Spotlight: Analyzing Content, Infrastructure, and Security of Hidden Services on Tor

Juno Mayer, Seth Tal, Alex Marozick

ABSTRACT

The Dark Web is a subset of sites on the Internet which operate through special protocols that create an anonymous and encrypted ecosystem for its users. The Onion Router (Tor) network is the most popular Dark Web network containing many discussion forums and anonymous marketplaces. Due to Tor's provided anonymity and lack of global moderation, fraudulent behavior and malicious users are prevalent across many hidden services, leaving users vulnerable to many forms of attacks. As a result of these security risks, users are less likely to gain a first-hand understanding of Tor's content. We build and deploy a web crawler capable of accessing Tor to showcase political content and sentiment. As an example, we show that the word "Trump" is associated with negative sentiments across the 8Kun anonymous forum. Furthermore, we explore how users and proprietors of Tor marketplaces maintain security and anonymity while making necessary exchanges of sensitive data.

1. INTRO

The Internet is broken up into three layers: The Surface or Clear Web, the Deep Web, and the Dark Web. The Clear Web consists of any website on the World Wide Web that can be indexed by search engines. Surprisingly, only 0.03% of the Internet falls into this category. (CITE) The vast majority of the internet resides within the Deep Web and includes benign sites used by most Internet users that are protected by security measures (Email or any other password protected site falls into this category). The Dark Web is a small subsection of the Deep Web which operates through special encryption protocols implemented on top of HTTPS. Multiple Deep Web network protocols exist, the largest of which is Tor (The Onion Router Protocol). Tor gets its namesake from the layered end-to-end encryption protocol it employs to anonymize user data: When a user makes a request on the Tor protocol, their traffic is routed through a random selection of relay nodes, each of which adds a layer of encryption. As the traffic reaches an exit node, these layers are peeled back and contact with the desired endpoint is made. This prevents third parties from knowing the true origin IP of a request.

It is important to note that this protocol is not immune to attacks which de-anonymize users. Due to the open source nature of Tor, anyone can open a relay node to contribute to the health of the network. However, if an entity owns many relay nodes of the network, they can monitor traffic and identify Tor sites called hidden services (HS). In a 2015 paper, Dr. Gareth Owens used this technique to catalog activity levels of hidden services. This work showed that a large amount of traffic on Tor included illicit marketplaces and political discussion forums.

We focus on these two facets of the Tor network. We seek to understand its political content through the creation and deployment of Spotlight, a web crawler and text analysis framework. We gather text content of hidden services and employ frequency analysis to gather common sentiments behind political language. We show that Biden, Trump, God and Twitter are among the most common discussion topics on the 8Kun anonymous forum. We apply context frequency analysis to show that negative sentiments surrounding Trump are most common. We also investigate the network security best-practices of Tor marketplaces. We discuss how marketplace vendors maintain privacy and security while selling products on Tor, as well as how marketplaces enforce trust systems and prevent fraud on their platforms.

2. BACKGROUND

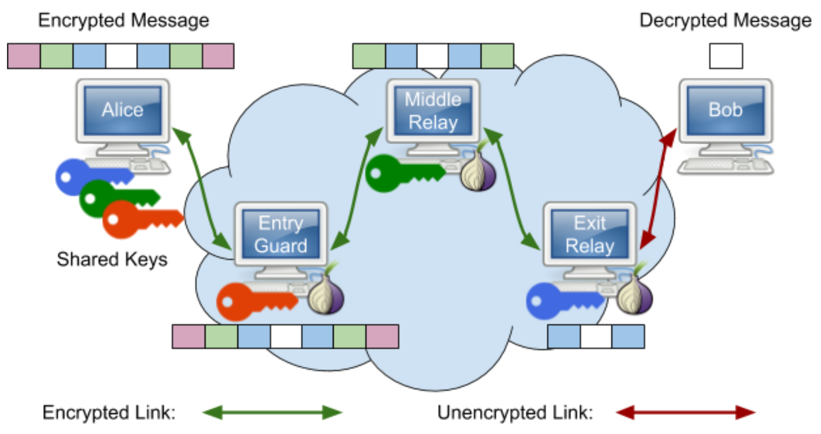
2.1 Tor Architecture

Onion routing, the core principle of Tor, was first invented through US Naval research in the mid 1990s with the purpose of protecting U.S. intelligence communications online. The first version of the Tor network was developed in 1997 as a means to provide anonymity to users by trafficking connections through multiple routers such that a user's location is obfuscated. (The Tor Project: Privacy & Freedom Online)

Within any given connection on Tor, there is a client, server, and three types of nodes known as entry/guard, relay, and exit nodes. Onion routing sends messages encrypted multiple times using three different keys derived from these nodes and are encrypted in layers. These layers of encryption do not add any length to the message that is being passed and by standard are 512 bytes that are wrapped with headers of three different protocols: TLS, TCP, IP. (Computerphile)

