uc3m | Universidad **Carlos III** de Madrid

Master in Cybersecurity
2017-2018

Master Thesis

"MalwareWorld: An Intelligence-driven Malware Detection System based on Open Sources"

Carlos Polop Martin

Supervisor:
Juan Manuel Estevez Tapiador

July 2018

MalwareWorld: An Intelligence-Driven Malware Detection System based on Open Sources

Carlos Polop Martín
100376385@alumnos.uc3m.es

Juan Tapiador (*Advisor*)
jestevez@inf.uc3m.es

Abstract—**This work presents MalwareWorld, a system aimed at detecting if a host (domain or IP) is related with suspicious activities. MalwareWorld leverages hundreds of publicly available blacklists to collect information about hosts that are probably involved in some kind of malicious activity. The system integrates and merges all collected data into a common database, thus offering a multi-list detection capability similar to what VirusTotal does with multi-engine anti-virus tools. In addition, some external intelligence source (i.e., tools similar MalwareWorld) are also queried to figure out if a host is related to some kind of suspicious activities. MalwareWorld offers three interfaces for different uses cases: a nodeJS library, a web API, and a Telegram bot. Finally, the project has been merged with two others (Maltrail and dockvpn) to provide an integrated system offering malicious network activity detection, the privacy of a VPN, and the easy deployment of a docker container.**

Keywords—blacklist, malware, spammer, phising, tor, dga, attacker, cryptocurrency, proxy, node, malwareworld, maltrail

I. INTRODUCTION

Nowadays, most of the people already know that Internet harbour a lot of danges. However, what most people do not know is how to protect themselves from them. Internet is a tool that is being used by the population more and more, and this increase is exponential. It is used to learn, to work, to communicate with family and friends, to spend free time, etc. It is needed almost in every aspect of the life of everyone. Despite the fact that most people are concerned about the dangers online, most people think that just installing a free antivirus will protect them from cyber criminals that somehow will try to take advantage of them. Besides, it is known by all the cybersecurity experts that it is impossible to protect a device against all kind of attacks. These are bad news for all the people and companies who wants to protect themselves against all the malicious attacks.

However, it is not all lost. It is true that it is impossible to protect a device against all kinds of malicious intentions but there is a lot that a person or a company can do in order to difficult the attack so much that maybe the attacker will stop trying to compromise them. There are a lot of tools that can be used to protect a single device or an entire organization. Depending on the number of devices and the importance of the information that they are going to work with and the exposure of this device to internet, more protections could be implemented to avoid the attackers to compromise and use the device.

In this paper we present a new and free tool that could be used to improve the security of any device connected to the internet. It could be used with any device and in any environment, and it is specially focused for all the highly unprotected devices whose security is difficult to improve because it is not possible to install anything inside them, are connected in some way to Internet and its number is growing exponentially: the Internet of Things. Besides, this project is also very helpful to protect and improve the privacy of a device that everybody uses, is used to process all kind of important data, and usually it is highly unprotected and connected to all kind of uncontrolled environments: the smartphone.

It is known that most malware created these days needs to communicate with some server controlled by the attackers where the malware will send information of the victim and will receive instructions. Then, the malware will need to send packets through internet to communicate with the criminals. From this idea, is where this project called MalwareWorld borns. MalwareWorld is a system based in a lot of publicly available information that collects information about hundreds of thousands of hosts that are related with suspicious activities. It collects information about hosts related with Malware, Spam, Phishing, CryptoCurrencies, Attackers, etc. from hundreds of pubic blacklists. All this information is organized so that it is very easy to query if a host is related with any kind of suspicious activity. In summary, MalwareWorld is capable to identify if a host is probably taking part of a malicious activity.

However, in order to make it easier to use, a web page and a telegram bot was created where you can query if a host is suspicious. The web page is:

https://malwareworld.com

and the telegram bot is called **@MalwareWorldBot**.

As it was said above, this project can be useful to detect any kind of suspicious packet sent by any device because you can search if any of the hosts which a device is communicating to is in the database of MalwareWorld. However, the project was focused in the weakest devices and in the devices that are connected to a lot of uncontrolled environments (Internet of Things and smartphones). Then, MalwareWord was merged with a project called Maltrail. Together they create MaltrailWorld, which is a tool that allows to detect malicious traffic in real time using the database created by MaltrailWorld. Then, if all the traffic of any IoT device or any smartphone go through MaltrailWorld, could be detected malicious activities in real time.

Besides, it is known that smartphones are usually connected to a lot of uncontrolled environments like WiFis of hotels or restaurants. The best countermeasure to the dangers of doing this is to use a VPN. Then, the project MaltralWorld was merged with a project that creates a VPN server using docker (dockvpn). The resulting project was called: MaltrailWorld-VPN. MaltrailWorld-VPN allows anyone to start a docker container that will create a VPN server where he can connect any device that supports VPN (for example, his smartphone) and the container will also execute MaltrailWorld. So, the user will have the privacy provided by the VPN and will know if the phone is sending some kind of suspicious packet.

The rest of the paper is organized as follows. In section 2 it is described related work in the area of malware blacklisting. In Section 3, is described how does MalwareWorld works and which services does it offers. In Section 4, we discuss how this project can be used and some results are shown. Finally, Section 5 concludes the paper and discusses limitations and future work.

