ECAI 2019 - International Conference – 11th Edition
Electronics, Computers and Artificial Intelligence
27 June -29 June, 2019, Pitesti, ROMÂNIA

Methodology of dark web monitoring

Nicolas Ferry
UFR MIM
Université de Lorraine
Metz, France
nicolas.ferry9@etu.univ-lorraine.fr

Thomas Hackenheimer
UFR MIM
Université de Lorraine
Metz, France
thomas.hackenheimer7@etu.univ-lorraine.fr

Francine Herrmann
LCOMS
Université de Lorraine
Metz, France
francine.herrmann@univ-lorraine.fr

Alexandre Tourette
Alyze.info
Le Pradet, France
alexandre.tourette@gmail.com

***Abstract*—The darkweb is spreading more and more and allows many malicious or criminal activities through the Tor network. To our knowledge, there is no public tool for monitoring the darkweb. We designed a new strategy to monitor the dark web. Moreover, this automatic methodology spares visualization by human being of deviant contents. We have implemented this new monitoring methodology, potentially useful for those who regularly scan this dark side of the Internet. We tested our software [a], we proceeded some experimental tests and we analyzed the results comparing them to previous analogous experimental studies.**

***Keywords*—dark web, security, web monitoring, hidden service, Tor network, onion routing, semantic analysis**

I. Introduction

There are many side of the Internet with more or less legal content. Internet can be summed up in the metaphor of the iceberg: an emerging part, accessible to all, in which web pages are indexed by conventional search engines such as Google or Yahoo, this is the surface web ; and a submerged part, where a minimum of computer knowledge are needed to get there, because these sites are not referenced by traditional search engines, this is the **deep web**. Finally, if we refer to this definition of deep web, we are dealing with it every day, because access to our bank account is not referenced, just like access to our mailbox for example. It is not these last elements that we will be interested in here, we will rather study the deep web in which it is necessary to go through an Internet network that grants anonymity (the Tor network in our case). Intrinsically, the deep web is not necessarily dangerous or outlawed. However, individuals have given themselves the right to pervert the concept initially healthy of protecting privacy, in an environment where all kinds of illegal and sometimes perverse contents overflow. This is commonly called the **dark web**. The dark web is the darkest part of the Internet, and is a subset of the deep web.

The objective of this project is therefore to define a methodology to monitor the dark web, without physically going there, and being potentially confronted (at least visually) with the different shocking contents that this part of the Internet is hiding in its depths. Such a work proved to be very interesting to carry out, and rather useful, because there is hardly not much free software dedicated to monitoring the dark web. The development of a tool to explore and analyze the dark web is crucial, because the search for dark web sites is not easy, although there are some specific search engines on the surface web, or even directly on the Tor network (Duck-DuckGo by default), but they are not always very reliable, especially because of the high volatility of the Tor network and hosted sites. We mainly used software, or programs originally intended for the visible web and impossible to use initially with the Tor network. We deployed various tools to be able to quick analyze a set of sites with a Top-Level Domain (TLD) in `.onion`. In this article, we will sometimes use the term "*hidden service*" to talk about these dark web sites. In the next section we present related analogous works in literature. The main concepts of our methodology is explained in section III. Our technical implementation is described in section IV. Our results are analyzed in section V. Sections VI and VII show possible applications and future perspectives of our work, before our conclusion in section VIII.

[a]https://github.com/N1col4s5742/methodoDarkWeb, Methodology of dark web monitoring available in opensource.

978-1-7281-1624-2/19/$31.00 ©2019 IEEE

II. Related works

Before starting the design phase of our monitoring methodology, we first dove into the scientific literature in order to realize a state-of-the-art review, and to have knowledge in techniques and software already existing. The goal is not to reproduce something similar to what can be found on the visible web, especially on free software platforms.