When a client initiates communication with a server, the client will be connected through intermediate nodes which act as a proxy. The last node in this chain directs the traffic to the intended server. A key exchange algorithm is used to encrypt/decrypt the message as it traverses the nodes: The client acquires a public key from each intermediate node and the message is encrypted in a layered fashion. The outer layer is encrypted with the shared key for the entry node and the innermost layer is encrypted with the shared key for the exit node. As the message traverses through the nodes, it is decrypted layer by layer, resulting in a fully decrypted message being passed from the exit node to the server. The process is reversed when the server wants to send a message back to the client. Each intermediate node only contains information about its predecessor and successor, preventing an attacker from tracing the origin of a request given data collected from an intermediate node. (Computerphile)

Figure 1: Key Exchange Across Tor Network



One of the most important security measures that Tor employs is relay node randomization. By choosing a random set of relay nodes for a connection, attackers cannot predict connection paths or analyze traffic to obtain user location. Tor accomplishes this by following a set of constraints. First, Tor must choose an exit node with available bandwidth from Tor's consensus, a document containing a list of trusted nodes spread amongst a set of trusted servers tasked with monitoring the global health of the network.(Applebaum) Next, Tor chooses entry and middle relay nodes so that each node belongs to a different operator. To accomplish this, Tor follows a set of constraints:

(1) Each relay must not be in the same /16 subnet (So Tor only uses one IP in the range from 10.12.0.0 to 10.12.255.255 [10 and 12 can be any number]).
(2) Operators are expected to declare relays as their 'family' and Tor will avoid picking two nodes within a single family.
(3) A client can also set a family option which Tor will adhere to

Additionally, no two nodes are chosen within a single path which is similar to the rule in which no two nodes are chosen from the same family. Routers that are non-running or non-valid are not chosen unless they have been configured to. By default non-valid routers are allowed in "middle" and "rendezvous" positions. (R. Dingledine and N. Mathewson)

2.2 Tor's Weaknesses

The Tor network is not immune to security attacks. For example, there exists vulnerability between the exit node and the server in which sensitive or revealing information can be eavesdropped due to a lack of encryption. This can be accounted for by employing a secondary form of message encryption. For example, a client and server can use PGP encryption in conjunction with onion routing to obfuscate data between the exit node and server. Another point of weakness is the exit node's shared key. If an attacker were to intercept the exit node key, they would be able to fully decrypt the message. Although the attacker would be able to read the message and see the server side activity, the identity of the user would not be revealed as no node within the system contains the full path back to the user. Likewise, if the input node and client shared key is exposed, the message would still be incoherent as it still is encrypted with the keys of the intermediate and exit nodes.

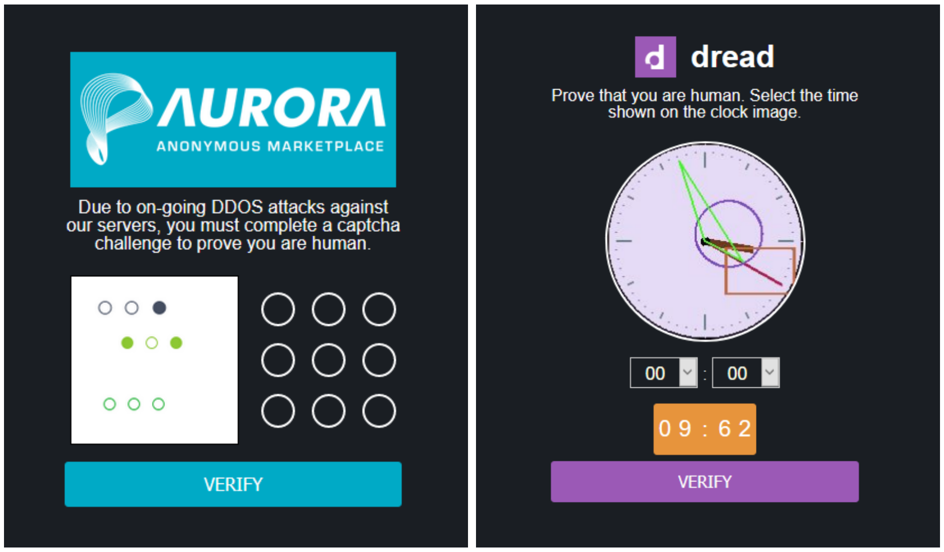
Furthermore, an attacker can de-anonymize data by injecting malicious nodes into the network which scrape network traffic. This method of attack is possible through controlling all intermediate nodes or the entry and exit nodes within any given connection. Through traffic analysis, the attacker would be able to correlate traffic passing through the nodes and trace the connection back to the user's origin. This problem remains unsolved but is unlikely to occur due to the high amount of traffic and thousands of intermediate nodes that are used. Even if an attacker were to gain control of a full node path, connection tracing is not guaranteed at high traffic volumes as each node on the network is used for many lines of communication.

