtyMoe instances to the desired functionality within a few hours. DirtyMoe just downloads an encrypted payload, corresponding to the required functionality, and injects the payload into itself.

DirtyMoe's self-defense and hiding techniques can be found at local and network malware layers. The core of the DirtyMoe is the service that is protected by VMProtect. It extracts a Windows driver that utilizes various rootkit capabilities such as service, registry entry, and driver hiding. Additionally, the driver can hide selected files on the system volume and can inject an arbitrary DLL into each newly created process in the system. The network communication with a mother server is not hard-coded and is not invoked directly. DirtyMoe makes a DNS request to one hard-

coded domain using a set of hardcoded DNS servers. However,  the final IP address and port are derived using another sequence of DNS requests. So, blocking one final IP address does not neutralize the malware, and we also cannot block DNS requests to DNS servers such as Google, Cloudflare, etc.

This is the first article of the DirtyMoe series. We present a high-level overview of the complex DirtyMoe's architecture in this article. The following articles will be focused on detailed aspects of the DirtyMoe architecture, such as the driver, services, modules, etc. In this article, We describe DirtyMoe's kill chain, sophisticated network communication, self-protecting techniques, and significant symptoms. Further, we present statistical information about the prevalence and location of the malware and C&C servers. Finally, we discuss and summarize the relationship between DirtyMoe and PurpleFox. Indicators of Compromise lists will be published in future posts for each malware component.

## 1. Kill Chain

The following **Figures** illustrate two views of DirtyMoe architecture. The first one is delivery and the second one is exploration including installation. Individual Kill Chain items are described in the following subsections.