The dark web is an astounding subject, and we can find many research articles dealing with this field, from different angles and for various purposes. The first article we studied was the one written by the Trend Micro team in 2015 [1]. It was our entry point into the subject. They developed a deep web analyzer named DeWa (for Deep Web Analyzer) to collect data, load and store new urls from multiple sources. The software has a graphical interface to facilitate the exploitation of results. Unfortunately, it is not made available to the community. In this article, researchers also gave an overview of the different themes tackled in dark web sites, such as the drug trade, the purchase of counterfeit banknotes and stolen bank accounts, or the falsification of identity documents. For our part, we have done similar work to confirm their study. Among other writings that caught our attention, we can mention an article explaining the techniques of "crawlers" of a dark web site [2]. They have developed a semi-automatic software to help humans monitoring of dark web. For this, the software will explore the tree of the given site, and establish a whole range of information, including clustering to find properties common to all entities and assign them a category. Their work is very much oriented around the terrorist movement, and they use programs that are quite heavy to put in place. Their classification is therefore based on different terrorist categories, but the general method is interesting to eventually adapt to different points of our problematic. Another research paper presents a method for managing and analyzing external documents on the dark web, such as attachments or compressed files. It is also a "crawler" but it has the advantage of detecting the vicious links that would see a "crawler" analyzing their site, to avoid redirect it towards an infinite loop so that it cannot continue its work [3]. Another article proposes a somewhat different technique for extracting data from a web page, essentially based on the physical layout of the different elements, and the associated texts [4]. For example, the text of a table caption can give important information about the topic covered in the page. This data extraction technique is called LITE (Layout-based Information Extraction Technique), and works in conjunction with a database previously filled in order to guide the robot in its classification choices once the extraction step is completed. Regarding this classification, in [5] the researchers were able to deduce about twenty categories very common on the dark web, such as the sale of arms, drugs, terrorism, child pornography, or the sale of computer fraud among other. To carry out these tests, their starting point of the hidden sites to be analyzed are those that can be found on the surface web, so they crossed urls found in various sources in order to elaborate a consistent base of dark web sites to analyze. Their classification is possible thanks to a large semantic analysis engine, on which they add weight to the relevance of the collected data in order to be able to assign them a category.

These various works previously carried out allowed us to advance on the subject by knowing the existing achievements, but also helped us to make choices at several points in our methodology. However, these different researches are often complex to set up, and this is not necessarily ideal for a quick overview or a preview of the topics covered in a *hidden service*. Moreover, they are frequently specialized in a specific topic, while we want to implement a general classification methodology, without going too much into details, always with a view to a first approach to help a specialist in computer security in charge of monitoring the whole dark web. Moreover, as far as we know, none of these previous studies published any public nor free software. This is why we present a new methodology, which will be explained from section III onwards.

III. New proposed methodology

This section aims to present briefly our methodology for monitoring dark web sites. The conceptual operation of the application is represented in algorithm 1. As it will be explained in the following sections, the principle of the methodology implemented is based on the semantic analysis of a site to extract the keywords and their respective occurrences. We will create a list of sites of the dark web, and will analyze them successively, in order to assign them a category to facilitate the understanding.

Algorithm 1 Conceptual Algorithm

```
1: Get a list of .onion sites ;
2: Make a semantic analysis of the dark web sites contained in the previous list:
3: for all sites in list of sites do
4:     Recover "physically" the site ;
5:     Analyze the site to build a list of keywords and their occurrences ;
6:     Classify the site according to its keywords and the categories presented in Table 1 ;
7:     Save the different steps ;
8: end for
```

The objective of algorithm 2 below is not to detail all the code implemented to build our methodology, but to have a global presentation of the essential functions used to conduct our work, and how are they ordered. For the semantic analysis part, we made the technical choice to use the API named Alyze[b] (see section 4.2). We initially

[b] https://alyze.info, accessed May 2019.

open an SSH tunnel to establish communication between Alyze API and the dark web site sucked up and temporarily hosted on an Apache2 server. On each iteration, we delete the json file returned by the API corresponding to the previous analysis, since it is no longer useful because the essential information is transcribed in the global json file, where all the analyzes performed are recorded.