While individual hidden services can protect against DoS attacks, the Tor network as a whole is vulnerable to DoS attacks. The Tor Protocol contains Directory Authority (DA) nodes, which collect information about relay node heath. Every hour, these nodes form a consensus on which relays to route traffic through based on their health. This process is essential for maintaining connections across the Tor network. If this consensus cannot be reached, clients and servers will not be able to communicate due to a lack of usable relay nodes. (Stem Documentation) In January 2021, a large-scale DoS attack was carried out on the Tor network by occupying the bandwidth of DA nodes, preventing intercommunication and the ability to reach a consensus. (Dark Net Daily, Mental Outlaw) This attack resulted in the denial of service to all HSs operating on the Tor V3 protocol for two days.

2.3 Robustness of Hidden Services

Operators of hidden services implement various methods of protection on their sites to prevent DoS attacks and malicious users. One method of DoS protection is replacing the landing page of a hidden service with a Captcha. When a user navigates to a hidden service, they must fill out a prompt based on a series of images before accessing the service. Captchas prevent repetitive and scripted attacks on hidden services as they mandate a human be behind any given web request. Landing page redirects are also used to prevent scripted access to hidden services. If a script requests an onion URL using this method, a placeholder webpage will be returned instead. However, in the Tor browser, after making the first request, the placeholder page forwards the user to the landing page of a website. This limits the frequency that a webpage can be accessed as a waiting time is imposed by the redirect operation. User authentication is a final method of keeping malicious users out of a hidden service. By placing an overhead on accessing a forum via user registration, the hidden service can deter malicious users from spamming forums with bot accounts. These methods all impact the ability to be scraped using our crawler. We further elaborate on these impacts in the Methodology section.

Figure 2: Examples of Captchas on Forums and Marketplaces


Due to on-going DDOS attacks against our servers, you must complete a captcha challenge to prove you are human.

Prove that you are human. Select the time shown on the clock image.

2.4 Related Work

We are not the first to programmatically crawl the Dark Web in search of information. Government agencies and criminologists have high interest in collecting data from the Dark Web to gain further insights into criminal behavior that occurs on hidden services and take actions towards apprehending perpetrators. As such, there exists a body of work discussing approaches to minimizing risk when collecting data en masse from hidden services. Dr. Richard Frank at Simon Fraser University's International CyberCrime Research Center developed a crawler "for the purpose of collecting, classifying, and interpreting extremist content online and on a large scale" using machine learning sentiment analysis. Dr. Gareth Owens at the University of Portsmouth conducted a survey of the Tor network by opening 40 relay nodes and compiling a comprehensive list of hidden services over a 6 month period. Owens classified these services and catalogued frequency of traffic per category.

These works were possible due to an ample amount of time and computing power. As we do not have the means to open relay nodes or the computing power to crawl Tor's entire Distributed Hash Table, we aim to crawl specific hidden services in search of political discussion. The project's inception took place during the 2020 Election. During this time, a large amount of political activity and discussion took place on the Tor forum 8Kun, an anonymous image board known for fostering the extreme right wing conspiracy QAnon. We aim to target 8Kun as a source for deriving political sentiments on Tor.

3. METHODOLOGY

3.1 Crawler Architecture

We developed a web crawler capable of interfacing with Tor. The final product is *Spotlight*, a .NET console application written in C#. Spotlight takes URLs as input and scrapes text content from these URLs for analysis. C# is widely used for Web Development, and as such has extensive built in support for handling HTTP requests. We simplified the process of crawling through the Tor Network and scraping text content by utilizing the following libraries: TorSharp, Abot, and HtmlAgilityPack. We employ these libraries to route HTTP requests through Tor, crawl web pages, and parse text content.

Figure 3: Spotlight System Architecture Diagram

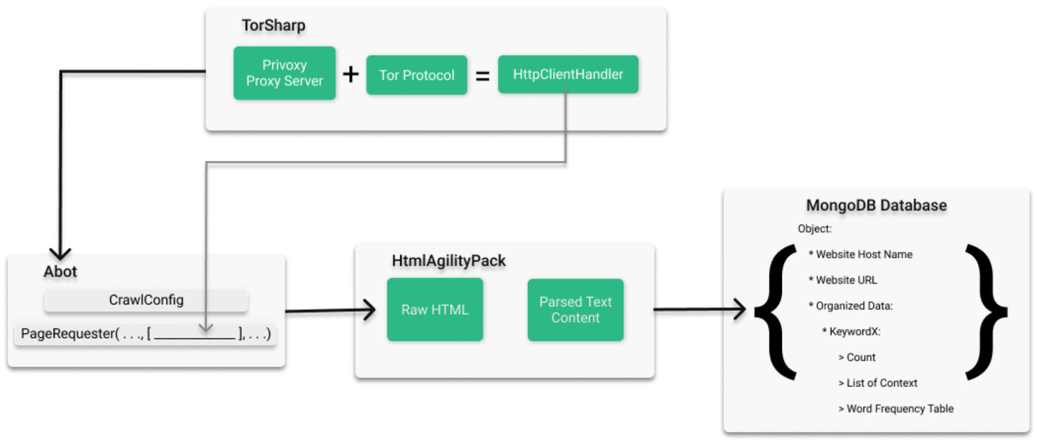
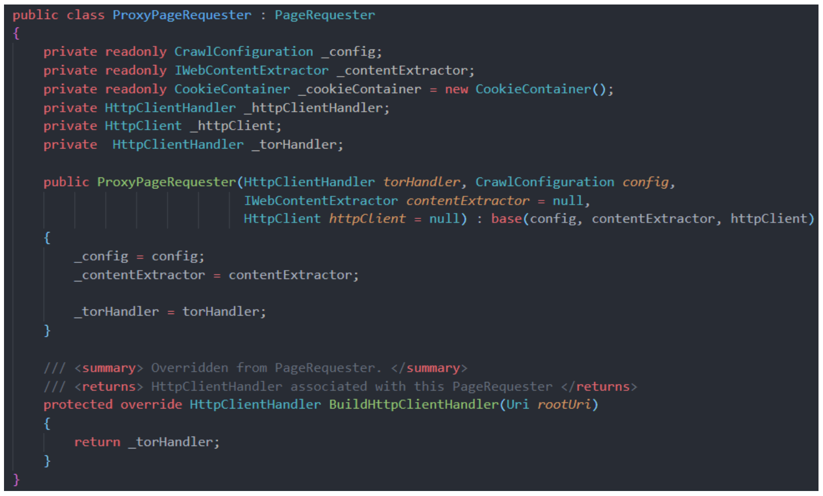