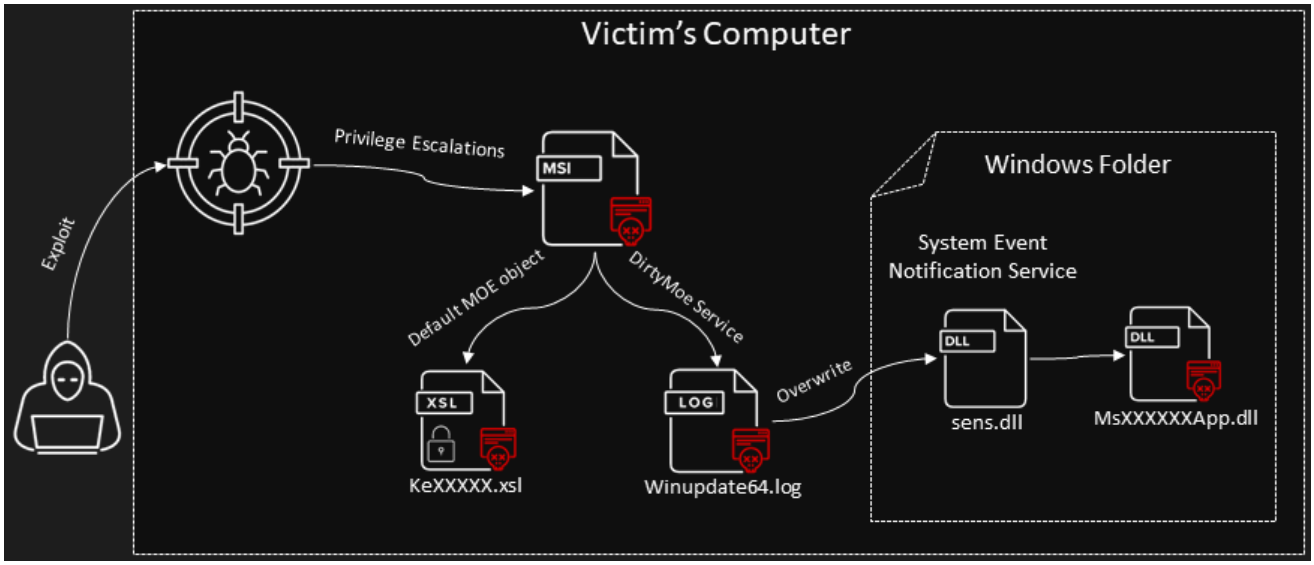


**Figure 1. Kill** Chain: Delivery

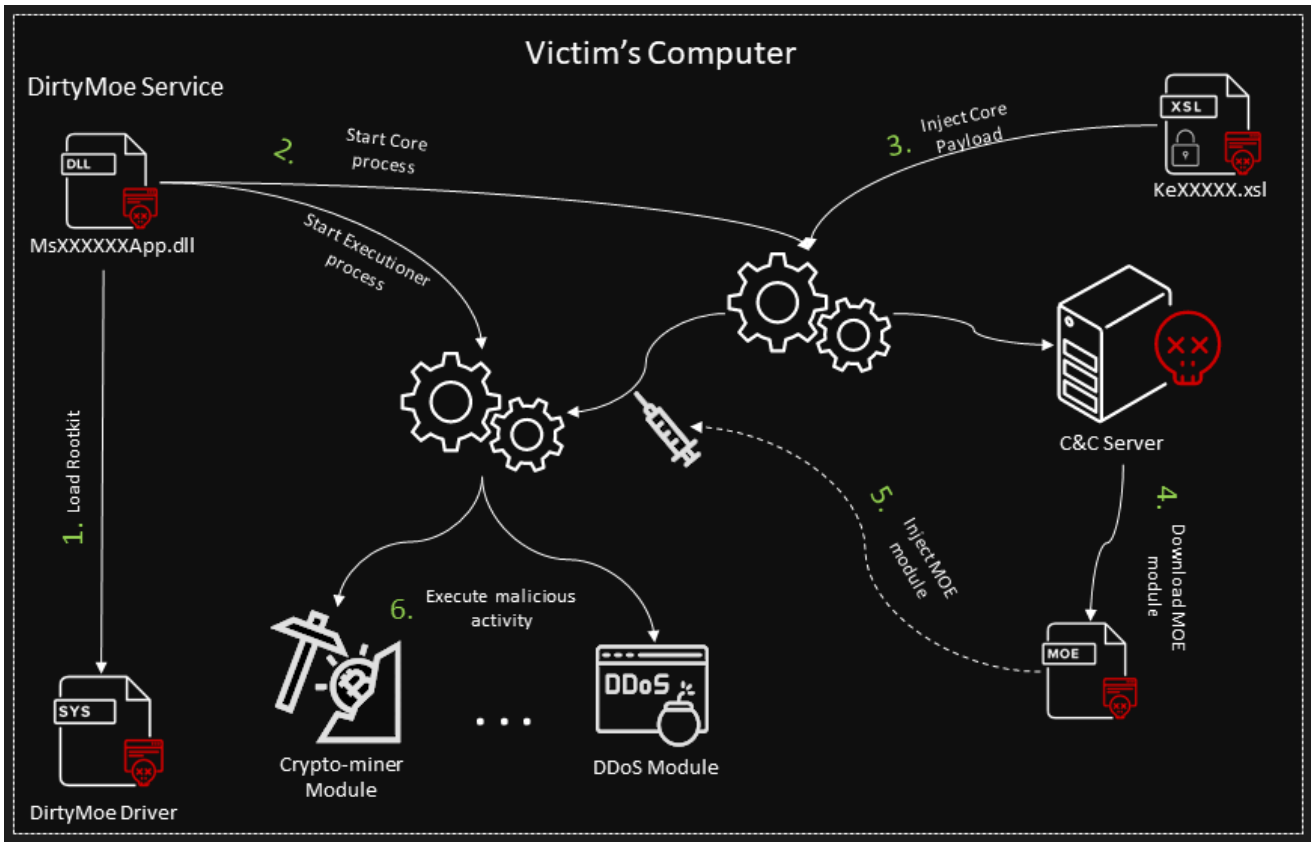


**Figure 2.** Kill Chain: Exploitation and Installation

## 1.2 Reconnaissance

Through port-scanning and open-database of vulnerabilities, the attackers find and target a large number of weak computers. PurpleFox is the commonly used exploit kit for DirtyMoe.

## 1.3 Weaponization

The DirtyMoe authors apply several explicit techniques to gain admin privileges for a DirtyMoe deployment.

One of the favorite exploits is EternalBlue ( `CVE-2017-0144` ), although this vulnerability is well known from 2017. Avast still blocks around 20 million EternalBlue attack attempts every month. Unfortunately, a million machines still operate with the vulnerable SMBv1 protocol and hence opening doors to malware include DirtyMoe via privilege escalation [3]. Recently, a new infection vector that cracks Windows machines through SMB password brute force is on the rise [2].

Another way to infect a victim machine with DirtyMoe is phishing emails containing URLs that can exploit targets via Internet Explorer; for instance, the Scripting Engine Memory Corruption Vulnerability ( `CVE-2020-0674` ). The vulnerability can inject and damage memory so that executed code is run in the current user context. So, if the attacker is lucky enough to administrator user login, the code takes control over the affected machine [4]. It is wondering how many users frequently use Internet Explorer with this vulnerability.

Further, a wide range of infected files is a way to deploy DirtyMoe. Crack, Keygen, but even legit-looking applications could consist of malicious code that installs malware into victim machines as a part of their execution process. Customarily, the malware installation is done in the background and silently without user interaction or user's knowledge. Infected macros, self-extracting archive, and repacked installers of popular applications are the typical groups.

We have given the most well-known examples above, but how to deploy malware is unlimited. All ways have one thing in common; it is Common Vulnerabilities and Exposures (CVE). Description of the exact method to install DirtyMoe is out of this article's scope. However, we list a shortlist of used CVE for DirtyMoe as follows:

- `CVE-2019-1458`, `CVE-2015-1701`, `CVE-2018-8120`: Win32k Elevation of Privilege Vulnerability
- `CVE-2014-6332`: Windows OLE Automation Array Remote Code Execution Vulnerability

## 1.4 Delivery

When one of the exploits is successful and gains system privileges, DirtyMoe can be installed on a victim's machine. We observe that DirtyMoe utilizes Windows MSI Installer to deploy the malware. MSI Installer provides an easy way to install proper software across several platforms and versions of Windows. Each version requires a different location of installed files and registry entries. The malware author can comfortably set up DirtyMoe configurations for the target system and platform.

### 1.4.1 MSI Installer Package

The malware authors via MSI installer prepare a victim environment to a proper state. They focus on disabling anti-spyware and file protection features. Additionally, the MSI package uses one system feature which helps to overwrite system files for malware deployment.

#### Registry Manipulation

There are several registry manipulations during MSI installation. The most crucial registry entries are described in the following overview:

- Microsoft Defender Antivirus can be disabled by the `DisableAntiSpyware` registry key set to `1`.
  `HKCU\SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware`
- Windows File Protection (WFP) of critical Windows system files is disabled.
  - `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SFCDisable`
  - `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SFCScan`
- Disables SMB sharing due to the prevention of another malware infection
  `HKLM\SYSTEM\CurrentControlSet\Services\NetBT\Parameters\SMBDeviceEnabled`

- Replacing system files by the malware ones
  - `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AllowProtectedRenames`
  - `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\PendingFileRenameOperations`
- Sets the threshold to the maximum value to ensure that each service is run as a thread and the service is not a normal process.
  `HKLM\SYSTEM\CurrentControlSet\Control\SvcHostSplitThresholdInKB`

#### File Manipulation

The MSI installer copies files to defined destinations if an appropriate condition is met, as the following table summarizes.

