Dark web Forensic investigation and mitigation of the data using ML algorithm

Problem statement:

The unauthorized distribution of sensitive information, including personal information of an individual, organizational data, and other critical identifiers, has become increasingly prevalent on the dark web. Often, this data is traded without the knowledge or consent of the affected individuals or entities, leading to significant privacy concerns. The exposure of Personally Identifiable Information (PII) to unauthorized parties poses substantial risks, including the potential misuse of such data by larger organizations. This situation underscores the urgent need to address the privacy and security implications associated with the illegal distribution of sensitive information.

Breakdown:

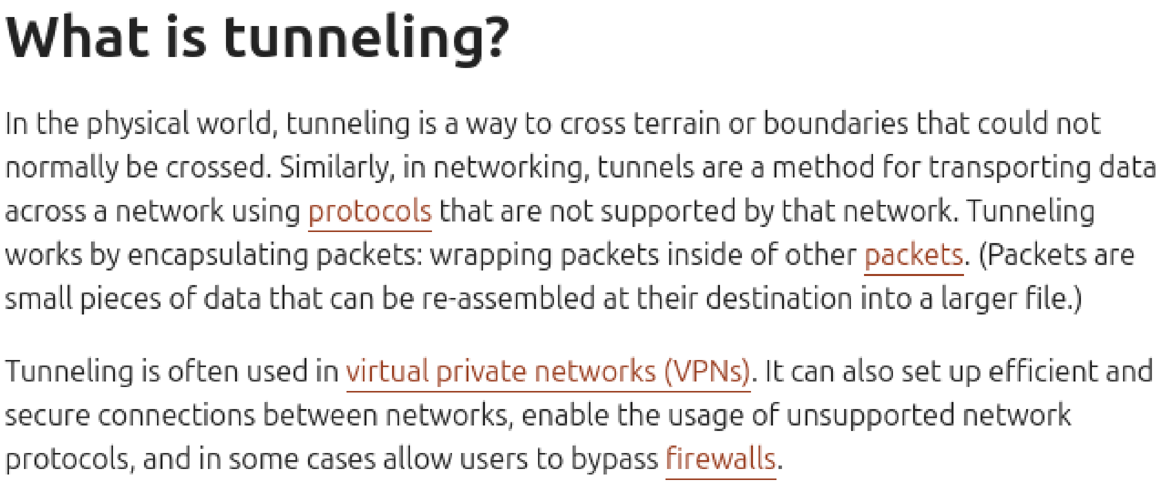
1. **Tunnelling**: Creating a new network tunnel only accessible for us.
2. **Dark Web Crawling**: Conduct a comprehensive search across the dark web to identify any instances where the gathered data may be present.
3. **Data Verification**: Analyze the collected data to determine its legitimacy and relevance.
4. **IP Address Deanonymization:** If the data is verified as legitimate, proceed with the deanonymization of the IP address associated with the data owner.
5. **Data Removal**: Take necessary steps to remove the unauthorized data from the dark web.
6. **Final Verification**: Confirm the successful removal of the data and ensure that the information is no longer accessible on the dark web.

Initiation

Create a new Tunnel to the Dark web which is private to me.

Here will take the Entity name as input from the user and search for the relevent legit data from the crawled data over the entire Dark web.

https://www.cloudflare.com/en-gb/learning/network-layer/what-is-tunneling/



What is tunneling?

In the physical world, tunneling is a way to cross terrain or boundaries that could not normally be crossed. Similarly, in networking, tunnels are a method for transporting data across a network using protocols that are not supported by that network. Tunneling works by encapsulating packets: wrapping packets inside of other packets. (Packets are small pieces of data that can be re-assembled at their destination into a larger file.)

Tunneling is often used in virtual private networks (VPNs). It can also set up efficient and secure connections between networks, enable the usage of unsupported network protocols, and in some cases allow users to bypass firewalls.

Other resources :

https://www.youtube.com/watch?v=32KKwgF67Ho

Goal : to create a tunnel to the Dark web

crawling

paper-1 :: CRATOR: a Dark Web Crawler {C-1}

Abstract : This crawler handles the security checks like captchas efficiently. The approach of the paper drills down to use seed URL's list primarlily, link analysis, and scanning to discover new content, and also incorporate methods for user-agent rotation and proxy usage to maintain anonymity and avoid detection.

Previous work mentoined :

- Tor crawler managed to scrape 251 pages in 20 minutes and eight minutes to scrape The Guardian’s clear website of 223 pages with an average of circa 6.4 pages.
- There is an open-source Tool **TorBot (python-based tool)** for the dark web that enables users to gather and analyze information from hidden websites and services. it effectively Collects information such as website content, metadata, and links.
- Features involved in TorBot :

    1. keyword-based searching
    2. content analysis
    3. automated data extraction

    ```
    https://github.com/DedSecInside/TorBot
    ```

- The second open-source tool **ACHE (Java-based tool)**, an adaptive crawler designed to identify entry points to hidden web resources. The proposed crawler adapts to the structure of the hidden web by leveraging machine learning techniques to identify entry points and improve its crawling efficiency automatically.
- Key feature :

    The algorithm uses a combination of content-based and link-based analysis to identify hidden web entry points. It continuously updates its search strategy based on the results of previous crawls.