Figure 4: Custom Tor Capable Page Requester



```
public class ProxyPageRequester : PageRequester
{
    private readonly CrawlConfiguration _config;
    private readonly IWebContentExtractor _contentExtractor;
    private readonly CookieContainer _cookieContainer = new CookieContainer();
    private HttpClientHandler _httpClientHandler;
    private HttpClient _httpClient;
    private  HttpClientHandler _torHandler;

    public ProxyPageRequester(HttpClientHandler torHandler, CrawlConfiguration config,
                              IWebContentExtractor contentExtractor = null,
                              HttpClient httpClient = null) : base(config, contentExtractor, httpClient)
    {
        _config = config;
        _contentExtractor = contentExtractor;

        _torHandler = torHandler;
    }

    /// <summary> Overridden from PageRequester. </summary>
    /// <returns> HttpClientHandler associated with this PageRequester </returns>
    protected override HttpClientHandler BuildHttpClientHandler(Uri rootUri)
    {
        return _torHandler;
    }
}
```

To route HTTP requests from Spotlight through Tor, we deploy a local proxy server using the TorSharp open-source library. TorSharp uses Privoxy to instantiate the proxy server and takes Http requests routed through the client and anonymizes them through the Tor protocol. Privoxy is particularly useful since it does not cache web documents as they are being routed. The proxy server is then accessed by Spotlight through a custom C# Http Client Handler (HCH) object.

Spotlight employs the Abot crawler framework to make requests to URLs and harvests data. Abot's modularity allows the user to create components of a crawler with custom behavior utilizing an object oriented model. Abot uses a PageRequester object to define the HCH it uses to make requests to a page. As shown in Figure 4, we create a custom PageRequester class with the ability to accept our custom HCH upon instantiation. Spotlight is initialized with our PageRequester and Tor HCH which Abot uses to crawl hidden services on the Tor network and harvest raw HTML text content.

This method presents multiple challenges:

1. Many modern web services utilize dynamic JavaScript rendered in the browser to display text content. On these sites, the raw HTML will contain unrendered JavaScript code instead of plain text. To surmount this problem we use the Abot extension library AbotX to render JavaScript from dynamic web pages.

2. Not all text content on a given webpage is written by a human. The goal of Spotlight is to analyze sentiments from user interaction, thus we must exclude all text content of UI elements such as buttons and dropdown menus. To ignore this content, we employ the HTML Agility Pack (HAP) library to filter text content from harvested HTML. HAP converts raw HTML to a tree of C# objects which can be queried and filtered. To exclude UI elements, we filter out all HTML Form elements and only query for text content within the Body element to prevent gathering data from a site's header and footer. The resulting product contains user-generated text content.

With these problems solved, we then store all of the text content gathered into a serialized document format called BSON (Binary JSON) on the MongoDB platform. (Storage conventions elaborated in Sentiment Analysis section)

3.2 Selection of Hidden Services

Along with the Dark Web's inherent lack of accessibility comes the act of discovering what the URL for a given service is. In our search for web content, we discovered that many individual users have cataloged lists of active onion URLs. One such site catalog is an entity known as "The Hidden Wiki". This wiki contains an organized list of sites and services on Tor such as: Marketplaces, Forums / Blogs, Hacker Sites, and many more. The wiki has an accessible version hosted on the Clearnet, but has an uncensored Tor-Network-equivalent only accessible as an onion URL. Many other wikis similar to The Hidden Wiki exist, and acted as the primary means of accessing different services and forums on the Tor network.