| File | Destination | Condition |
|---|---|---|
| sysupdate.log | WindowsFolder | Always |
| winupdate32.log | WindowsFolder | NOT VersionNT64 |
| winupdate64.log | WindowsFolder | VersionNT64 |

Windows Session Manager ( `smss.exe` ) copies the files during Windows startup according to registry value of `PendingFileRenameOperations` entry as follows:

- Delete (if exist) `\\[WindowsFolder]\\AppPatch\\Acpsens.dll`
- Move `\\[SystemFolder]\\sens.dll` to `\\[WindowsFolder]\\AppPatch\\Acpsens.dll`
- Move `\\[WindowsFolder]\\winupdate[64,32].log` to `\\[SystemFolder]\\sens.dll`
- Delete (if exist) `\\[WindowsFolder]\\AppPatch\\Ke583427.xsl`
- Move `\\[WindowsFolder]\\sysupdate.log` to `\\[WindowsFolder]\\AppPatch\\Ke583427.xsl`

### 1.4.2 MSI Installation

The MSI installer can be run silently without the necessity of the user interaction. The installation requires the system restart to apply all changes, but the MSI installer supports delayed restart, and a user does not detect any suspicious symptoms. The MSI installer prepares all necessary files and configurations to deploy the DirtyMoe malware. After the system reboot, the system overwrites defined system files, and the malware is run. We will describe the detailed analysis of the MSI package in a separate article.

## 1.5 Exploitation

The MSI package overwrites the system file `sens.dll` via the Windows Session Manager. Therefore, DirtyMoe abuses the Windows System Event Notification (SENS) to be started by the system. The infected SENS Service is a mediator which deploys a DirtyMoe Service.

In the first phase, the abused service ensures that DirtyMoe will be loaded after the system reboot and restores the misused Windows service to the original state. Secondly, the main DirtyMoe service is started, and a rootkit driver hides the DirtyMoe's services, registry entries, and files.

### 1.5.1 Infected SENS Service

Windows runs the original SENS under NT AUTHORITY\SYSTEM user. So, DirtyMoe gains system-level privileges. DirtyMoe must complete three basic objectives on the first run of the infected SENS Service.

1. Rootkit Driver Loading
   DirtyMoe runs in user mode, but the malware needs support from the kernel space. Most kernel actions try to hide or anti-track DirtyMoe activities. A driver binary is embedded in the DirtyMoe service that loads the driver during DirtyMoe starts.
2. DirtyMoe Service Registration
   The infected SENS Service registers the DirtyMoe Service in the system as a regular Windows service. Consequently, the DirtyMoe malware can survive the system reboot. The DirtyMoe driver is activated at this time; hence the newly created service is hidden too. DirtyMoe Service is the main executive unit that maintains all malware activities.
3. System Event Notification Service Recovery
   If the DirtyMoe installation and service deployment is successful, the last step of the infected SENS Service is to recover the original SENS service from the `Acpsens.dll` backup file; see File Manipulation.

### 1.5.2 DirtyMoe Service

DirtyMoe Service is registered in the system during the maware's deployment. At the start, the service extracts and loads the DirtyMoe driver to protect itself. When the driver is loaded, the service cleans up all evidence about the driver from the file system and registry.

The DirtyMoe malware is composed of two processes, namely *DirtyMoe Core* and *DirtyMoe Executioner*, created by DirtyMoe service that terminates itself if the processes are created and configured. DirtyMoe Service creates the *Core* and *Executioner* process through several processes and threads that are terminated over time. Forensic and tracking methods are therefore more difficult. The DirtyMoe services, processes, and their details will be described in the future article in detail.

## 1.6 Installation

All we described above is only preparation for malicious execution. *DirtyMoe Core* is the maintenance process that manages downloading, updating, encrypting, backuping, and protecting DirtyMoe malware. *DirtyMoe Core* is responsible for the injection of malicious code into *DirtyMoe Executioner* that executes it. The injected code, called MOE Object or Module, is downloaded from a C&C server. MOE Objects are encrypted PE files that *DirtyMoe Core* decrypts and injects into *DirtyMoe Executioner*.

Additionally, the initial payload that determines the introductory behavior of *DirtyMoe Core* is deployed by the MSI Installer as the `sysupdate.log` file. Therefore, *DirtyMoe Core* can update or change its own functionality by payload injection.

## 1.7 Command and Control

*DirtyMoe Core* and *DirtyMoe Executioner* provide an interface for the malware authors who can modularize DirtyMoe remotely and change the purpose and configuration via MOE Objects. *DirtyMoe Core* communicates with the command and control (C&C) server to obtain attacker commands. Accordingly, the whole complex hierarchy is highly modularized and is very flexible to configuration and control.

The insidiousness of the C&C communication is that DirtyMoe does not use fixed IP addresses but implements a unique mechanism to obfuscate the final C&C server address. So, it is impossible to block the C&C server on a victim's machine since the server address is different for each C&C communication. Moreover, the mechanism is based on DNS requests which cannot be easily blocked because it would affect everyday communications.

## 1.8 Actions on Objective

Regarding DirtyMoe modularization, crypto-mining seems to be a continuous activity that uses victim machines permanently if malware authors do not need to make targeted attacks. So, crypto-mining is an activity at a time of peace. However, the NSFOCUS Threat Intelligence center discovered distributed denial-of-service (DDoS) attacks [1]. In general, DirtyMoe can deploy arbitrary malware into infected systems, such as information stealer, ransomware, trojan, etc.

## 2. Network Communication

As we announced, the network communication with the C&C server is sophisticated. The IP address of the C&C server is not hardcoded because it is derived from DNS requests. There are three pieces of information hardcoded in the code.

1. A list of public DNS servers such as Google, Cloudflare, etc.
2. A hard-coded domain `rpc[.]1qw[.]us` is an entry point to get the final IP address of the C&C server.
3. Bulgarian constants[1)] that are used for a translation of IP addresses.