    ```
    https://github.com/VIDA-NYU/ache
    ```

- The third open-source tool **Kalpakis,** explores an interactive search engine for Home Made Explosives recipes on the Surface and Dark Web.
- Key feature :

    The hybrid architecture combines a dedicated crawler and domain-specific queries, adapting the widely used web crawler, **Apache Nutch**.

https://github.com/apache/nutch

- The Fourth open-source tool **Celestini,** presents a flexible toolkit for structure and content mining on the Web, including the Tor dark Web. They have a customised crawler **bUibiNG**
- Key feature :

  The toolkit incorporates web crawling, extraction, indexing, and mining modules.

Limitations with the above crawlers :

Limitations encountered with the availability and suitability of certain crawlers. The TorBot crawler was excluded from our analysis due to outdated external de- pendencies that rendered it unusable. Additionally, the custom crawlers developed by Kalpakis and Celestini based on Apache Nutch and bUbiNG, respectively, were not publicly released, making their replication challenging. Furthermore, both the crawlers Nutch and bUbiNG were not specifically designed to work with the dark web.

Design of CRATOR :

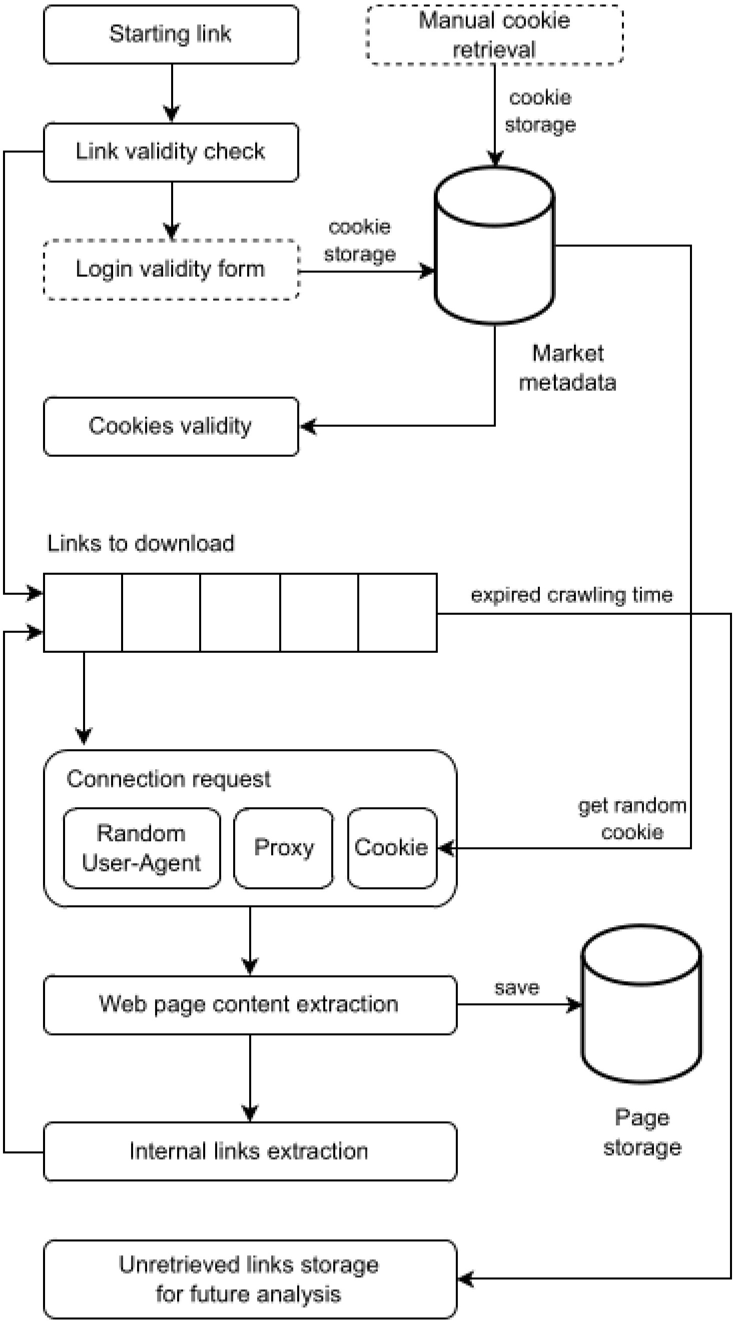


Fig. 1: Architecture.

Methodology

1. Firstly, the crawler receives a list of potential links, namely starting links, which it can obtain from public dark web directories or from the user directly, Next, the crawler performs a validity check to ensure that one of these links is accessible.If there is a valid link, it becomes the starting point of the crawling process.

2. If the website has Login page and captcha check for the bot, cookies must be provided for those to access a valid link.The architecture offers two methods for providing cookies to the crawler :
   - The first method involves human intervention, where the user manually writes one or more cookies for the crawler to use during the crawling process.
   - The second method involves the implementation of a login validity script, which tries to automatically log in and bypass captcha on a dark web link by obtaining login credentials from a Market metadata file.
   - In addition, the architecture incorporates a layer for verifying the validity of cookies, in which all collected cookies undergo validation by inspecting for any unexpected redirects.
3. After the verification of link and cookie validity, the crawling process starts by downloading the first link along with all internal links, by following the **Breadth First approach**.