As mentioned previously, we sought to employ Spotlight to scrape for political content on the Dark Web. Among the list of sites we compiled we specifically chose 8Kun as our catalyst for content. 8Kun was chosen for targeted scraping due to its high volume of web traffic from real active users. Being one of the largest platforms for discussion on the Dark Web, it offers a unique opportunity to collect and analyse consistent and meaningful data. Furthermore, it acts as a means of showcasing aspects of the Dark Web that aren't inherently illicit.

Spotlight in its current form lacks the capability to handle any forms of authentication designed to keep non-human entities from accessing a given website. We found that many of Tor's hidden services employ measures to prevent scripted access such as captchas, and landing page redirects. Captchas are automated turing tests aimed at combating automated bot attacks. There are many reasons to employ the use of redirecting URLs, but most commonly on the Dark Web this tactic is used as a security measure to reduce the amount of bot-related-traffic trying to reach a given webpage.

3.3 Sentiment Analysis and Data Storage

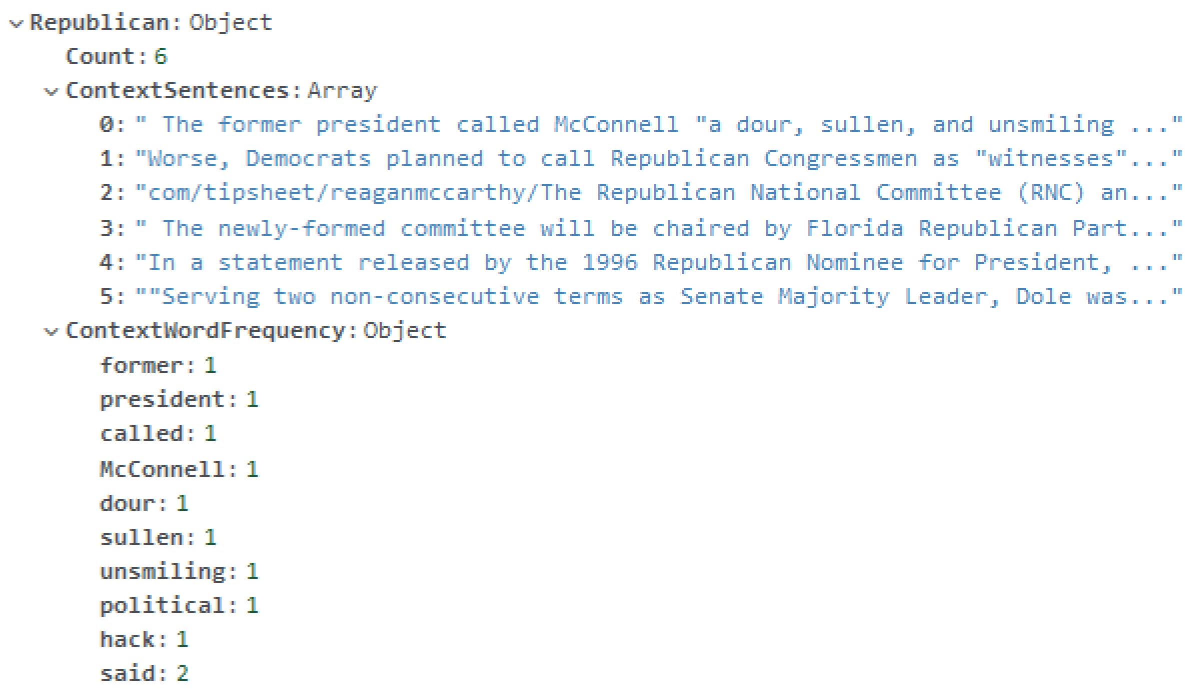
Spotlight scrapes text content which is then analyzed to shed light on its sentiment. This sentiment analysis works hand-in-hand with our method of backend storage. After we parse text content from a given web page, the raw text is run through multiple functions to organize the data into an intuitive format for understanding the general sentiment of the content as a whole. The order of events is as follows:

1. All of the raw text is passed through a function which identifies keywords and stores the number of times each keyword occurred along with its context: a list of all sentences in which the words were used. No repeating sentences are allowed, and a predefined list of constraints is provided as input to parse out potential keywords which may not possess inherent sentimental value.

2. We then iterate through the N most common keywords identified by our program and apply frequency analysis to the context sentences of each keyword. Through this, we determine the most common topics and sentiments associated with a given keyword.

After algorithmically identifying keywords and caching their context, we generate serialized documents (Figure 5) and upload them to our MongoDB database. This document consists of: the N most common keywords, their frequency, the context from which these words are used, and the word frequency table generated from the context.