II. Related Work

Detecting malicious activities in the network using blacklists has been used since long time ago. Indeed, the use of blacklists was one of the first methods used to protect a network against attacks. However, what this project provides is a simple method to collect and merge data of hundreds of public blacklists and it makes very easy to collect data from new blacklists. Besides, what is really new is the capability of creating a VPN and looking for bad behaviours as simple as using a docker container.

There are other projects dedicated to look for hosts related with the dangers of the internet, such as: FraudGuard[1], FortiGuard[2], ProjectHoneyPot[3], SpamCop[4], or Web Of Trust[5], to name a few. These projects use different methods to figure out if a host is involved in some kind of potentially undesirable activity. They are currently using several types of different honey pots, these are machines that looks vulnerable in purpose in order to attract attacks and then investigate who is attacking and how is doing it. Other ways to detect malicious activities is by searching in the internet web pages that are being using for Phishing attacks, or to collect the data provided by thousands of users about web pages to know if a web is clean or not.

With this in mind, MalwareWorld was designed to improve the service offered by existing services. As all the commented projects and MalwareWorld have the same purpose (detecting hosts related with malicious activities in the internet), it was thought that using all of them in the same platform would be appreciated. That is why APIs for all the commented projects were created and they were implemented in MalwareWorld. Then, MalwareWorld is able to detect suspicious hosts not only based in hundreds of blacklists but also in other projects that are also looking to protect the internet.

There are other publicly available systems that provide malware intelligence services. One of the best known is VirusTotal[6], an online service where you can upload an executable and it will tell you information about if it contains something malicious. It also can provide existing information about submitted domains, including if it is thought to be malicious.
Finally, one last platform that deserves to be mentioned in this section is MISP[7]. This platform allows to share all kind of information about attack tactics, techniques and procedures between researches, companies and the public sector. This allows to share the detection of cyber attacks in a very fast way between the sectors that more need it.

III. Approach

This section describes all the components of the MalwareWorld Project. It is first discussed in detail the basis of the project, namely how it has been implemented to be scalable and how the engine collects the data of several blacklists and correlate them in a fast and efficient way. Then, it is described the integration with more external intelligence sources and how they were implemented, as most of them did not have any NodeJS API. Section 3.3 describes the MalwareWorld web platform and the resources it offers. Subsequently, it is described the integration with Maltrail and dockvpn.

A. Architecture

This subsection explains the bases of the project. How was implemented the engine with speed and efficiency in mind and how were implemented the blacklists with scalability in mind. Figure 1 shows the main components of the architecture and how are they correlated.

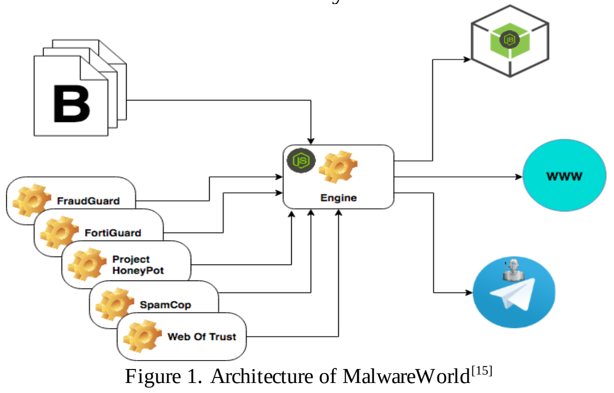


Figure 1. Architecture of MalwareWorld[15]

1) Blacklists

As it has been said in the introduction, the main objective of the project is to collect and merge several public blacklists with the purpose of identifying if an IP or domain is related with some kind of suspicious activity. As the project is based in the use of as much blacklists as possible, the scalability of the number of blacklists used is very important. Then, it has been designed in a way that if anyone wants to add a new blacklist, he only needs to configure it in a file called “blacklists_list.js”. Inside this file, in order to set a new blacklist, the user has to create a new JSON object with the following fields:

```
{
      reference: “”,
      title: “”,
      type: “”,
      mode: “”,
      submode: “”,
	comment: “”,
	is_valid: “”
}
```

The reference is the URL were the blacklist is located.
The title is the title that is going to be given to every host included in the blacklist. It can be a predefined title, a custom title or a regular expression that will search for some information inside each line of the blacklist. The predefined titles are included in the “title” variable inside the same file:

```
var title = {default: "Malicious Host", bitcoin: "Bitcoin Node", spammer: "Spammer", phishing: "Phishing", proxy: "Proxy", tor: "Tor", ransomware: "Ransomware", zeus: "Zeus", web_proxy: "Web Proxy", browser_hijaking: "Browser Hijacking", cryptocurrency: "Cryptocurrency", cheap_domain: "Cheap Domain", get_my_ip: "Get my public ip", web_tor: "Web to Tor", superfish: "Superfish", shellcode: "Shellcode", adware: "Adware"}
```

The type of each blacklist reveals the type of every host that is going to be find inside the blacklist and has to be one the 10 predefined types inside the variable “type”:

```
var type = {default: "BadReputation", malware: "Malware", bad_reputation: "BadReputation", known_attacker: "KnownAttacker", spammer: "Spammer", phishing: "Phishing", cryptocurrencies: "CryptoCurrencies", hide_source: "HideSource", adware: "Adware", dga: "DGA"}
```

The types are self-explanatory, but it could be interesting to notice that the type “BadReputation” is used by default for some blacklists when it is not clear which method is used to collect the hosts.