A DNS request to the `1qw[.]us` domain returns a list of live servers, but it is not the final C&C server. DirtyMoe converts the returned IP address to integer and subtracts the first Bulgarian constant (*anything making the result correct*) to calculate the final C&C IP address. A C&C port is also derived from the final IP address using the second Bulgarian's constant to make things even worse.

The insidiousness of the translation is an application of DNS requests that cannot be easily blocked because users need the IP translations. DirtyMoe sends requests to public DNS servers that resolve the `1qw[.]us` domain recursively. Therefore, user machines do not contact the

malicious domain directly but via one of the hardcoded domain name servers. Moreover, the list of returned IP addresses is different for each minute.

We record approx. 2,000 IP addresses to date. Many translated IP addresses lead to live web servers, mostly with Chinese content. It is unclear whether the malware author owns the web servers or the web servers are just compromised by DirtyMoe.

## 3. Statistics

According to our observation, the first significant occurrence of DirtyMoe was observed in 2017. The number of DirtyMoe hits was in the thousands between 2018 – 2020.

## 3.1 DirtyMoe Hits

The increase of incidences has been higher in orders of magnitude this year, as the logarithmic scale of **Figure 3** indicates. On the other hand, it must be noted that we have improved our detection for DirtyMoe in 2020. We record approx. one hundred thousand infected and active machines to date. However, it is evident that DirtyMoe malware is still active and on the rise. Therefore, we will have to watch out and stay sharp.

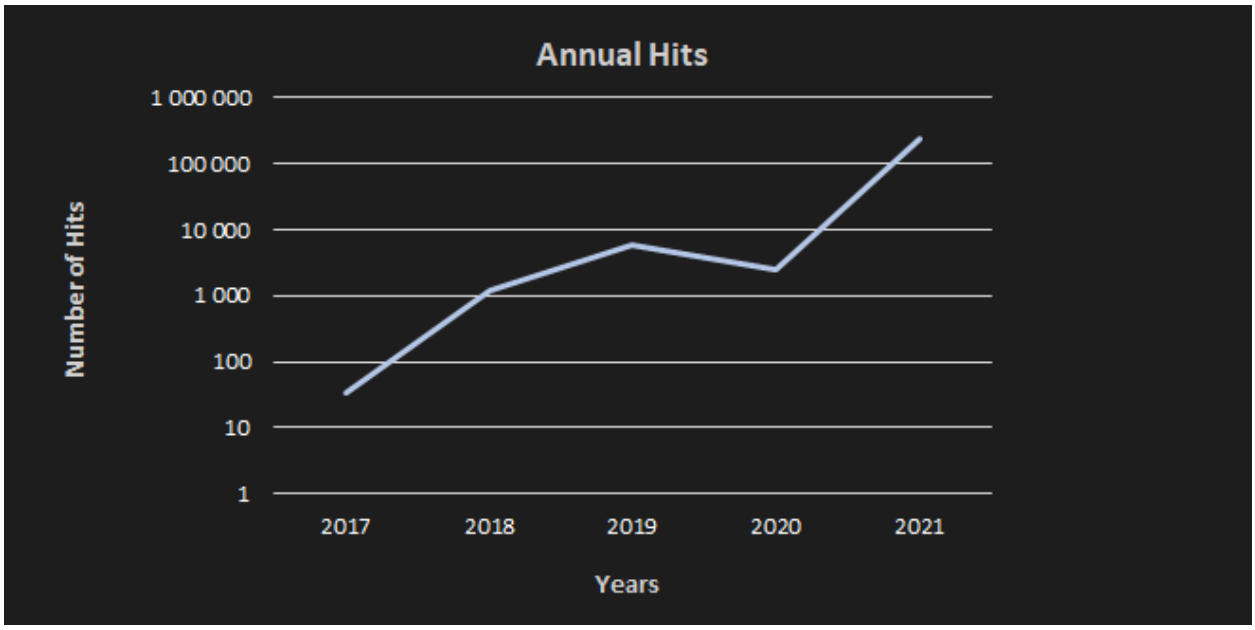


**Figure 3.** Annual occurrences of DirtyMoe hits

The dominant county of DirtyMoe hits is Russia; specifically, 65 k of 245 k hits; see **Figure 4**. DirtyMoe leads in Europe and Asia, as **Figure 5** illustrates. America also has a significant share of hits.