Figure 5: Example of Stored Data



```
v Republican: Object
     Count: 6
   v ContextSentences: Array
        0: " The former president called McConnell "a dour, sullen, and unsmiling ..."
        1: "Worse, Democrats planned to call Republican Congressmen as "witnesses"..."
        2: "com/tipsheet/reaganmccarthy/The Republican National Committee (RNC) an..."
        3: " The newly-formed committee will be chaired by Florida Republican Part..."
        4: "In a statement released by the 1996 Republican Nominee for President, ..."
        5: ""Serving two non-consecutive terms as Senate Majority Leader, Dole was..."
   v ContextWordFrequency: Object
        former: 1
        president: 1
        called: 1
        McConnell: 1
        dour: 1
        sullen: 1
        unsmiling: 1
        political: 1
        hack: 1
        said: 2
```

4. RESULTS

4.1 Application of Crawler to Anonymous Forums

Given that our crawler aggregates data for word frequency analysis, we target forums that group discussion topics, enabling us to more accurately determine the sentiment for each topic. This allows us to reduce the scope of infrastructure built for any given site while maintaining accuracy. We target 8Kun as a case study as it contains an abundance of political discussion across many boards. It is important to note that data was gathered around the time of the 2020 election in which Biden was elected over former president Donald Trump. We expect to see the hostile and polarizing environment created during the election reflected in the language of 8Kun users across politically themed boards.

Figure 6 and Table 1 show 8Kun users' most frequently used words that have intrinsic value across the 10 most popular boards. This depicts the most commonly discussed topics on 8Kun. We identify that these forums discussed politically charged topics such as the recent election, religion, and racially insensitive material. As expected, Biden and Trump were the most frequently used words, along with Twitter, President, and Administration.

Figure 6: 8Kun Topic Frequency

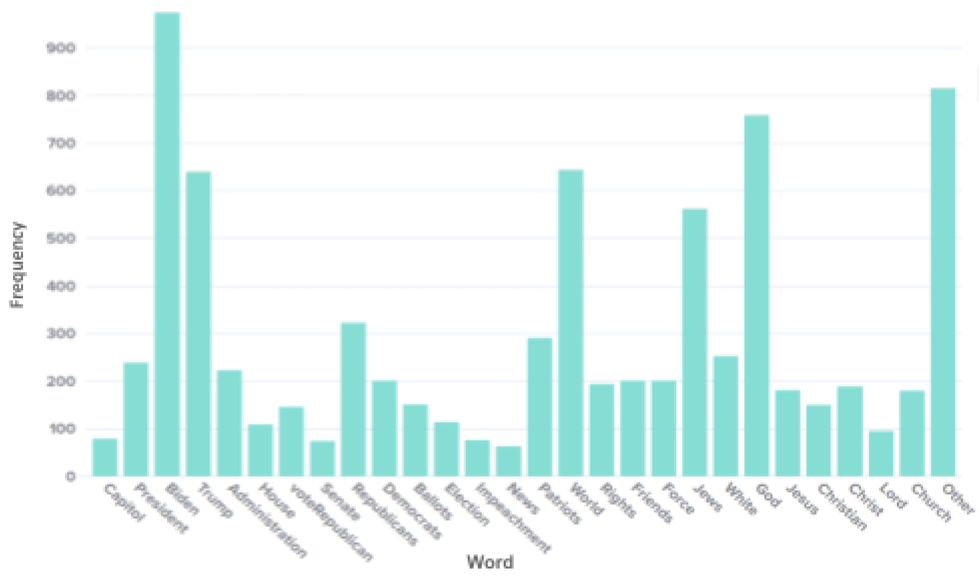


Table 1: Top 10 words used across 8Kun

Word	Frequency
Biden	1196
Trump	823
God	759
World	746
Twitter	667
Jews	562
Patriots	291
White	253
President	239
Administration	223

We compiled the word frequencies for the words used in conjunction with the word "Trump" to understand the context in which former president Donald Trump is discussed (Figure 7). From the data shown within Figure 7, we see that Twitter has a high frequency count which aligns with Trump's notorious use of the platform as a direct line of communication with the public. It is important to note that when looking through the context-words outlining the sentiment for former president Donald Trump, that a high majority (roughly 90%) of the adjectives used to describe him were negative. This is seen through a diverse set of negative adjectives with low frequency counts used in conjunction with his name. Due to the use of many adjectives expressing the same sentiment, no single adjective garners enough frequency to be represented in Figure 7. Programmatically categorizing the various adjectives as positive or negative would provide a numerical expression of sentiment, however, this method of analysis is out of scope.