The mode indicates what is going to be found inside the blacklist and which regular expression should be used to detect it. There are already defined regular expressions for URLs, URLs without the protocol, domains, ips and ip ranges inside the variable “mode”. Currently, every blacklist configured inside the project uses one of these regular expressions.

The submode indicates how the information needs to be extracted. Currently there are 4 modes and are indicated inside the “submode” variable. The submodes are: Extract the information from every line of the list, extract all the matches, and 2 custom ways for extracting the information needed for some blacklists.

The comment field should only be completed if blacklists have comments and should be completed only with one character. The one that determine that a line is a comment.

The is_valid field is currently not in use.

The MalwareWorld project currently uses more than 500 blacklists, and each one is configured in a JSON object with the fields explained above. This is a major feature of the project, namely the ability to merge a very large number of blacklists into one and the easiness with which more blacklists can be added.

The code of this part of the project can be found at:

https://github.com/carlospolop/MalwareWorld/blob/master/blacklists_list.js

2) Engine

The main purpose of the engine is to gather the information of each blacklist configured and download and parse it correctly. In order to do it, the engine will visit every URL and download the content of each blacklist. Then, depending on the parameters configured in “mode”, “submode” and “comment” of the JSON object where the blacklist was configured, it will parse the list in order to take all the necessary information (at least, the hosts inside it).
Every host found in every blacklist will be saved in memory as a JSON object with the following parameters:

```
“IP/DOMAIN/RANGE”: {
	title: “”,
	type: [],
	urls: [],
	location: {lat: “”, lng: “”},
	references: []
}
```

Every host will have at least one title that is taken from the “title” field of the blacklist were the host was found. A host will save something in the “urls” field only if the blacklist were the host was found had any url (for example, in some blacklists exists an url pointing where a malicious file was found).

Only the JSON objects referred to an IP will have something in the “location” field. This is because in order to find the location of a host the library “geoip-lite” is used, and this library only accepts as input an IP. At some point of time it was implemented a DNS query for each suspicious domain found, but as most of the domains of the blacklists does not exist anymore and using the currently implemented blacklists more than 100.000 suspicious domains are found, making a DNS query for each domain is useless.

Another problem that needed to be solved was what happens if two hosts appears in two different blacklists. This situation is very common as usually this project finds more than 300.000 suspicious host, 200.000 of them unique and 100.000 are repetitions. So, it was implemented that if the same host was found in several blacklists, all the information is merged into the same JSON object. Then, if the titles are different, the final title of the JSON object will merge both titles with a “/” in the middle. If the types are different, both types will be saved in the “type” array. All the urls found in both blacklists related to the same hosts will be recorded in the "url" array if they are different. And finally, all the URLs of the blacklists will be saved in the "references" array. The "location" field was not mentioned as the only JSON objects that has this field completed are the IPs and independently of the blacklists were the IP was found, the location is going to be the same.

The language used to develop the engine of the project is NodeJS. This is because using this language is the easiest way to implement a highly asynchronous program. Is has already been told that the project currently implements more than 500 blacklists and parsing one by one would be very inefficient. Then, it was decided to use the NodeJS library "request" which lets the engine to visit every url and parse it in an asynchronous mode, so, when executed, the engine is working as much as it can and no delay due to waiting for a blacklist to be download exists. Despite these advantages, due to the number of blacklists that needed to be downloaded and parsed, creating the big database in memory of JSON objects takes 6 minutes to a MacBook Pro of late 2011 with 16GB of memory, and 10 minutes to a server with 2 x86_64bit Cores and 2GB of memory.

Usually more than 200,000 unique suspicious hosts are found inside all the blacklists and knowing all the parameters that a JSON object related to a host need to be saved, the final database of JSON objects can occupy an important size in memory. Besides, the NodeJS library "geopip-lite" is used to find the location of the IPs, and this library also makes a heavy use of the memory. Then it has been discovered that in orden to run the project is needed at least a device with 2GB of memory. However, it was decided to store all the JSONs of the suspicious hosts in memory instead of in a database because then the search of a suspicious host is faster.

It has already been explained how the engine works and why it has been programmed as it is. Now it is going to be explained how a user or an external program can interact with it. The engine has been programmed as a NodeJS library, so a NodeJS program could import methods and interact with the engine. The most important method that the library exports is `isMalicious(String)`. This method can receive an url, a domain or an IP and the response will a be a promise that will return all the data that the database has about that host if it has something. The function automatically detects if the argument is an URL, a domain or an IP.

There are also other useful methods exported:

- `renewAllData()`: This function deletes the database and creates it again.
- `renewNotRespondingData()`: When the database is created, if the engine could not gather the information of a blacklist, it is recorded so when this method is called the engine will try again to gather the information of all the blacklists that didn't work the last time.
- `renewAllDataInterval(Int)`: This function allows the programmer to set a time in minutes so when this time passed, the database will be reconstructed. It is not recommended to set the interval to a value less than 12 hours.
- `renewNotRespondingDataInterval(Int)`: This function allows the programmer to set a time in minutes so when this time passed, the function renewNotRespondingData will be called.
- `getMalIpsList()`: This function returns all the information gathered about all the suspicious IPs found.
- `getMalDomainsList()`: This function returns all the information gathered about all the suspicious domains found.
- `getMalRangesList()`: This function returns all the information gathered about all the suspicious ranges of IPs found.
- `getNotRespondingList()`: This function return all the blacklists that did not answer when the engine try to download them.
- `getMalHostsOf(String)`: This function return all the suspicious hosts of a type (the valid types are the ones commented in the section 3.1.1).
- `setDebug()`: This function is very useful to debug the code as a lot of comments are generated while the code is running.
- `quitSha1()`: This function is useful if the programmer does not want the sha1 of every blacklist to be calculated.
- `getStatistics()`: This function returns how many hosts were found in each blacklist an the sha1 of the blacklist.
- `getGeneralStatistics()`: This function returns how many suspicious IPs, domains and ranges has been found, the number of blacklists used, the number of blacklists that did not work, the number of unique suspicious hosts found, the number of repeated hosts found and information about the activated external intelligences.
- `listenInPort(Int)`: This function is used to set the function isMalicious to listen in a port.

The code of the engine can be found at:

https://github.com/carlospolop/MalwareWorld/blob/master/malwareworld.js

The next section describes a number of improvements added to the already detailed engine.

B. *External Integillence Sources*

The base of the system is the power of merging a huge number of blacklists. However, ignoring that there are other projects similar to this one would be a big mistake. To address this issue, we integrated 5 projects into ours. These projects don't have publicly available the blacklist that they are using, but you can ask them if an IP and/or a domain is suspicious. This is the main reason why the blacklists of these projects are not integrated in the blacklist that

MalwareWorld creates. Instead, these projects are integrated as external intelligence sources that will be queried for a host when the function `isMalicious` is used.

This improvement implies that when a user wants to know if a host is related with suspicious activities, he could use this project to check in several blacklists and in several intelligences at the same time.

Because of this development, a number of new functions are exported:

- `deactivateBlacklists()`: This function disables the creation of the blacklists merging all the blacklists configured. This is useful if you only want to use the external intelligences.
- `setDebugv2()`: This function enable the print of logs useful for checking if the external intelligences are working correctly.
- There are more exported functions for each external intelligence integrated, the purpose of these functions is to enable/disable the external intelligence.

The five blacklisting services integrated into MalwareWorld are described next.

1) FraudGuard

FraudGuard[1] is a service dedicated to find host related with suspicious activities by analyzing internet traffic. It has a friendly API and a free plan that allows 1000 requests per month. As there was not any implementation of the API in NodeJS we made our own, which can be found at:

https://github.com/carlospolop-node-apis/node-fraudguard

In order to activate this source, you have to set your Fraudguard's user and password using the functions `setFraudguardUser(String)` and `setFraudguardPassword(String)`.

2) FortiGuard

FortiGuard[2] is a service dedicated to find all kind of dangers all over internet: from vulnerabilities to malware and bad reputation hosts. This intelligence service has a free service that you can use to check if a IP is relate with suspicious activities, so I create my own NodeJS API to use it. It can be found at:

https://github.com/carlospolop-node-apis/node-fortiguard

As no credentials are required to use this service, MalwareWorld will use it by default. If the user wants to disable it, it can be done by using the function `deactivateFortiguard()`.

3) Project HoneyPot

HoneyPot[3] is another service dedicated to find host performing illegitimate actions. It is focused in finding spammers and it has a free API. A NodeJS API was also created in order to integrate it with MalwareWorld. The implementation can be found here:

https://github.com/carlospolop-node-apis/node-projecthoneypot

In order to use this intelligence source the user has to set the Token associated with his account. This can be done using the function `setProjecthoneypotKey(String)`.

4) SpamCop

As the Project Honey Pot, SpamCop[4] is focused on finding hosts used to send spam. Like Fortiguard, this intelligence source has a free API that can be used to check if a host is related with suspicious activities. Then, a script implementing the use of this API in NodeJS was created, which be found at:

https://github.com/carlospolop-node-apis/node-simplespmacop

By default, MalwareWorld will use this source as it does not required any credentials. It can be disabled by using the function `deactivateSpamcop()`.

5) WOT(Web Of Trust)

Web Of Trust[5] is a intelligence powered by millions of users and artificial intelligence. Its main goal is determining if a domain is legitimate or exhibit suspicious behaviours based on the opinion of users. WOT offers a free API once the user registers and obtains an account for free. An implementation of the API was made for NodeJS and the code can be found at:

https://github.com/carlospolop-node-apis/node-weboftrust

In order to activate the use WOT, the user has to set his API token using the function `setWOTKey(String)`.

6) DGA-Detective

DGA is the acronym of Domain Generation Algorithm. There are a lot of malware that generates hundreds of thousands of possible domains (using DGA), where the Command and Control could be located, but at the end only one domain will be the C&C. There are some blacklists that are updated with all these possible domains, but they are so huge that MalwareWorld gets frozen if it tries to parse them all. To address this issue, we decided to create a very simple and useful tool that, based on how the name of the domain is constructed, it determines if the domain is a real one or it has been created using DGA. Our approach does this mainly by looking the length of the name, the letters and numbers that compose the name, how are they distributed, its entropy, and checking the Ecosia search engine to know if the name of the domain gives some kind of result.