Algorithm 2 Schematic algorithm of the methodology

Input listSites = list of `.onion` sites
Output global json = json file containing listSites analysis (and previous analyses)

```
Open SSH tunnel ;
for each site in listSites do
    Empty var/www/html ;
    Remove old json from the API ;
    if site was never analyzed then
        Aspire (site) through the Tor network ;
        if any error then
            Move site to var/www/html ;
            Semantic analysis: call Alyze's API and recover json associated ;
            Fill in global json with new data ;
            Classify(site);
        else
            Save url + date in error json file ;
        end if
    end if
end for
Close SSH tunnel ;
```

The classification method is presented in algorithm 3. Six categories have been established based on the work of Celestini and Guarino [5] (see Table 1), and for each keyword, we look for each of them if we can classify it following these six existing categories. In this way, a site can be classified in several categories with a greater or lesser weight depending on the number of occurrences of each keyword. If at the end of the keyword loop, none could be assigned to a category, then the site will be categorized as "Other" (finally corresponding to our seventh category by default).

Figure 1 summarizes and visualizes the different "actors" necessary for our methodology, and how they interact between them. The `.onion` site list analysis is iterative because we loop on each line of the file containing sites to be analyzed, each corresponding to a different site.

The core of the application is the stable part of our system, it will not be affected if someone wants to use this software. The list of dark web sites, as well as the classification part can be modified and configured if the user wishes it. As mentioned in section IV-C, a script is implemented in order to enrich the classification database,

Algorithm 3 Algorithm of the classification part

Input site = dark web site in `.onion`
listOfCategories = list of all categories with their respective words

```
for each keyword in site do
    for each category in listOfCategories do
        if keyword is present in category then
            site.classification.category += keyword .occurrence ;
        end if
    end for
end for
if no classified keyword then
    category['Other'] += site ;
end if
```

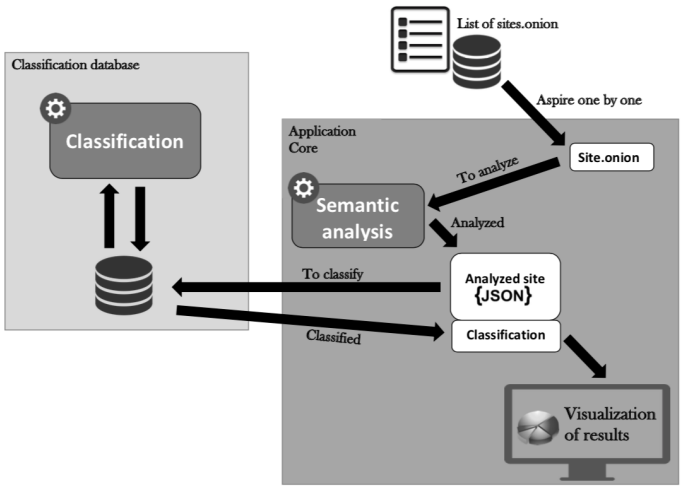

Fig. 1: Sketch of the application.

so it could be possible to erase it and then rebuilt it at the user's discretion thanks to this script.

IV. Development of the methodology

One of our objective, which will be explained in this section, is to be able to get automatically an overview of a dark web site without running a browser on the Tor network, such as TorBrowser. We want to get a panel of dark web sites important enough to be able to draw conclusions concerning the various topics addressed. Our monitoring tool, coded in Python3, is available in open-source on GitHub[a].

A. Get an onion site

The first point of our program is to recover a site in `.onion` for later analysis. For this, we used native Linux tools, such as the `wget` or `curl` commands. With the `wget` program, it is very easy to recover any site. In the majority of cases, launch the `wget siteName` command to download the desired site locally. However, the

Tor network is not accessible directly, and inevitably a native Linux command like `wget` uses the "classic" Internet network by default, and doesn't allow to aspire a `.onion` site. Fortunately, an existing command allows to "torify" the connection, namely to force to make a request on the Tor network in place of the Internet "classic". For this, a simple mention to the command torify is necessary. The command to aspire a site of the dark web thus becomes the following one: `torify wget siteName.onion`. We can also aspire a classic website through this command, but it is not really interesting in this work. It is obviously possible to add arguments to this command, such as the number of attempts if the site is not found, as well as a timer that allows to determine the time during which the site is looked for if is not found instantly. These settings allow our program to avoid deadlocks on a site that does no more exist. It is also necessary to recover the error codes which make it possible to know if the site was well sucked up, or if, on the contrary, an anomaly occurred on a site, in order to skip over this problematic site, with the aim that the global program can always continue its process. The `wget` command returns 0 if all went well, but with the use of `torify` it happens regularly that a code equals to 8 is returned, meaning that the server has issued an error response. This comes from the fact that `wget` was not designed to be used with the Tor network, but just capture error code 8 as a success code in our case. Note that using the command as presented above, we only get the `index.html` page of the site, but this is more than enough to get an idea of the topic on the site. Considering the whole site is an easy improvement presented in section VII. In the case where recovering the site is not possible and has generated an error, we record it in an error file with the url and the date stamp.