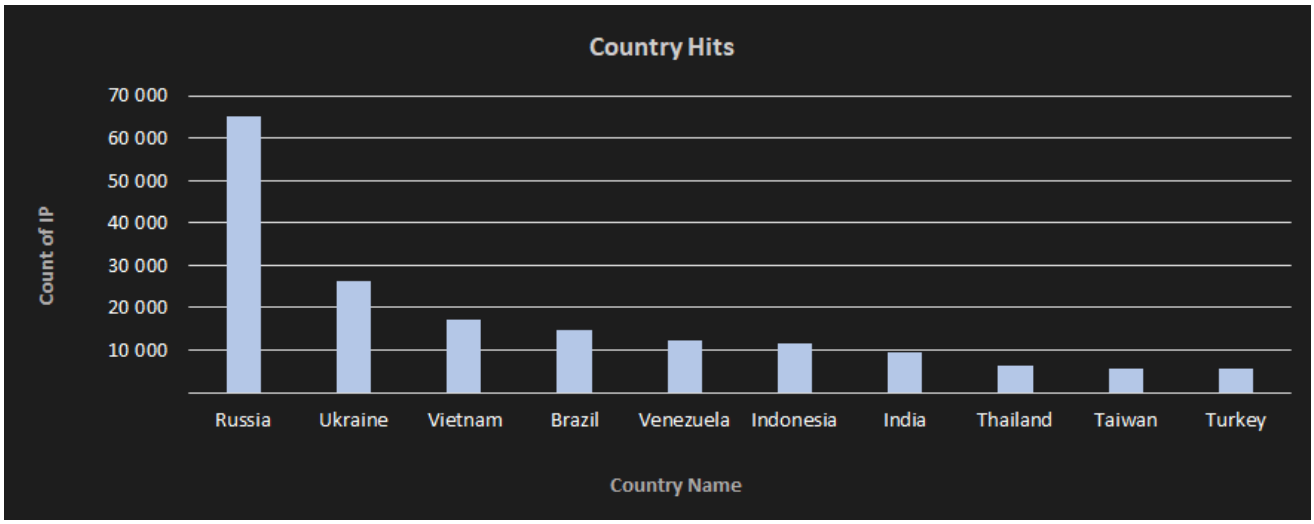


**Figure 4**. Distribution of hits in point of the country view

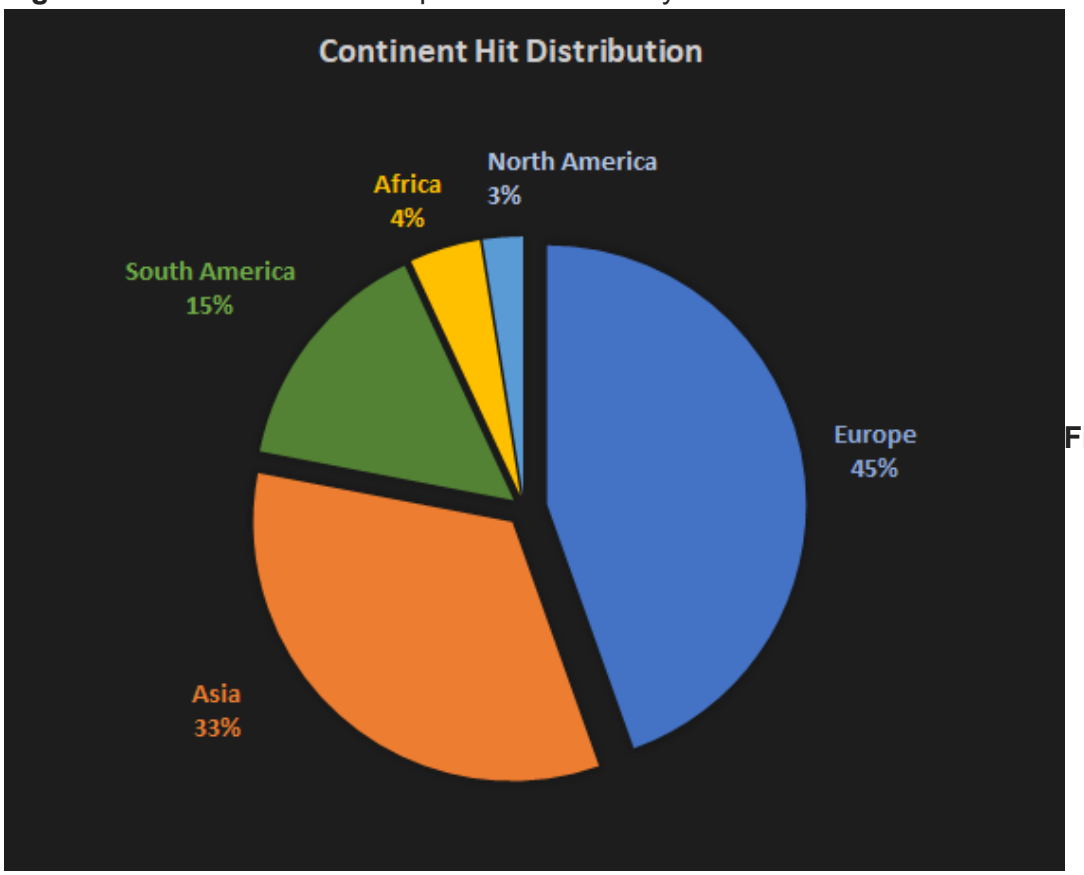


**Figure 5**. Distribution of hits in point of the continent view

## 3.2 C&C Servers

We observed the different distribution of countries and continents for C&C servers (IP addresses) and sample hits. Most of the C&C servers are located in China, as **Figures 6** and **7** demonstrate. We can deduce that the location of the malware source is in China. It is evident that the malware authors are a well-organized group that operates on all major continents.