This have been implemented as a library in NodeJS and its code and more information about how it works can be found at:

https://github.com/carlospolop-node-apis/dgadetective

C. MalwareWorld Web Service

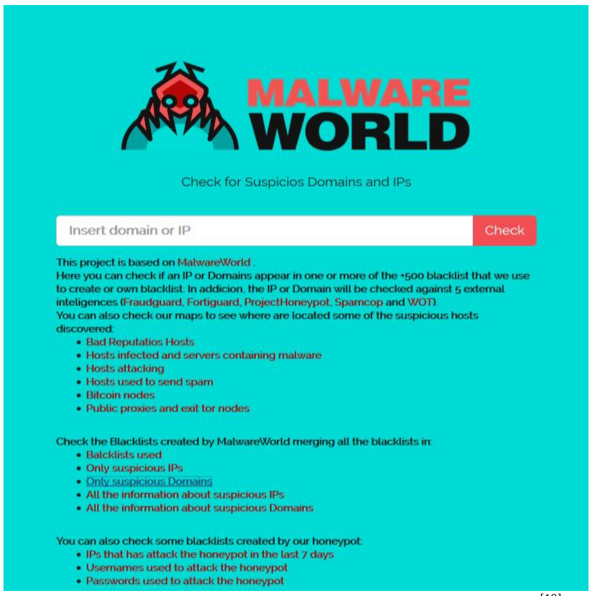
This section describes MalwareWorld's web interface, which is available at:

https://malwareworld.com

We also describe a Telegram bot that implements the web API to check if an IP or domain is suspicious.

1) *Main Page*

The main page (See Figure 2) consists on two parts. The first one is an input field that allows the user to introduce an IP or a domain and will provide him with all the information that MalwareWorld and the 5 external intelligence sources have about it, if any. The second part is a summary about other paths of this page that could be of interest to the user. We next describe this in detail.



Check for Suspicios Domains and IPs

Insert domain or IP | Check

This project is based on MalwareWorld .
Here you can check if an IP or Domains appear in one or more of the +500 blacklist that we use to create or own blacklist. In addicion, the IP or Domain will be checked against 5 external inteligences (Fraudguard, Fortiguard, ProjectHoneypot, Spamcop and WOT).
You can also check our maps to see where are located some of the suspicious hosts discovered:

- Bad Reputatios Hosts
- Hosts infected and servers containing malware
- Hosts attacking
- Hosts used to send spam
- Bitcoin nodes
- Public proxies and exit tor nodes

Check the Blacklists created by MalwareWorld merging all the blacklists in:

- Balcklists used
- Only suspicious IPs
- Only suspicious Domains
- All the information about suspicious IPs
- All the information about suspicious Domains

You can also check some blacklists created by our honeypot:

- IPs that has attack the honeypot in the last 7 days
- Usernames used to attack the honeypot
- Passwords used to attack the honeypot

Figure 2. Screenshot of the web https://malwareworld.com.[10]

2) *Maps*

The web of the MalwareWorld project takes advantage of the location of the IPs of the final database and creates maps were all the suspicious IPs are located. Due to the big number of suspicious IPs, if the user tries to load all the IPs with their descriptions in only one map, the Browser will probably get frozen. This is why all the suspicious host are distributed along the 6 maps shown in Table I.

In each of the maps appears an input field that can be used to check if an IP or domain is suspicious (see Figure 3):

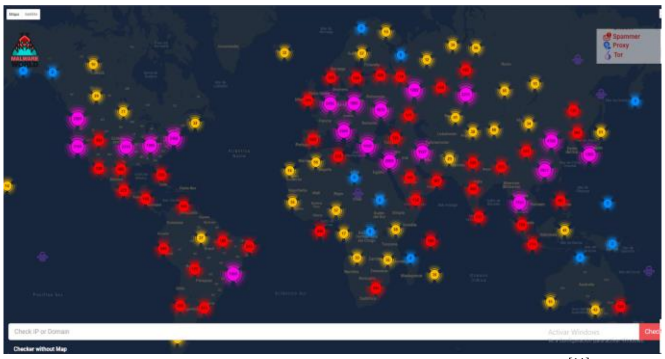


Figure 3. Screenshot of the map showing known attackers.[11]

Map	Description	URL
Bad Reputation	IPs that have bad reputation	https://malwareworld.com/badreputation
Malware	IPs related with malware	https://malwareworld.com/malware
Known Attackers	IPs detected to be attacking	https://malwareworld.com/attackers
Spammers	IPs sending spam	https://malwareworld.com/spammers
Cryptocurrencies	IPs related with cryptocurrencies	https://malwareworld.com/cryptocurrencies
Source being hidden	IPs of tor exit nodes, web-to-tor hosts and public proxies	https://malwareworld.com/hidesource

Table I. Maps available in the MalwareWorld web service.

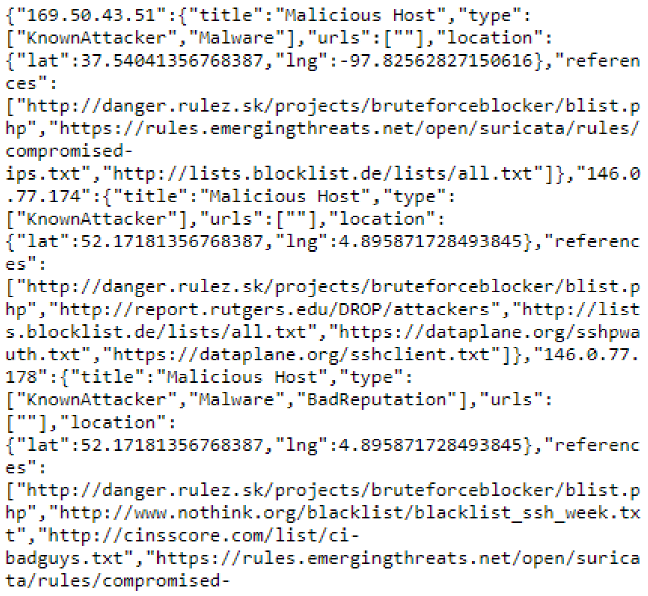
3) *Lists*

The main page also has also all the important lists that this project creates (see Table II).