   Example : links sharing the same domain as the initial link, such as https://en.wikipedia.org/wiki/IOT and https://en.wikipedia.org/wiki/Crawler, both having the domain en.wikipedia.org.

   NOTE : The connection module makes use of proxy settings to establish a connection with onion links, cookies to bypass security checks and random user-agent to avoid being identified as a bot.

4. After extracting the URL, we go for a link validity Check. URL reachability check involves sending a request to the URL or onion link using the HTTP or HTTPS protocol.based on the response code recieved we proceed accordingly.
   - If the crawler determines that a URL is not reachable, it marks the URL as broken or invalid, and remove it from the crawling queue. This helps to prevent unnecessary requests to unreachable URLs, which can slow down the crawling process and waste system resources.
   - In addition, the crawler may also check if the URL is aduplicate or a mirror of another page, and avoid crawling such pages to prevent

duplication of content.

5. To perform an automatic login, a web crawler needs to submit login credentials to the login page of the website. This can be done using HTTP POST requests with the appropriate form data, such as the username and password.
   To avoid being detected as a crawler, it is important to rotate the cookies periodically. This can be done by logging out and logging back in again to obtain a new cookie, or by using multiple cookies chosen with a fisher-yates shuffle algorithm.

6. To detect captchas, web crawlers need to analyze the page structure and content, and look for patterns that indicate the presence of a captcha. Here there may be human intervention is needed.

7. Python scripts that are designed to make connection requests on the dark web must be configured to use Tor Hidden Services to resolve domain names (Which are not similar to the regular domain names)and connect to the appropriate servers.

8. they have this stop criteria based on the factors like :
   - max depth
   - max no:of links downloaded
   - Time limit
   - Data crawled

External References :

https://youtu.be/m_3gjHGxIJc?si=soThSOrXVTuBbZ0x

Data extraction

To extract the data from the website we use web scraping

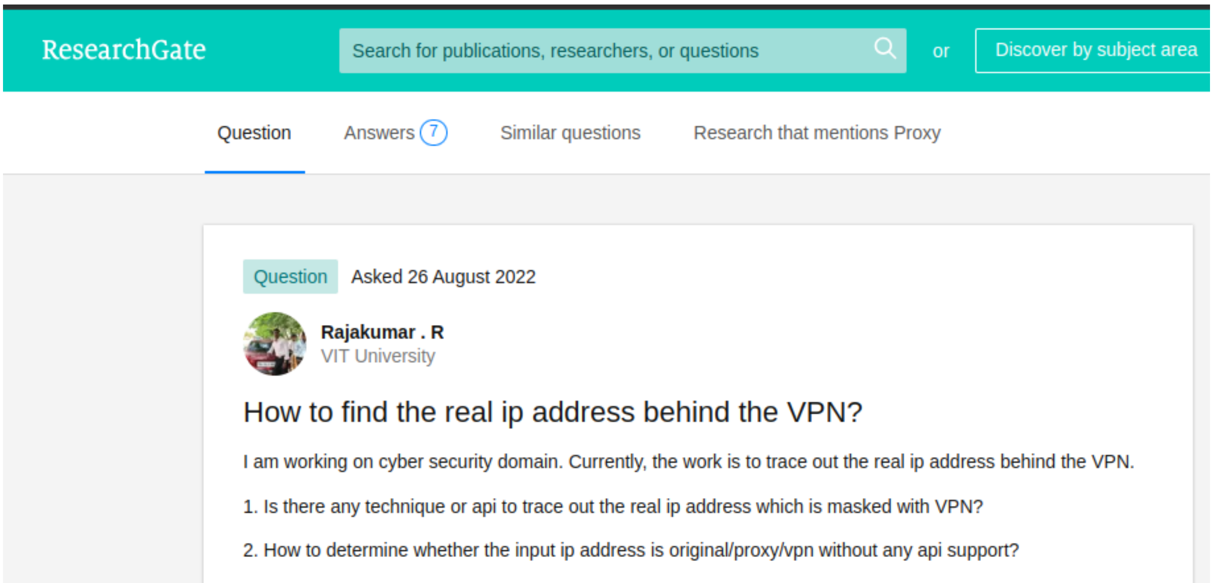
we are going to use Beautifulsoup4 module from bs4 library to scrape the data from python and lxml parser and request library from python.

https://www.youtube.com/watch?v=ng2o98k983k&t=681s

Helps in scraping out the data and converts it into a csv file.

1. what data need to be scraped?
2. what sites need to be crawled?
3. how do you gonna organise the data and feed it to model?
4. which ML model supports this type of feeding and searching over large data?

finding the owner



Asked 26 August 2022

Rajakumar . R
VIT University

How to find the real ip address behind the VPN?

I am working on cyber security domain. Currently, the work is to trace out the real ip address behind the VPN.