Figure 7: Sentiment for Trump

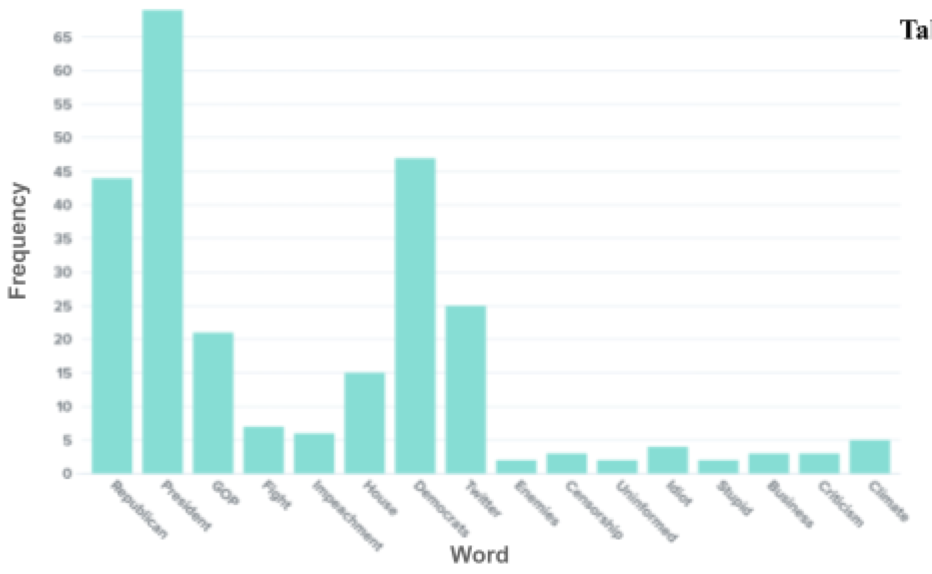


Table 2: Most common words associated with Trump

Word	Count
President	69
Democrats	47
Republican	44
Twitter	25
GOP	21
House	15

In its current state, our crawler and data processing techniques show an approximate representation of the sentiments expressed by 8Kun users. Additionally, our methods serve as a framework to develop further for more accurate results.

4.2 Security and Tor Marketplaces

While Tor provides the scaffolding for anonymous navigation of the internet, maintaining anonymity is ultimately the responsibility of the end user. An obfuscated IP and encrypted traffic alone do not guarantee a user's identity is hidden, especially when users engage in activity that requires the exchange of sensitive information. The trading of goods and/or services for currency is one such activity, however vendor/marketplace hidden services are amongst the most commonly trafficked sites on the Tornet (Owens). We seek to answer how vendors remain anonymous and secure on the Dark Web while exchanging sensitive information.

Many hidden service users browse Tor through Tails OS, a portable linux distribution system built for privacy and security. Tails is installed on a USB device which the user boots from. Tails runs entirely from RAM and does not interact with the user's primary storage devices. Once the user shuts down their machine, all data from that session is erased. A Tails user does not leave a trace of their Tor activity on their machine. (Rhysider, Tails)

Furthermore, marketplaces apply additional user authentication methods. For example, The AlphaBay market used a PGP key validation system where a user would submit their public key at login, receive a challenge phrase to decrypt with their private key, and submit the decrypted message back to AlphaBay to log in. Programs bundled with Tails OS are used for decrypting these messages. This is an anonymous form of two-factor-authentication that does not require the marketplace to have access to any other user information. Because Tails erases all data on shutdown, the user cannot store their private PGP key on their device. Instead, they store a seed phrase externally (written on paper, memorized, etc) which they use to generate their public and private PGP keys as needed. (Rhysider)

4.3 Fraud on the DarkNet

While a user can employ methods to maintain security and anonymity while using Tor marketplaces, one must also consider the trustworthiness of other users and trustworthiness of the marketplace itself. The anonymous nature of HSs combined with the illicit nature of goods on Tor marketplaces create a space where scams and fraud are common. Some HS catalogues list over a thousand scam sites, claiming to sell discounted Amazon gift cards, Bitcoin investment schemes, hackers and hitmen for hire, and drugs. With this prevalence of fraud, it is important that users navigate hidden services with skepticism. (Winzen)

Marketplace operators employ trust rankings and escrow processes to ensure that vendors using their platform are not committing fraud. Much like clearnet marketplaces like Ebay, trustworthy marketplaces display statistics for sellers and allow users to leave reviews of products. For vendors with low trust, marketplaces will enforce an escrow process: Once a buyer purchases an order, the marketplace will hold the buyer's bitcoin until they have received and confirmed the product is legitimate. The seller is only paid out after the buyer has confirmed. Vendors with high trust can obtain finalized entry status which bypasses the escrow process. **(**Rhysider**)**

5. SUMMARY AND CONCLUSIONS

To understand user security on Tor, we discuss the strengths and weaknesses of the Tor protocol. We show that users still have to employ many security tactics on top of the Tor network to maintain privacy and security while performing actions like buying or selling goods on Tor marketplaces. A chain of security is only as strong as its weakest link; the Tor protocol should only be treated as one such link in the chain, not an impenetrable security solution.

To showcase political content on the Tor network, we created Spotlight, a framework for collecting text content and performing a rudimentary sentiment analysis. We use open source .NET libraries to make requests to hidden services by routing our crawler through a Tor proxy. By utilizing HTML parsing libraries, we target text content only posted by humans, opposed to website-generated text from UI elements. We discard words with no intrinsic value and use frequency analysis to count the most common descriptors associated with a given political word. We focus our crawling on the Tor forum 8Kun due to its large user base and uniform UI arrangement. Our sentiment analysis shows that on 8Kun, mentions of the word "Trump" are associated with primarily negative sentiments despite the established right-leaning user base of the forum.