Name	Description	URL
Blacklists	Blacklists used by MalwareWorld to create the final blacklist	https://malwareworld.com/textlists/blacklists.txt
IPs (only)	All the suspicious IPs (only IPs)	https://malwareworld.com/textlists/suspiciousIPs.txt
Domains (only)	All the suspicious Domains (only domains)	https://malwareworld.com/textlists/suspiciousDomains.txt
IPs	All the information of every suspicious IP	https://malwareworld.com/textlists/ips.txt
Domains	All the information of every suspicious	https://malwareworld.com/textlists/domains.txt

Table II. Lists available in the MalwareWorld web service.

Figure 4 provides an extract of the IPs located inside a list and all the associated fields.



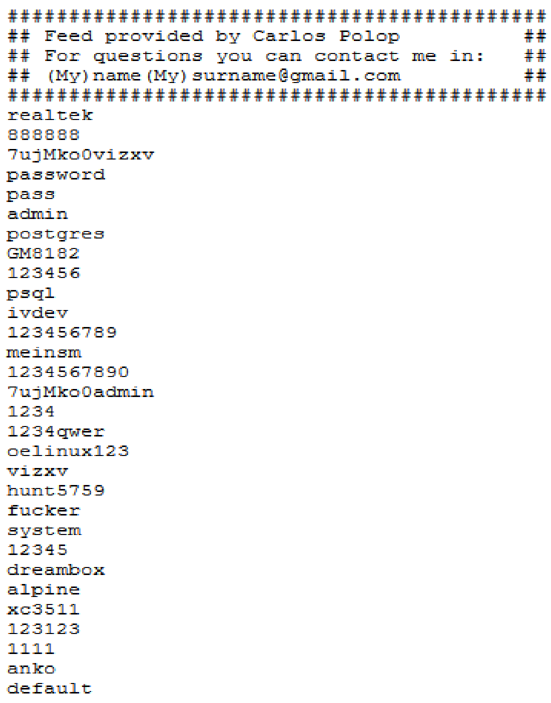
```
{"169.50.43.51":{"title":"Malicious Host","type":
["KnownAttacker","Malware"],"urls":[""],"location":
{"lat":37.54041356768387,"lng":-97.82562827150616},"referen
ces":
["http://danger.rulez.sk/projects/bruteforceblocker/blist.p
hp","https://rules.emergingthreats.net/open/suricata/rules/
compromised-
ips.txt","http://lists.blocklist.de/lists/all.txt"]},"146.0
.77.174":{"title":"Malicious Host","type":
["KnownAttacker"],"urls":[""],"location":
{"lat":52.17181356768387,"lng":4.895871728493845},"referenc
es":
["http://danger.rulez.sk/projects/bruteforceblocker/blist.p
hp","http://report.rutgers.edu/DROP/attackers","http://list
s.blocklist.de/lists/all.txt","https://dataplane.org/sshpwa
uth.txt","https://dataplane.org/sshclient.txt"]},"146.0.77.
178":{"title":"Malicious Host","type":
["KnownAttacker","Malware","BadReputation"],"urls":
[""],"location":
{"lat":52.17181356768387,"lng":4.895871728493845},"referenc
es":
["http://danger.rulez.sk/projects/bruteforceblocker/blist.p
hp","http://www.nothink.org/blacklist/blacklist_ssh_week.tx
t","http://cinsscore.com/list/ci-
badguys.txt","https://rules.emergingthreats.net/open/surica
ta/rules/compromised-
```

Figure 4. Screenshot of part of https://malwareworld.com/textlists/ips.txt[12]

It has also been configured a publicly available HoneyPot in order to find suspicious hosts. This honey pot is constantly receiving attacks and creating blacklists of IPs that attack it and of which usernames and passwords have the attackers used to try to take control of the machine. These blacklists can be accessed here:

- Blacklist of IPs: https://lists.malwareworld.com/blacklist
- Usernames used to attack the HoneyPot: https://lists.malwareworld.com/users
- Passwords used to attack the HoneyPot: https://lists.malwareworld.com/passwords

Figure 5 shows an extract of the list containing captured passwords.