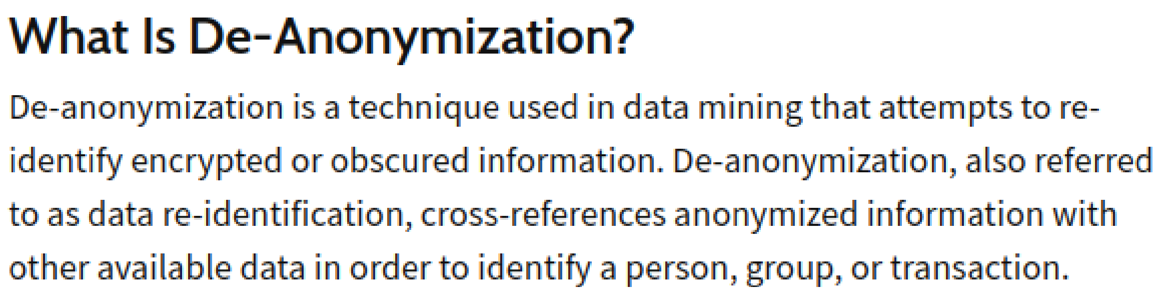
1. Is there any technique or api to trace out the real ip address which is masked with VPN?

2. How to determine whether the input ip address is original/proxy/vpn without any api support?

> Tracing a VPN user's IP address is technically and ethically difficult. VPNs hide IP addresses to protect user privacy. Tracing an actual IP address behind a VPN without violating ethics or law is difficult. Details about your questions:
>
> 1. Finding Real IP Behind VPN:
>
> Some legal considerations: Legal implications must come first. Tracing a VPN's IP address may breach privacy regulations and VPN provider agreements.
>
> The technical aspects: Without VPN assistance, tracing the real IP address is practically difficult. VPNs encrypt traffic, making IP addresses hard to identify. Complex, resource-intensive, and unreliable methods include timing attack and traffic correlation analysis.
>
> Cooperation with Authorities: Law enforcement can legally request VPN user data from providers in cases of illicit activity. Legal processes rigorously govern this.
>
> #2. Identifying VPN/Proxy IP Addresses:
>
> Compare IP Address to VPN/Proxy Lists: VPN and proxy IP ranges are listed in databases. Checking an IP address against these lists can reveal a VPN/proxy IP.
>
> Traffic Pattern Analysis: High encrypted traffic volumes may imply VPN/proxy use, however this is not necessarily true.
>
> Analysis of Ports and Protocols VPNs use certain ports and protocols more. Network traffic analysis can reveal them.
>
> - Behaviour Analysis: Geographic inconsistency or fast IP changes may suggest VPN/proxy use.
>
> References from research
>
> - "Network Traffic Analysis for VPN Detection" discusses VPN usage detection methods.
>
> "Privacy and Anonymity in the Internet Era" discusses VPNs and other privacy and anonymity technologies.
>
> Ethical and legal issues:
>
> Respecting user privacy is crucial. Any de-anonymization attempt must be legal and justified.
>
> Legal Compliance: Comply with local and international regulations when tracing IPs or identifying VPN usage.

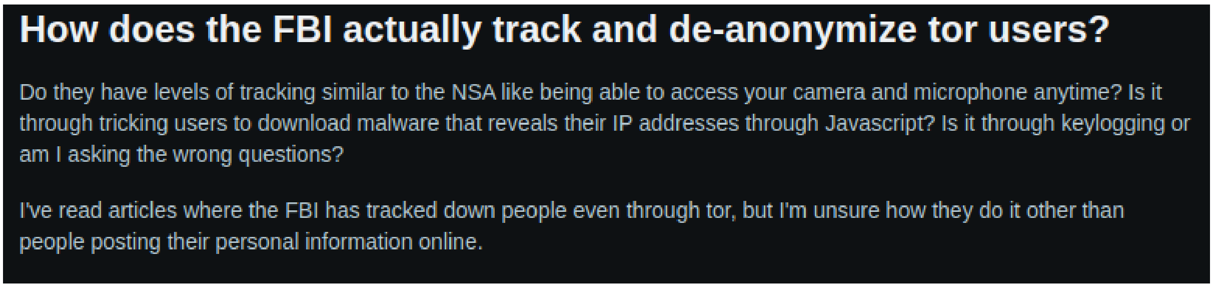
Deanonymisation

Here our main aim is to de-anonymise the owner who inputs the data to that particular website.



What Is De-Anonymization?