B. Analyzing the recovered site

Once the site is sucked, we have to establish a semantic analysis process in order to count the most frequent keywords used. We investigated for free software which could help us on this aspect and we finally choose Alyze website and Alyze API[b]. We give to Alyze API an URL in entry and Alyze website displays an array with the most frequent occurrences. Keywords are sorted on account of their weight depending of the HTML markup of the web page. Alyze API was initially created to improve the referencing of a given website but its function is really interesting and useful in our researches because it returns keyword's occurrences respective to a website.

Concerning the analysis of a dark web site the work logically come out of the one explained in the section IV-A. We are able to get a hidden website now we have to analyze it through Alyze API. In order to do that, we use an Apache2 server. The dark website we aspired is transferred in the local folder of Apache `/var/www/html` in `index.html` file as it is configured in Apache's configuration file. Once the website is dropped on the right place in the Apache server, we have to establish communication between Alyze API and the dark website locally hosted. To do so we used an SSH tunneling to redirect a port. With SSH, we can use *Remote Port Forwarding*. It allows to redirect a local port towards a distant SSH server. The syntax of the command is as follows:

```
ssh -R <remote_port> : <local_ip> : <local_port> user@host
```

- `-R`: *Remote Port Forwarding*;
- `remote_port`: the port of the distant computer which will be redirected. It is the port 80 in our case because it is the port that allows the consulting of HTTP server from a browser;
- `local_ip`: the share server (`localhost` in our case);
- `local_port`: the port of the computer which will be redirected;
- `user@host`: the name of the distant computer on which runs the SSH server;

By launching this command with the right arguments, we open a SSH tunnel, and an url address is returned by the program and gives access to the content of `var/www/html` from a distant server. By entering the url address on Alyze API, and by starting an analysis, the software can now run his program on a dark web website.

Once the json file has been returned by the Alyze API, we have to look over it in order to extract the useful information for our methodology. To do so, we create a second json file (but the use of a database may be an alternative, see section VII), and save for each input the name of the website, its url address and the date of analysis. Then, we sort the keywords and their occurrences (see figure 2). We took the decision to only keep the ten most used keywords. Indeed, it is enough to have an idea about the topics dealt with. The global json file is completed as we analyze website.

C. Classification of analyzed sites

Once the website has been vacuumed up and analyzed, we have to classify the site in one or several categories. Based on our own results as well as on results of researches previously cited [1], [5], the main topics present on the dark web are the sale of illegal merchandises, the crimes, pornography and pedo-pornography, and traffic of anything related to money. Consequently, we decided to create six "super-categories" less specific than the ones referred to in [5]. We grouped some categories from [5] because they where dealing with the same topic. For example, the arms and terrorism are two distinct categories, according to their classification model, but we choose to refer to it as "Crime" in our methodology. Our six categories are Drug, Market, Money, Crime, Virus and Adult as in table I.

With these six categories, we can nearly classify each topic on the dark web. We defined them so the seventeen specific categories presented in the article [5] can fit in them because they are more generic.

TABLE I: Presentation of the 6 categories of classification.