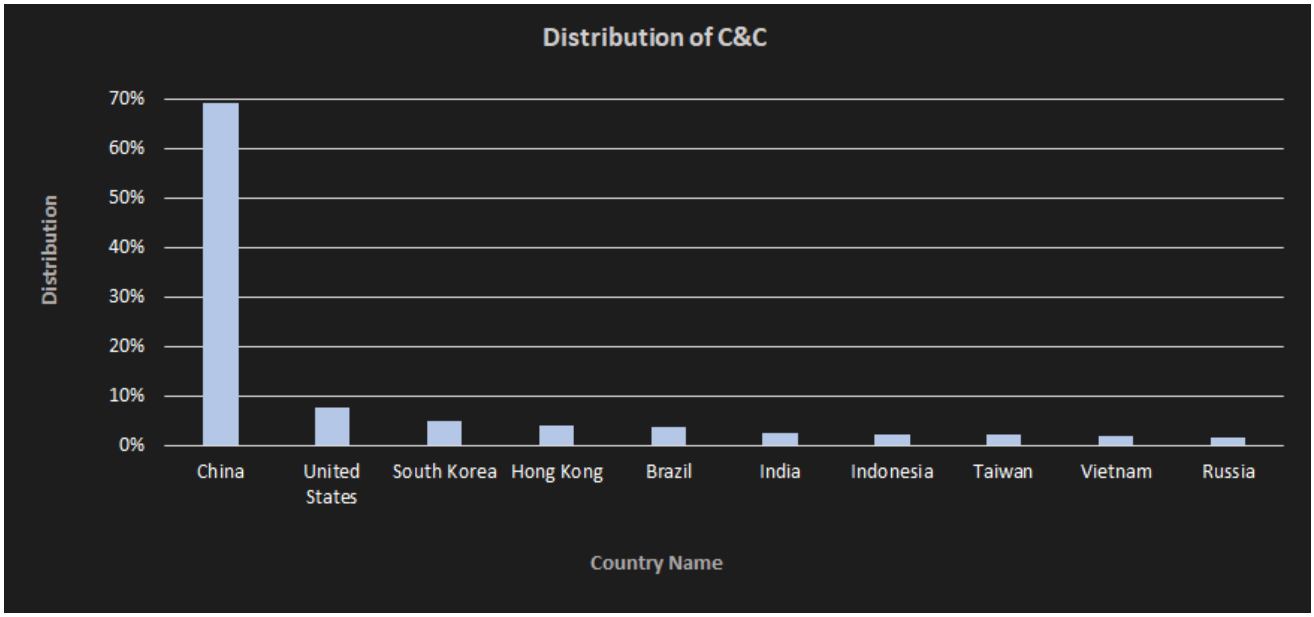


**Figure 6**. Distribution of C&C servers in point of the country view

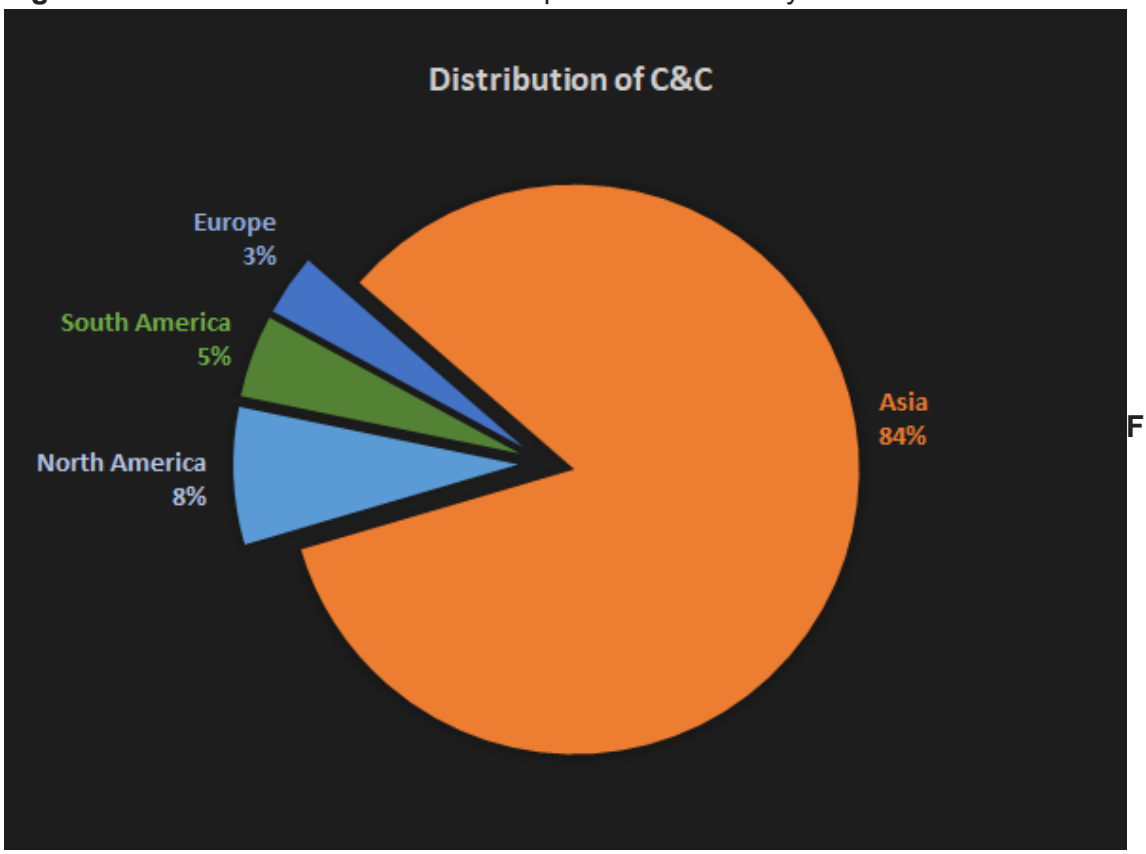


**Figure 7**. Distribution of C&C servers in point of the continent view

## 3.3 VMProtect

In the last two years, DirtyMoe has used a volume ID of the victim's machines as a name for the service DLL. The first versions of DirtyMoe applied a windows service name selected at random. It was dominant in 2018; see **Figure 8**.