As discussed above, many hidden services employ robustness measures to prevent scripted access. Our crawler cannot bypass these measures and only retrieves text content from hidden services which are accessible upon first request, with no captcha or redirect. Implementing methods to bypass these robustness measures are either out of scope for the project, or impossible without direct human intervention.

We can take multiple avenues to improve our crawler in future works. To improve scalability and efficiency, we can utilize AbotX's parallelization engine which allows Abot to recursively crawl multiple pages at once. While our crawler only supports the Tor protocol, there are many other Dark Web protocols with their own hidden services. With the ability to crawl multiple protocols, we would gain the ability to compare content across networks. Another avenue of improvement would involve implementing an NLP library for more rigorous sentiment analysis. Implementing such a library would allow us to produce numerical values for sentiment (e.g. a positivity / negativity score), allowing for a more rigorous comparison of words across contexts.

6. APPENDIX: SOFTWARE DESCRIPTION

Spotlight is an open-source .NET console application developed with Microsoft's .NET standard 5.0, and is written in C#. The program is meant to be run from any command line tool, and requires the use of specifically defined input commands. Spotlight is the culmination of multiple open-source libraries to simplify the process of crawling and scraping text content from the Dark Web. Amongst these libraries includes: Abot, TorSharp, HtmlAgilityPack, MongoDB, CommandLineParser, and Serilog.

In order to run the application, Spotlight requires the use of a Windows machine (specifically Windows 10), as well as the latest .NET 5.0+ SDK. Once these requirements are met, one may simply clone the codebase from the publicly available GitHub repository to any desired (secure) location on their computer. From any command line tool navigate into the "/webcrawler" folder inside the repository and run the following dotnet commands in order:

1. "dotnet restore"
2. "dotnet build"

This will create a working executable of the project inside of "/bin/Debug/net5.0/". The executable will be named "webcrawler.exe". This executable should be run from a command line tool or program as in order for the crawler to execute correctly the user must specify specific commands for the crawler to interpret.

Commands:

```
 -s, --single    Crawl a single URL. Specify the URL.

 -m, --multi     Crawl multiple URLs. Pass input file containing URLs.

 -h, --handler   (Group: Page Handler) Specify page handler type:
             * wordFrequency
             * sentimentAnalysis

 --help          Display this help screen.

 --version       Display version information.
```

Please refer to the publicly available GitHub repository for more information about Spotlight.

https://github.com/alexmarozick/DarkWebCrawler

REFERENCES

Carnegie Mellon University CyLab, “The reCAPTCHA Project,” *The reCAPTCHA Project - Carnegie Mellon University CyLab*. [Online]. Available: https://web.archive.org/web/20171027203659/https://www.cylab.cmu.edu/partners/success-stories/recaptcha.html. [Accessed: 14-Mar-2021].

Computerphile, YouTube, 31-May-2017. [Online]. Available: https://www.youtube.com/watch?v=QRYzre4bf7I. [Accessed: 11-Mar-2021].

Dark Net Daily, “All v3 Onion Addresses Down After Attack On The Tor Network,” Dark Net Daily, 27-Jan-2021. [Online]. Available: https://darknetdaily.com/?p=1030. [Accessed: 14-Mar-2021].

“Leave no trace on the computer,” Tails. [Online]. Available: https://tails.boum.org/about/index.en.html. [Accessed: 11-Mar-2021].

MentalOutlawStudios, YouTube, 12-Jan-2021. [Online]. Available: https://www.youtube.com/watch?v=4L8Svvdxn08. [Accessed: 14-Mar-2021].

G. Owens and N. Savage, “The Tor Dark Net,” *Global Commission on Internet Governance*, vol. 20, 2015

J. M. Applebaum, “A Model of Outbound Client Traffic on The Tor Anonymity Network.”

J. Rhysider, The Vendor – Darknet Diaries. [Online]. Available: https://darknetdiaries.com/transcript/81/. [Accessed: 11-Mar-2021].

K. Sangeetha and K. Ravikumar, “Defense Against Protocol Level Attack in Tor Network using Deficit Round Robin Queuing Process,” Egyptian Informatics Journal, 03-Apr-2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866517301457. [Accessed: 11-Mar-2021].

R. Dingledine and N. Mathewson, “ Tor Path Specification,” path-spec.txt - torspec - Tor's protocol specifications. [Online]. Available: https://gitweb.torproject.org/torspec.git/tree/path-spec.txt. [Accessed: 15-Mar-2021].

“The Tor Project: Privacy & Freedom Online,” *Tor Project*. [Online]. Available: https://www.torproject.org/about/history/. [Accessed: 12-Mar-2021].

“Tor Project: Relay Operations,” Tor Project | Relay Operations. [Online]. Available: https://community.torproject.org/relay. [Accessed: 11-Mar-2021].

D. Winzen, Onion link list. [Online]. Available: https://onions.danwin1210.me/onions.php. [Accessed: 11-Mar-2021].

A. T. Zulkarnine, R. Frank, B. Monk, J. Mitchell, and G. Davies, “Surfacing collaborated networks in dark web to find illicit and criminal content,” *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, 2016.