Name of category	Subjects discussed (not exhaustive)
Drug	Any term related to drugs. (ex: cannabis, heroin, drug, etc.)
Market	Any term related to the sale and purchase, illegal sales platforms, etc. (ex: marketplace, sold, buy, etc.)
Money	Any term related to money and currency, stolen or not. (ex: bitcoin, credit card, Paypal, etc.)
Crime	Any term related to crime, hitmen, assassinations, etc. (ex: hit men, murder, kill, etc.)
Virus	Any term related to computers, cyber-threats, exploits, hidden-wiki, etc. (ex: virus, malware, onion, etc.)
Adult	Any term related to pornography. (ex: pornography, sex, etc.)
Other	All that can't be classified previously

In order to assign a word in a category, we set up a database in json format (here, a genuine database can be used, as mentioned in section VII). To create the database, we wrote a Python script which calls an API from WordAssociations website[c]. The principle of this website is to associate to a given word matching terms. So if we enter any word, the algorithm will return a list of words linked to it. The call to the API is always the same, with an url address where we only have to input the value `text` by `text=word`, where `word` is the original term from which we want a list of linked terms. We can create a database semi-automatically thanks to this process, according to previously defined categories. For each category, we have to enter some words closely linked to it, for example for the "Drug" category, we enter "cannabis" and we get a list of twenty or so words. Thus, for a category, by entering some reasonable words, we get a well-stocked list of terms. We mainly focused our work on an English semantic because we noticed and the Trend Micro study [1] confirmed that over 75% of dark web sites are in English. Now if we want to add French entries in our database, it is possible to do it, we only have to change `en` by `fr` in the url address of the API and to entry a French word in the Python script, and it will broaden the database with foreign words. Hence it is possible to create a multilingual list of categories.

It is possible that one word is sorted in different categories at the same time, because of the way we filled our database. For instance, the word "buy" is legitimate to appear in the "Money" and "Market" category. This is intended because the developed application is rather a probability indicator of the content of a website, than an exact indicator. Of course, it is not difficult to create a designed database, or to refine the existing one by keeping only one occurrence of words classified in different categories at the same time.

Once the database is ready to use, to classify a website we were somehow inspired by the work on extracted data classification as explained in the article of Celestini and Guarino [5], that is to say the use of weights. They arbitrarily gave weights to the words, while we decided to use the number of occurrences as a weight, since it reveals more information. So it is not sensible to set arbitrary values when we already have ones. For each keyword extracted from the website, we see if it belongs to a category and we add up the score of a category to the weight of the keyword. In order to maximize the results, each word present in a website title is given an arbitrary weight (here, it is 10, because it shows how important is the word without deforming the results). Finally, for each website, we launch this algorithm, and complete the initial json file with a new entry containing this classification table. If for a given site, none of the keywords fit in one category, they are classified the "Other" category. Generally, it does not happen because our database is well-stocked, except if the analyzed website hardly contained any text, as it can sometimes happen on dark web site. In figure 2, it is shown how it works:

```
{
 "NameOfSite": [
   "infos": {
     "nom" : "nameOfSite",
     "url" : "url.onion",
     "date" : "dateAnalyse" },
   "keywords": {
     "keyword1" : "occurrence1",
     "keyword2" : "occurrence2",
     ... },
    "classification" : {
      "Drug" : occurrencesDrug,
      "Market" : occurrencesMarket,
     ... }, ...
 ] }
```

Fig. 2: Example of JSON file analysis

V. Exploitation of the results

A. Creating an onion sites database to analyze

The work of Celestini and Guarino [5] consists in creating website base by directly analyzing urls of *hidden services* present on surface web and by crossing the collected data from different sources. As they did, we implemented a Python script allowing to collect every url in `darkWebSite.onion` form from a traditional website, called Ahmia[d]. We make sure that the collected sites from the dark web have 16 characters in their url address, following the standard, and have a TLD in `.onion` in order not to block the site list vainly. The entry point to the dark web is the surface web, because we find dark web url addresses in the surface web. Some *hidden wikis* exist and can be accessed to from the Tor network, for instance "Torch" is the most used. In late April 2019, we collected

[c]https://wordassociations.net/en, accessed May 2019.

[d]https://ahmia.fi/address/, accessed May 2019.

more than 3,000 dark web sites among the tremendous numbers of sites because we used only one source to collect them. We analyzed several samples randomly picked from this list, because Ahmia sorts them alphabetically. This list is only used to lead studies concerning the dark web, but a person working to monitor the dark web can add in this file sites that they just found and analyze them. As we broaden the list, we can analyze a maximum of *hidden services* and in theory analyze the entire dark web, but it will be difficult to achieve because urls addresses change frequently and sites have a limited life span, thus a performing bot and great means are necessary.