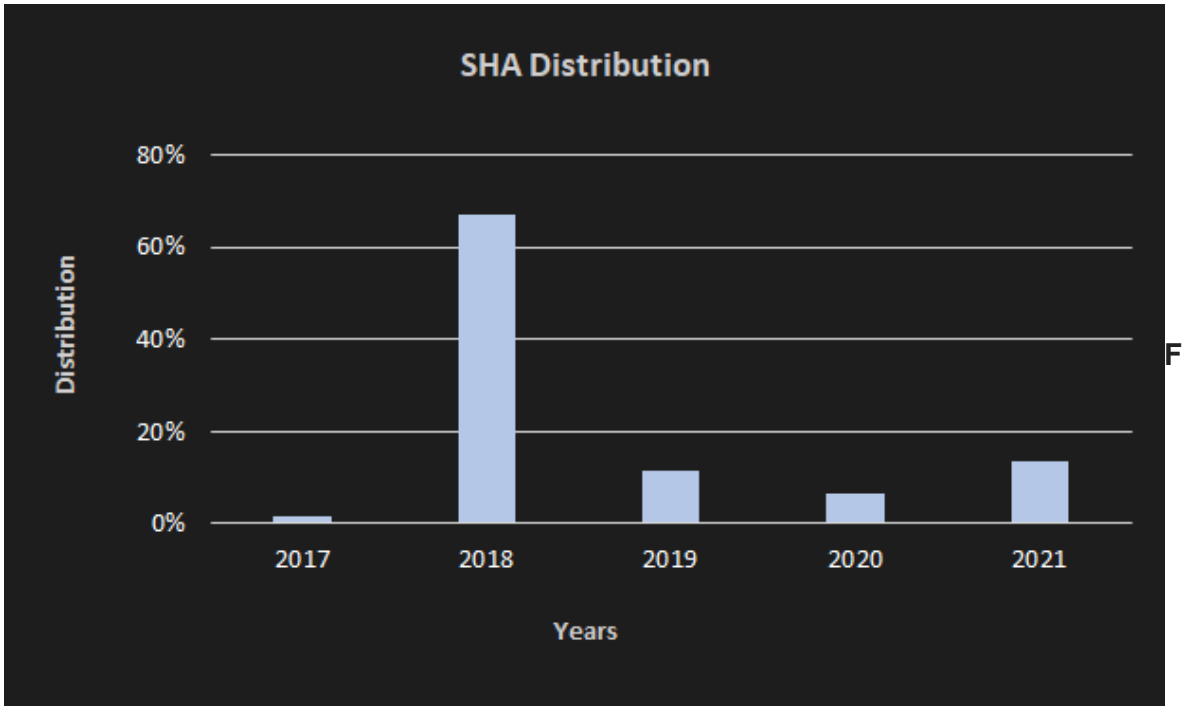


**Figure 8**. Distribution of cases using service names

We detected the first significant occurrence of VMProtect at the end of 2018. The obfuscation via VMprotect has been continuously observed since 2019. Before VMProtect, the malware authors relied on common obfuscation methods that are still present in the VMProtected versions of DirtyMoe.

## 4. Self Protecting Techniques

Every good malware must implement a set of protection, anti-forensics, anti-tracking, and anti-debugging techniques. DirtyMoe uses rootkit methods to hide system activities. Obfuscation is implemented via the VMProtect software and via its own encrypting/decrypting algorithm. Network communications are utilized by a multilevel architecture so that no IP addresses of the mother servers are hardcoded. Last but not least is the protection of DirtyMoe processes to each other.

## 4.1 DirtyMoe Driver

The DirtyMoe driver provides a wide scale of functionality; see the following shortlist:

- Minifilter: affection (hide, insert, modify) of a directory enumeration
- Registry Hiding: can hide defined registry keys
- Service Hiding: patches `service.exe` structures to hide a required service
- Driver Hiding: can hide itself in the system

The DirtyMoe driver is a broad topic that we will discuss in a future post.

## 4.2 Obfuscation and VMProtect

DirtyMoe code contains many malicious patterns that would be caught by most AV. The malware author used VMprotect to obfuscate the DirtyMoe Service DLL file. DirtyMoe Objects that download are encrypted with symmetric ciphers using hardcoded keys. We have seen several MOE objects which contained, in essence, the same PE files differing in PE headers only. Nevertheless, even these slight modifications were enough for the used cipher to produce completely different MOE objects. Therefore, static detections cannot be applied to MOE encrypted objects. Moreover, DirtyMoe does not dump a decrypted MOE object (PE file) into the file system but directly injects the MOE object into the memory.

DirtyMoe stores its configuration in registry entry as follows:
`HKLM\SOFTWARE\Microsoft\DirectPlay8\Direct3D`

The configuration values are also encrypted by the symmetric cipher and hardcoded key. **Figure 9** demonstrates the decryption method for the `DirectPlay8\Direct3D` registry key. A detailed explanation of the method will be published in a future post.

```
int decode(int input_array, int length, int key, int round)
{
  bytearray = input_array;
  if ( length )
  {
    input_array = round - 1;
    if ( round - 1 >= 0 )
    {
      i1 = round - 1;
      do
      {
        *bytearray += BYTE2(key);
        j = length - 1;
        if ( length - 1 > 0 )
        {
          i2 = 1;
          do
          {
            bytearray[i2] = key ^ (bytearray[i2] + bytearray[i2 - 1]);
            ++i2;
            --j;
          }
          while ( j );
        }
        bytearray[length - 1] += HIBYTE(key);
        input_array = length - 2;
        if ( length - 2 >= 0 )
        {
          do
          {
            bytearray[input_array] = BYTE1(key) ^ (bytearray[input_array] + bytearray[input_array + 1]);
            --input_array;
          }
          while ( input_array != -1 );
        }
        --i1;
      }
      while ( i1 != -1 );
    }
  }
  return input_array;
}
```

**Figure 9**. Decryption method for the Direct3D registry value

## 4.3 Anti-Forensic and Protections

As we mentioned in the Exploitation section, the forensic and tracking methods are more complicated since DirtyMoe workers are started over several processes and threads. The workers are initially run as `svchost.exe`, but the DirtyMoe driver changes the worker's process names to `fontdrvhost.exe`. DirtyMoe has two worker processes (*Core* and *Executioner*) that are guarded to each other. So, if the first worker is killed, the second one starts a new instance of the first one and vice versa.

# 5. Symptoms and Deactivation

Although DirtyMoe uses advanced protection and anti-forensics techniques, several symptoms indicate the presence of DirtyMoe.