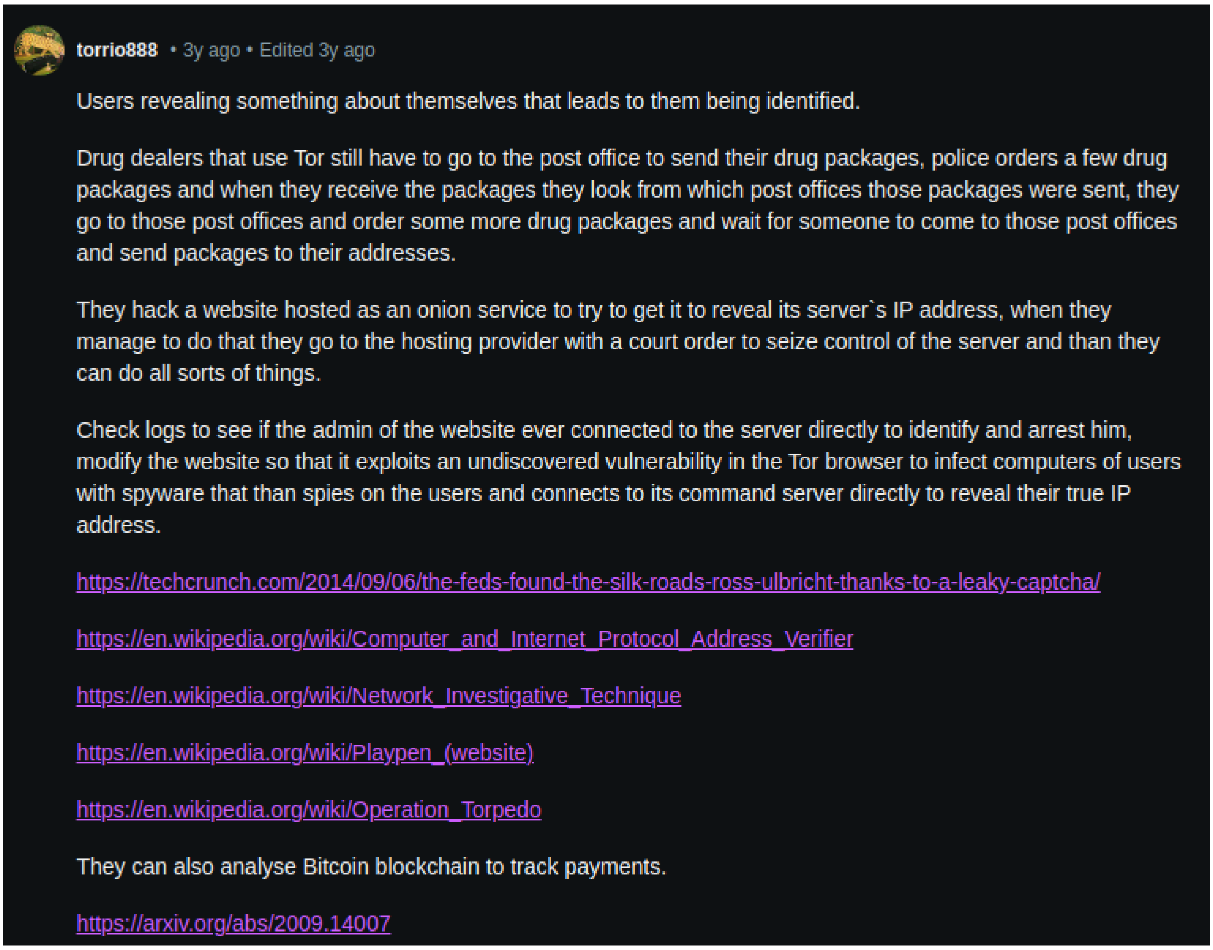
De-anonymization is a technique used in data mining that attempts to re-identify encrypted or obscured information. De-anonymization, also referred to as data re-identification, cross-references anonymized information with other available data in order to identify a person, group, or transaction.



How does the FBI actually track and de-anonymize tor users?

Do they have levels of tracking similar to the NSA like being able to access your camera and microphone anytime? Is it through tricking users to download malware that reveals their IP addresses through Javascript? Is it through keylogging or am I asking the wrong questions?

I've read articles where the FBI has tracked down people even through tor, but I'm unsure how they do it other than people posting their personal information online.



torrio888 • 3y ago • Edited 3y ago

Users revealing something about themselves that leads to them being identified.

Drug dealers that use Tor still have to go to the post office to send their drug packages, police orders a few drug packages and when they receive the packages they look from which post offices those packages were sent, they go to those post offices and order some more drug packages and wait for someone to come to those post offices and send packages to their addresses.

They hack a website hosted as an onion service to try to get it to reveal its server`s IP address, when they manage to do that they go to the hosting provider with a court order to seize control of the server and than they can do all sorts of things.

Check logs to see if the admin of the website ever connected to the server directly to identify and arrest him, modify the website so that it exploits an undiscovered vulnerability in the Tor browser to infect computers of users with spyware that than spies on the users and connects to its command server directly to reveal their true IP address.

https://techcrunch.com/2014/09/06/the-feds-found-the-silk-roads-ross-ulbricht-thanks-to-a-leaky-captcha/