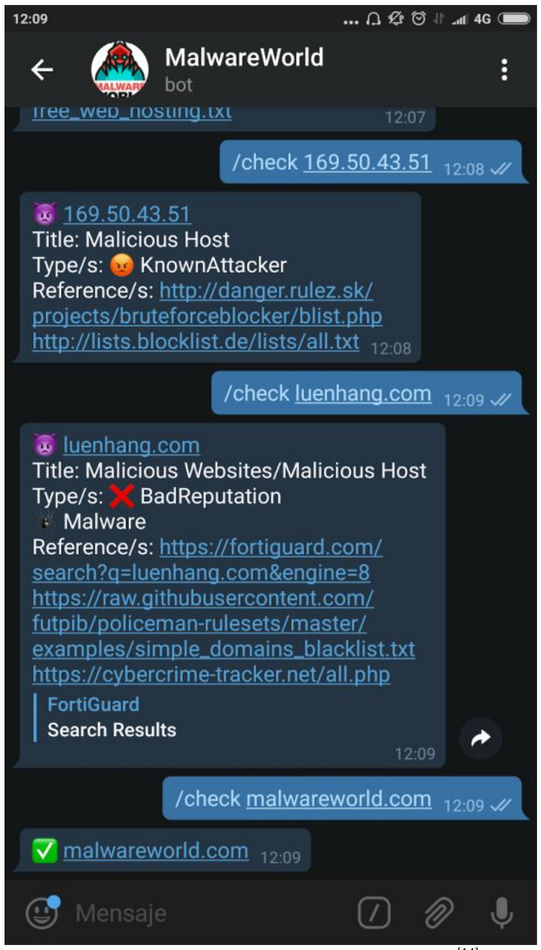


```
##############################################
## Feed provided by Carlos Polop            ##
## For questions you can contact me in:     ##
## (My) name (My) surname@gmail.com         ##
##############################################
realtek
888888
7ujMko0vizxv
password
pass
admin
postgres
GM8182
123456
psql
ivdev
123456789
meinsm
1234567890
7ujMko0admin
1234
1234qwer
oelinux123
vizxv
hunt5759
fucker
system
12345
dreambox
alpine
xc3511
123123
1111
anko
default
```

Figure 5. Screenshot of part of https://lists.malwareworld.com/passwords[13]

4) Telegram Bot

Taking advantage of the API created to ask in the web if a domain or IP is suspicious, it has been created a Telegram Bot (see Figure 6). The bot is available under the name of **@MalwareWorldBot** and can be queried to check if a domain or an IP is suspicious using the command */check*. For example: */check malwareworld.com*, will search if *malwareworld.com* is related with some kind of suspicious activities.



12:09

MalwareWorld
bot

free_web_hosting.txt 12:07

/check 169.50.43.51 12:08

169.50.43.51
Title: Malicious Host
Type/s: KnownAttacker
Reference/s: http://danger.rulez.sk/projects/bruteforceblocker/blist.php
http://lists.blocklist.de/lists/all.txt 12:08

/check luenhang.com 12:09

luenhang.com
Title: Malicious Websites/Malicious Host
Type/s: ❌ BadReputation
Malware
Reference/s: https://fortiguard.com/search?q=luenhang.com&engine=8
https://raw.githubusercontent.com/futpib/policeman-rulesets/master/examples/simple_domains_blacklist.txt
https://cybercrime-tracker.net/all.php
FortiGuard
Search Results
12:09

/check malwareworld.com 12:09

✅ malwareworld.com 12:09

Figure 6. Screenshot using the Telegram Bot[14]

D. Maltrail Integration

Maltrail[8] is a very interesting project that analyzes packets in real time and detects suspicious activities by known bad behaviours and by blacklists. After checking the blacklists of used by this project it was found that it relies on some old blacklists and it does not use some other interesting sources. We created a fork of the project and configured it to only use the blacklist created by MalwareWorld (see the list link provided in Section 3.3.3). In doing so, we leverage and improve the capabilities of Maltrail to analyze in real time and detect suspicious activities by behaviour using an updated compilation of blacklists. The project is called MaltrailWorld and can be found at:

https://github.com/carlospolop/MaltrailWorld

E. *OpenVPN & Docker Integration*

It has been discussed that MaltrailWorld is a very useful tool to discover for suspicious traffic. Then, it was decided to implement a very easy-to-use VPN that analyzes the traffic in real time. To do so, we forked the dockvpn[9] project, which creates a docker container with an OpenVPN server and lets the user request certificates to connect your devices to this VPN. In our fork of the project, we have integrated MaltrailWorld, so all the traffic that goes through the VPN is analyzed by MaltrailWorld. Thus, the user can check if any of the devices connected to the VPN is taking part in some kind of suspicious activities.

This project is very useful for devices that don't have internally integrated network security systems and/or are connected to several unknown networks, like smartphones.

IV. Usage Example

MalwareWorld has been developed to be very easy to use. It is a NodeJS library that can be installed using npm (`npm install malwareworld`).

Having MalwareWorld installed in the computer the only line needed to start it is: `var mw = require('malwareworld')` doing this, the library will start in a second to collect all the information of the blacklists if the function `mw.deactivateBlacklists()` is not used.

In about 5 to 10 minutes all the information will be collected, parsed and saved in database in memory, and the owner will be able to start querying it if a host is suspicious or not. The owner can also set the API keys of the external intelligences so when he checks if a host is suspicious they will be query too.

If you start the tool having configured the API keys and you invoke the function `mw.getGeneralStatistics()` you will receive an output like the following:

```
{ num_ips: 170817,
  num_domains: 131589,
  num_ranges: 48037,
  num_blacklists: 525,
  num_notResponding_blacklists: 0,
  total_unique: 350443,
  repeated: 136959,
  external_sources:
   { fraudguard: true,
       fortiguard: true,
       projecthoneypot: true,
       simplespamcop: true,
       wot: true }
}
```

In this output can be observed that has been found 170.817 IPs suspicious, 131.589 Domains suspicious, and 48.037 IP ranges suspicious. Together are 350.443 suspicious hosts.