B. Graphical interface for consulting results

It is useful to create a graphical interface to analyze the results we obtained. This is why we added a pie chart which shows all the websites analyzed by Alyze API and for each of them, all the occurrences of the keywords and the classification according to the categories present in the table I. This chart is effective for a quick visual analysis, indeed it is widely used in the articles we read to do our research. It can display all the websites that were analyzed on a given date, so the person who made the analysis will not be bothered by previous analyses. Finally, it is possible to have a summary of all the websites that were analyzed, and to go over all the occurrences for each category with an eventual pie chart visualization.

C. Results Analysis

We created 4 samples of a hundred website each. Since dark web sites have a short lifespan we can estimate that for a hundred url addresses, only 20% do not exist anymore. So we managed to have around 100 websites that work for each sample.

TABLE II: Number of occurrences by category and by sample.

	Samples				
Category	**1**	**2**	**3**	**4**	**Total**
Drug	651	412	517	476	2056
Market	724	833	863	854	3274
Money	833	1081	1065	1105	4084
Crime	439	129	120	179	867
Virus	1202	1302	1895	1603	6002
Adult	2377	342	311	2738	5768
Other	23	26	17	30	96
Number of analyzed sites	102	125	98	100	425

Table II shows the number of occurrences of each category for all 4 samples. The most reliable sample is the third because it has fewer occurrences that were not classified in any categories, almost 90% of the occurrences were assigned to a category, other than in the "Other" category. As we mentioned previously, the samples were made randomly, so a website can be in different samples at the same time. Figure 3 shows the collected data in pie charts with values in percentage.

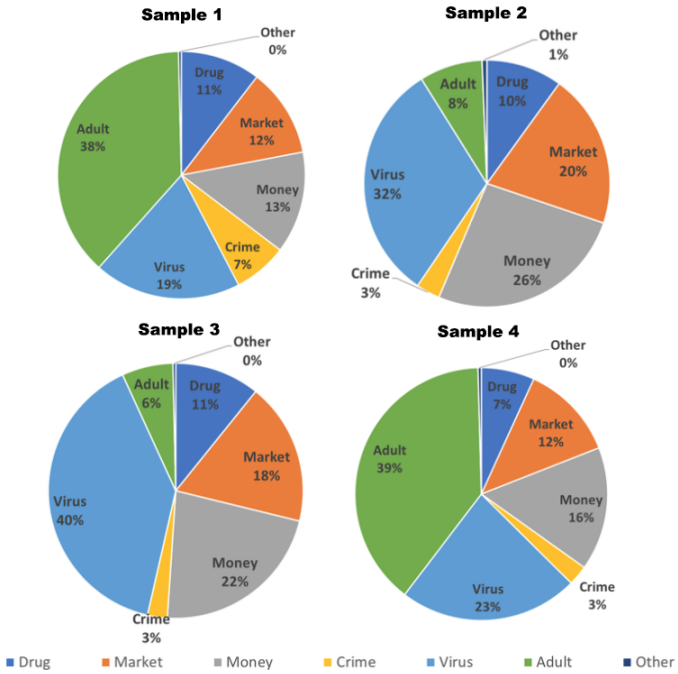

Fig. 3: Results of our 4 samples.

We can notice that samples 1 and 4, and samples 2 and 3, have similar results. Finally, we can observe coherent results from the websites that were analyzed. Except for the "Adult" section which varies from one sample to another, the other categories appear with the same proportion. It is explainable because for instance the "Crime" category includes all kinds of assassination services, contract killers, etc. which are very specific services and their applications in real life are limited, this is why this category has less occurrences despite having stable but low percentages. On the other hand, anything related to money traffic is frequent, with real-life applications for darknet users. This is why the "Money" section has stable results. The "Drugs", "Money", "Market" and "Virus" categories are widely represented in each sample and this results echoes the observations made by the Trend Micro team [1]. Concerning the "Adult" section which varies, it confirms the fact that in general these websites are omnipresent on dark web, and this software allows to have an idea of the content of these websites without having to see shocking images. The websites in the "Other" section had for the most part no content at all after inspection, thus no interesting keyword appeared. Some of them are German, or Spanish websites, which can not be classified by our model.