https://en.wikipedia.org/wiki/Computer_and_Internet_Protocol_Address_Verifier

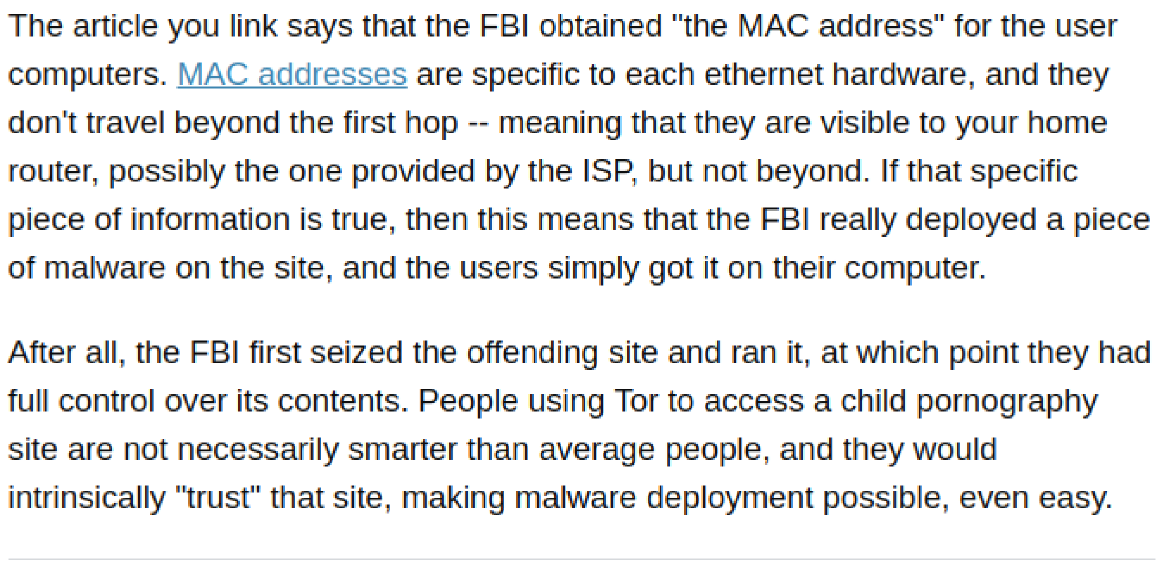
https://en.wikipedia.org/wiki/Network_Investigative_Technique

https://en.wikipedia.org/wiki/Playpen_(website)

https://en.wikipedia.org/wiki/Operation_Torpedo

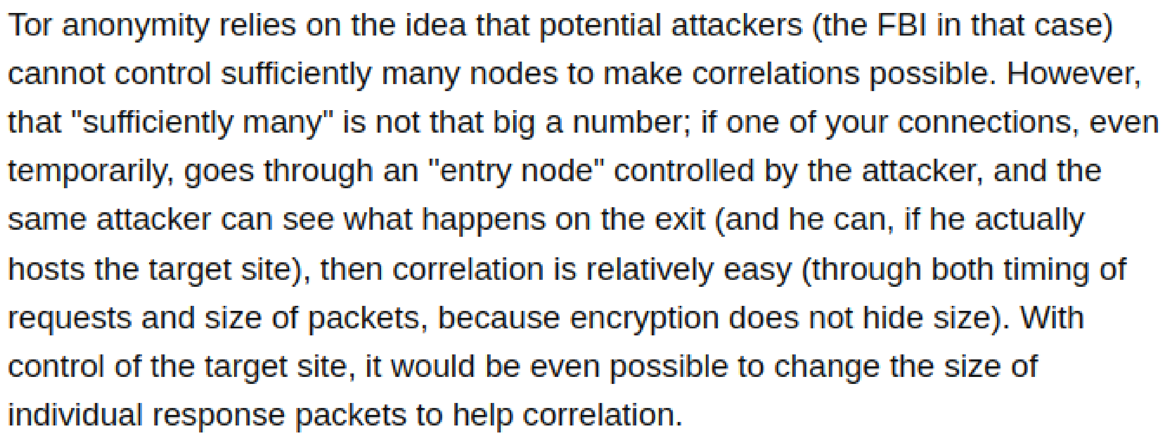
They can also analyse Bitcoin blockchain to track payments.

https://arxiv.org/abs/2009.14007



The article you link says that the FBI obtained "the MAC address" for the user computers. MAC addresses are specific to each ethernet hardware, and they don't travel beyond the first hop -- meaning that they are visible to your home router, possibly the one provided by the ISP, but not beyond. If that specific piece of information is true, then this means that the FBI really deployed a piece of malware on the site, and the users simply got it on their computer.

After all, the FBI first seized the offending site and ran it, at which point they had full control over its contents. People using Tor to access a child pornography site are not necessarily smarter than average people, and they would intrinsically "trust" that site, making malware deployment possible, even easy.



Tor anonymity relies on the idea that potential attackers (the FBI in that case) cannot control sufficiently many nodes to make correlations possible. However, that "sufficiently many" is not that big a number; if one of your connections, even temporarily, goes through an "entry node" controlled by the attacker, and the same attacker can see what happens on the exit (and he can, if he actually hosts the target site), then correlation is relatively easy (through both timing of requests and size of packets, because encryption does not hide size). With control of the target site, it would be even possible to change the size of individual response packets to help correlation.

First we need to determine whether the accessed ip address is a vpn or not, for that thing as the above

what is the difference between Proxy and vpn service?

A proxy server provides traffic source anonymization. It may also support traffic distribution, or potentially scan and check network data packets against predetermined security policies.
In contrast, a
VPN uses encryption to mask both the IP address and data so it’s unreadable by unauthorized users.

https://aws.amazon.com/compare/the-difference-between-proxy-and-vpn/#:~:text=A proxy server provides traffic,it's unreadable by unauthorized users.

paper-2 :: Detecting VPN Traffic through Encapsulated TCP Behavior

- They claim that their final product has 0.11% false positive rate which works efficient than existing machine learning models.