The final database has been made by collecting 525 blacklists and all of them has worked. In these blacklists 136.959 hosts were repeated.
All the external intelligences are activated so if MalwareWorld is asked if a host is suspicious, it will query each external intelligence.

Then, if it is asked if 70.32.94.216 is suspicious the response will be like the following:

```
mw.isMalicious("70.32.94.216").then(
    function(result){
      console.log(result);
      }, function(err) {
      console.log(err);
});
```

```
{ malicious: true,
  '70.32.94.216':
   {  title:  'honeypot_tracker/Malicious Host/Spammer',
      type:  [  'BadReputation',  'Spammer', 'Malware' ],
      urls: [ '' ],
      location:  {  lat:  34.0202,  lng:  -118.3928 },
      references:
      [ 'https://fraudguard.io/',
      'https://fortiguard.com/search?q=70.32.94.216&engine=8',
      'https://www.spamcop.net/w3m?action=checkblock&ip=70.32.94.216',
      'https://feodotracker.abuse.ch/blocklist/?download=ipblocklist',
      'https://rules.emergingthreats.net/open/suricata/rules/botcc.rules' ] }
}
```

And if it is asked if the domain malwareworld.com is suspicious the response say that it is not:

```
{ malicious: false, 'malwareworld.com': { title: '', type: [], urls: [], location: {}, references: [] } }
```

It has been explained that MalwareWorld also generates several blacklists and maps where you can see where are located the main focuses of suspicious activities. This is explained in the Section 3.3.3.

For some time we have been recording daily photos of the different maps created in malwareworld.com. Then, after observing them, some curious information has been revealed. Independently of the kind of malicious activity there are always 3 main points where the majority of the suspicious hosts are located: United States, France + Germany + United Kingdom + Belgium (this focus is going to be referred as Europe), and China + Thailand + Japan (this focus is going to be referred as Asia). Depending on the kind of suspicious activity there are more hosts in one site or in others.

In the case of hosts known to be performing attacks, Asia is usually the focus with more hosts (+12.000) and Europe and United States have near 10.000.

The case of hosts with bad reputation is very similar to the previous one, but Asia has more than 30.000 hosts. In the case of hosts related with cryptocurrencies United States and Europe usually has near 2.000 hosts, but Asia has 1.000 hosts or less.

From the data about hosts used to hide the source can be inferred that the three main focus have between 2.000 and 4.000 hosts, but Europe is usually the one who has more. In the case of hosts related to malware (hosts probably infected) the clear winner is Europe again. In Europe there are usually near 2.000 hosts, in United States the number of infected hosts is between 1.000 and 2.000 and it is surprising that in Asia there are less than 500.

Finally, when looking where are the spammers usually located it is found that the winner is again Europe (near 2.500 hosts), closely followed by United States (1.500 - 2.000) and the Asia focus have between 1.000 and 1.500).

V. Conclusions & Future Work

A. Conclusions

MalwareWorld is a free and open source project that collect and merge data from public blacklists allowing to take advantage of all these open sources to check if a host is related with some kind of suspicious activity. It has also integrated several projects similar to it to be more accurate in the verdict. Besides, a live example of how does MalwareWorld works and what does it offer is shown in a web page and a telegram bot.

MaltrailWorld was born by using the blacklists created by MalwareWorld inside the project Maltrail. This tool allows to detect malicious traffic in real time and could be very useful to protect highly unprotected devices like IoT or smartphones.

Finally, MaltrailWorld-VPN was created. This platform is very easy to deploy as it can be done with docker, and provides the privacy of a VPN and the security of MaltrailWorld. These features are very useful to any kind of device that is connected to uncontrolled networks and don't have a lot of security mechanisms (like smartphones).

B. Future Work

The aim of this project is to gather publicly available information from internet in order to find all the hosts related with suspicious activities. Then, it is important to check for new blacklists from time to time. Besides, it is important to locate inside all the used blacklists the ones are useless and delete them.

It is also important to be aware of possible changes made to the APIs of the external intelligences, because if this happens, a change in the appropriate NodeJS API will be needed. One last possible change is the implementation of a captcha. Currently anyone can communicate with the API of MalwareWorld in an automate way, but if a lot of people starts doing this, then the allowed requests that MalwareWorld can make to the external intelligences will not be enough. If this happens, a captcha will be implemented.

References

[1] FraudGuard. https://fraudguard.io
[2] FortiGuard. https://fortiguard.com
[3] ProjectHoneyPot. https://www.projecthoneypot.org
[4] SpamCop. https://www.spamcop.net
[5] Web Of Trust. https://www.mywot.com
[6] VirusTotal. https://www.mywot.com
[7] MSIP. http://www.misp-project.org
[8] Maltrail. https://github.com/stamparm/maltrail
[9] Dockvpn. https://github.com/jpetazzo/dockvpn
[10] Image 1. Screenshot of the web https://malwareworld.com
[11] Image 2. Screenshot of the map showing the known attackers
[12] Image 3. Screenshot of part of https://malwareworld.com/textlists/ips.txt
[13] Image 4. Screenshot of part of https://lists.malwareworld.com/passwords
[14] Image 5. Screenshot using the Telegram Bot
[15] Figure 1. Architecture of MalwareWorld