## 5.1 Registry

The marked indicator of DrityMoe existence is a registry key `HKLM\SOFTWARE\Microsoft\DirectPlay8\Direct3D`.
The DirtyMoe stores the configuration in this registry key.

Additionally, if the system registry contains duplicate keys in the path `HKLM\SYSTEM\CurrentControlSet\Services`, it can point to unauthorized manipulation in the Windows kernel structures. The DirtyMoe driver implements techniques to hide registry keys that can cause inconsistency in the registry.

The MSI installer leaves the following flag if the installation is successful:
`HKEY_LOCAL_MACHINE\SOFTWARE\SoundResearch\UpdaterLastTimeChecked[1-3]`

## 5.2 Files

The working directory of DirtyMoe is `C:\Windows\AppPatch`. The DirtyMoe malware backups the original SENS DLL file to the working directory under the `Acpsens.dll` file.

Rarely, the DirtyMoe worker fails during `.moe` file injection, and the `C:\Windows\AppPatch\Custom` folder may contain unprocessed MOE files.

Sometimes, DirtyMoe "forgot" to delete a temporary DLL file used for the malware deployment. The temporary service is present in Windows services, and the service name pattern is `ms<five-digit_random_number>app`. So, the `System32` folder can contain the DLL file in the form: `C:\Windows\System32\ms<five-digit-random-number>app.dll`

The MSI installer often creates a directory in `C:\Program Files` folder with the name of the MSI product name:
`FONDQXIMSYHLISNDBCFPGGQDFFXNKBARIRJH` or a similar format: `[A-Z]{36}`

## 5.3 Processes

On Windows 10, the DirtyMoe processes have no live parent process since the parent is killed due to anti-tracking. The workers are based on `svchost.exe`. Nonetheless, the processes are visible as `fontdrvhost.exe` under SYSTEM users with the following command-line arguments:

```
svchost.exe -k LocalService
svchost.exe -k NetworkService
```

## 5.4 Deactivation

It is necessary to stop or delete the DirtyMoe service named `Ms<volume-id>App`. Unfortunately, the DirtyMoe driver hides the service even in the safe mode. So, the system should be started via any installation disk to avoid the driver effect and then remove/rename the service DLL file located in `C:\Windows\System32\Ms<volume-id>App.dll`.

Then, the DirtyMoe service is not started, and the following action can be taken:

- Remove `Ms<volume-id>App` service
- Delete the service DLL file
- Remove the core payload and their backup files from `C:\Windows\AppPatch` `Ke<five-digit_random_number>.xsl` and `Ac<volume-id>.sdb`
- Remove DirtyMoe Objects from `C:\Windows\AppPatch\Custom` `<md5>.mos` ; `<md5>.moe` ; `<md5>.mow`

## 6. Discussion

Parts of the DirtyMoe malware have been seen throughout the last four years. Most notably, NSFOCUS Technologies described these parts as Nugget Phantom [1]. However, DirtyMoe needs some exploit kit for its deployment. We have clues that PurpleFox [5] and DirtyMoe are related together. The question is still whether PurpleFox and DirtyMoe are different malware groups or whether PurpleFox is being paid to distribute DirtyMoe. Otherwise, both malware families come from the same factory. Both have complex and sophisticated network infrastructures. Hence, malware could be a work of the same group.

The DirtyMoe authors are probably from China. The DirtyMoe service and MOE Modules are written in Delphi. On the other hand, the DirtyMoe driver is written in Microsoft Visual C++. However, the driver is composed of rootkit techniques that are freely accessible on the internet. The authors are not familiar with rootkit writing since the drive code contains many bugs that have been copied from the internet. Moreover, Delphi malicious patterns would be easily detectable, so the authors have been using VMProtect for the code obfuscation.

The incidence of PurpleFox and DirtyMoe has increased in the last year by order of magnitude, and expected occurrence indicates that DirtyMoe is and will be very active. Furthermore, the malware authors seem to be a large and organized group. Therefore, DirtyMoe should be closely monitored.

## 7. Conclusion

Cryptojacking and DDoS attacks are still popular methods, and our findings indicate that DirtyMoe is still active and on the rise. DirtyMoe is a complex malware that has been designed as a modular system. PurpleFox is one of the most used exploit kits for DirtyMoe deployment. DirtyMoe can execute arbitrary functionality based on a loaded MOE object such as DDoS, etc., meaning that it can basically do anything. In the meantime, the malware aims at crypto-mining.

The malware implements many self-defense and hiding techniques applied on local, network, and kernel layers. Communication with C&C servers is based on DNS requests and it uses a special mechanism translating DNS results to a real IP address. Therefore, blocking of C&C servers is not an easy task since C&C addresses are different each time and they are not hard-coded.

Both PurpleFox and DirtyMoe are still active malware and gaining strength. Attacks performed by PurpleFox have significantly increased in 2020. Similarly, the number of DirtyMoe hits also increases since both malware are clearly linked together. We have registered approx. 100k infected computers to date.  In each case, DirtyMoe and Purple Fox are still active in the wild. Furthermore, one MOE object aims to worm into other machines, so PurpleFox and DirtyMoe will certainly increase their activity. Therefore, both should be closely monitored for the time being.

This article summarizes the DirtyMoe malware in point of high-level overview. We described deploying, installation, purpose, communication, self-protecting, and statistics of DirtyMoe in general. The following articles of the DirtyMoe series will aim to give a detailed description of the MSI installer package, DirtyMoe driver, services, MOE objects, and other interesting topics related to the whole DirtyMoe architecture.

**References**


[1] Nugget Phantom Analysis
[2] Purple Fox Rootkit Now Propagates as a Worm
[3] EternalBlue Still Relevant
[4] Scripting Engine Memory Corruption Vulnerability
[5] Purple Fox malware


Tagged asbackdoor, cryptomining, DirtyMoe, modularized, series