Base idea : By identifying UDP connections that exhibit strong TCP-like behavior, we can infer the presence of tunneled TCP traffic that is likely to be indicative of VPN usage

Reason for obtaining this method : many popular VPN services (e.g., ExpressVPN, NordVPN, Surfshark, and Private Internet Access) use UDP-based protocols because the outer tunnel does not need to provide any TCP features.

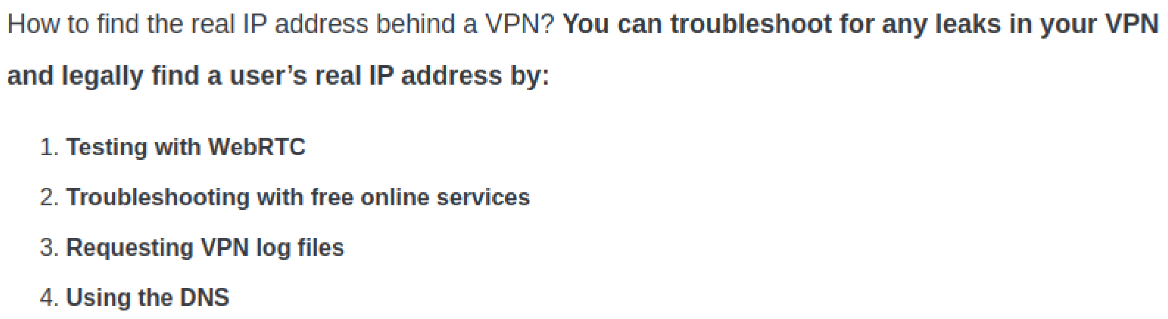
Methodology

They choose these characteristics as the base one for checking the TCP packets hiding under UDP.

- 3WHS: The presence of a three-way SYN, SYN-ACK, ACK handshake to open the connection (RFC 9293, Section 3.5).
- 500msACK: The presence of an ACK packet generated with- in 500 ms of the arrival of a data segment (RFC 9293, Section 3.8.6.3).
- 2RMSS: The presence of an ACK packet generated after the receival of 2×RMSS bytes of data, where RMSS is the maximum segment size (MSS) specified by the TCP endpoint receiving the segments (RFC 9293, Section 3.8.6.3).

Other Resources : https://fingerprint.com/blog/vpn-detection-how-it-works/#other-vpn-detection-techniques

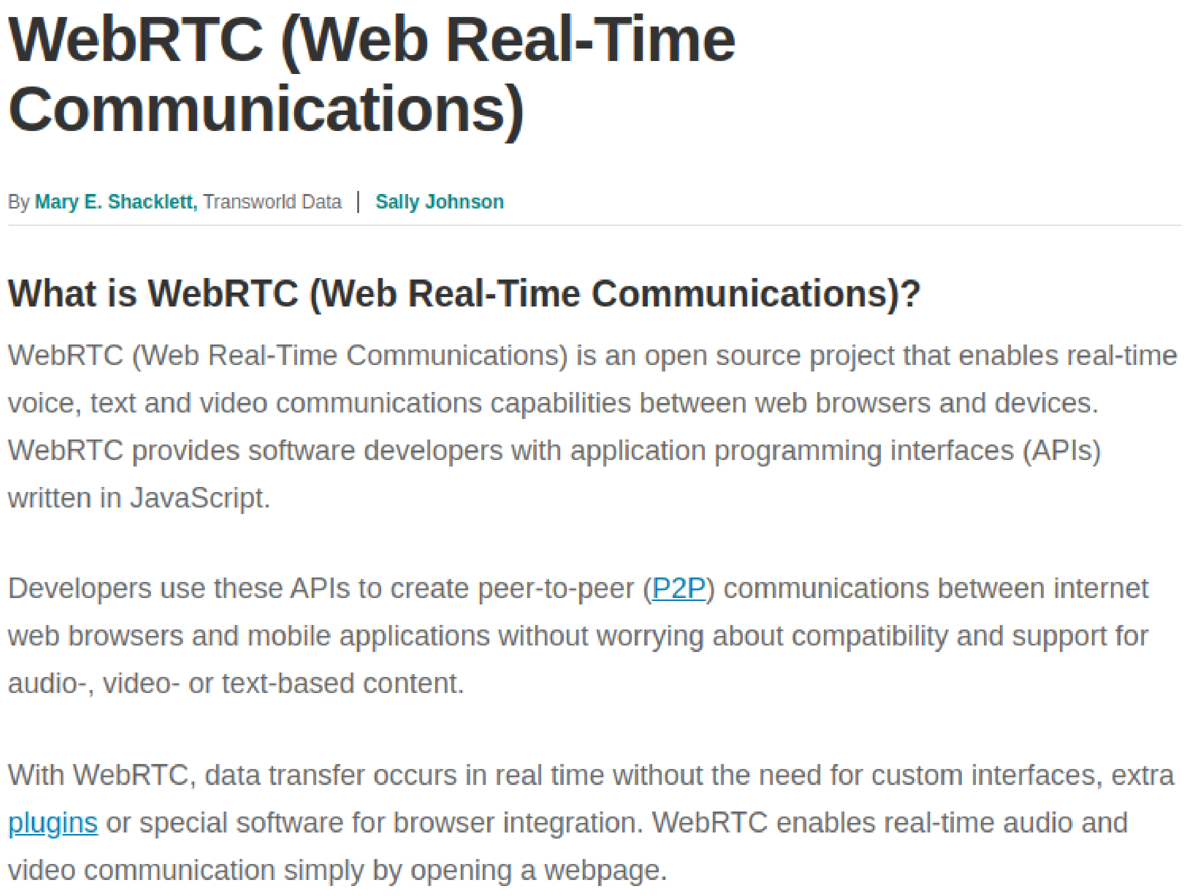
Second step after determining whether it is a proxy or vpn