VI. Potential applications

This methodology could be helpful for an expert in IT-security, who wants to have an insight about new dark

web sites before visited them physically. We imagine this application more like a probe in order to give instruction about different *hidden services*. The clear conscience of the human being remains essential and this methodology has not the ambition to substitute it. This is the reason why the development of this analysis method must be see as a first help for a person who is in charge of dark web security and supervision.

Thanks to enrich the global json file, we could have a quicly preview of all the dark web in theory, and a summary of the classification of these sites according to our different categories. However, it is very difficult to analyze the totality of the dark web, because web sites are very volatiles. Police authorities managed to close some illegal dark web sites, but most of them open again with new url address a few days later. We noticed that, to establish our samples of a hundred websites, we had to consider 20% more sites, because many are unreachable, even when their url address was get recently. The list of these nonfunctional sites is kept in order to possibly save their url address, or make futur tests in this list, in the event of some would be functional again.

VII. Future prospects

In this methodology, we made a semantic analysis. It could be interesting to do a classification based on pictures hosted in dark web sites. Indeed, there are already artificial intelligence research to recognize and classify pictures, so we should use these works in order to classify the website according to its photographic content. It would be useful, in additional of the semantic method, or independently of the semantic method, including for blogs' analyze which are websites without a lot of semantic content.

To go on our work, it would be usefull to replace json files by real databases. It would be judicious to redo the classification part of websites, to add categories, and especially refine the list of words related to each category.

Initially, our idea was to analyze the whole sites. It is technically possible to complete the `wget` command with different parameters in order to get a mirror website, perfectly identical to the original site. This technique works good, but require a lot of process time. That being said, it would be only an optimization compared to the work done because it is enough to analyze the `index.html` in most of the cases.

It is also possible to consider our software to analyze continuously the dark web. Indeed, a bot could permanently scan the dark web, and add to our `.onion` list new url addresses that it would discover during its process. In this way, we could have a maximum of dark web sites in our database, and we could have a global overview of the topics mentioned in the dark web. It would be also possible to specialized the list of keywords in order to guide fields to monitor, and follow the current trends in real time. So, the developed software can be used in addition to another.

VIII. Conclusion

As a conclusion, we can state that the dark web is full of illegal and harmful contents. The anonymity offered by the Tor network allows malevolent people host almost freely outlaw content, which we cannot find on the surface web. Drug commerce, credit card frauds and the buying of stolen account, commerce of counterfeiting goods, sale of harmful services such as assassinations, or to buy IT viruses are example of all the practices of this parallel commerce. The last few years, police forces paid close attention to the dark web, because a genuine underground economy is developing and is easy to access thanks to TorBrowser. In addition to economic problems linked to the dark web, a sense of insecurity is growing and authorities cannot leave it a lawless area. In this context, and this perspective of surveillance we did our researches, allowing to create an additional tool to help the IT experts. It is interesting to quickly and briefly visualize the topics that are dealt with in different *hidden services*, and to classify them under different categories. This research also allowed us to test our methodology on list of dark web sites, in order to make statistics, and to attest to the previous researches that had been led.

References

[1] Trend Micro, Balduzzi M., C. V. T. M. (2015). Cybercrime in the deep web. Black Hat.

[2] Chen, H., Chung, W., Qin, J., Reid, E., Sageman, M., and Weimann, G. (2008). Un- covering the dark web : A case study of jihad on the web. J. Am. Soc. Inf. Sci. Technol., 59(8) :1347– 1359.

[3] Zhou, Y., Qin, J., Lai, G., Reid, E., and Chen, H. (2006). Exploring the dark side of the web : Collection and analysis of u.s. extremist online forums. In Mehrotra, S., Zeng, D. D., Chen, H., Thuraisingham, B., and Wang, F.-Y., editors, Intelligence and Security Informatics, pages 621–626, Berlin, Heidelberg. Springer Berlin Heidelberg.

[4] Burget, R. (2007). Layout based information extraction from html documents. In Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), volume 2, pages 624–628.

[5] Celestini, A. and Guarino, S. (2017). Design, implementation and test of a flexible tor-oriented web mining toolkit. In Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS '17, pages 19 :1–19 :10, New York, NY, USA. ACM.