How to find the real IP address behind a VPN? **You can troubleshoot for any leaks in your VPN and legally find a user's real IP address by:**

1. **Testing with WebRTC**
2. **Troubleshooting with free online services**
3. **Requesting VPN log files**
4. **Using the DNS**

WebRTC



WebRTC (Web Real-Time Communications)

By **Mary E. Shacklett**, Transworld Data | **Sally Johnson**

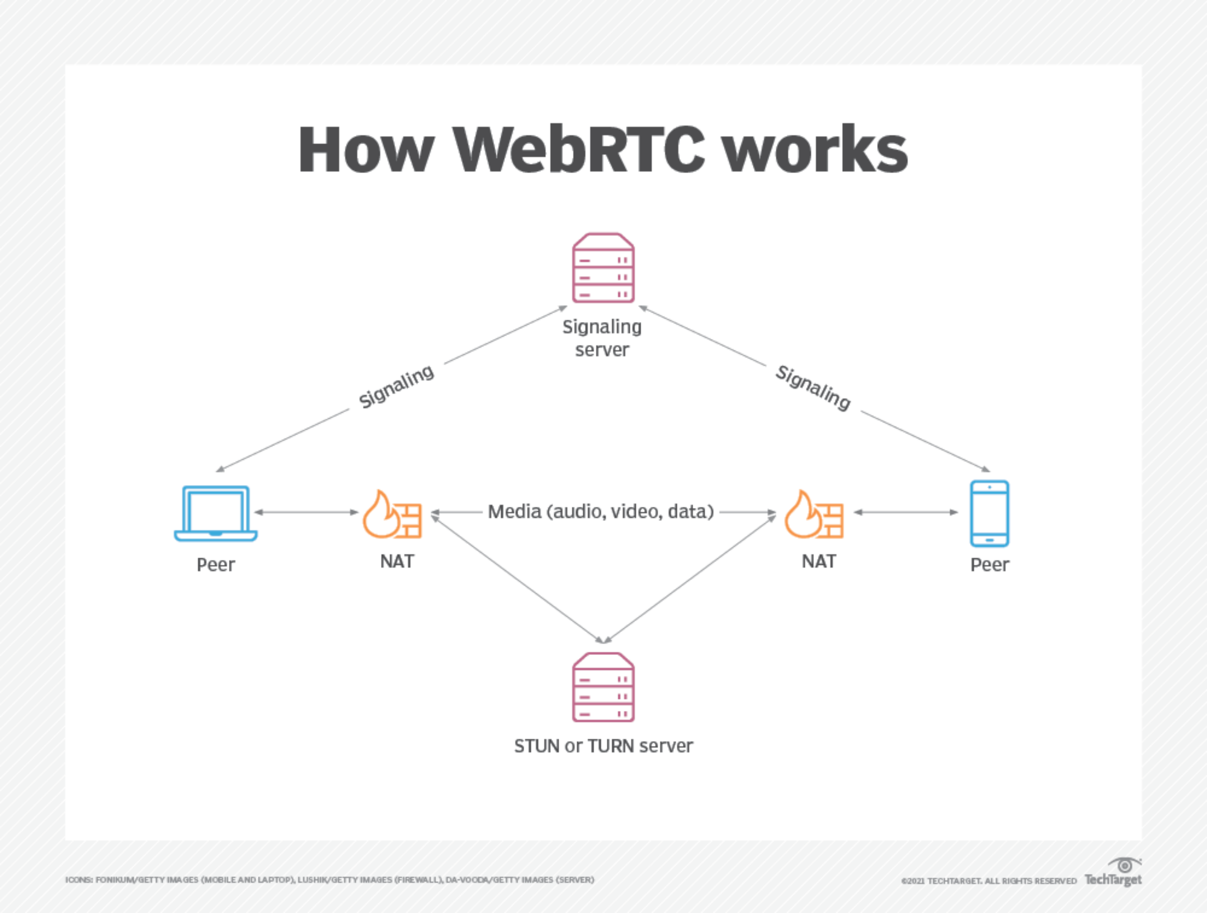
What is WebRTC (Web Real-Time Communications)?

WebRTC (Web Real-Time Communications) is an open source project that enables real-time voice, text and video communications capabilities between web browsers and devices. WebRTC provides software developers with application programming interfaces (APIs) written in JavaScript.

Developers use these APIs to create peer-to-peer (P2P) communications between internet web browsers and mobile applications without worrying about compatibility and support for audio-, video- or text-based content.

With WebRTC, data transfer occurs in real time without the need for custom interfaces, extra plugins or special software for browser integration. WebRTC enables real-time audio and video communication simply by opening a webpage.

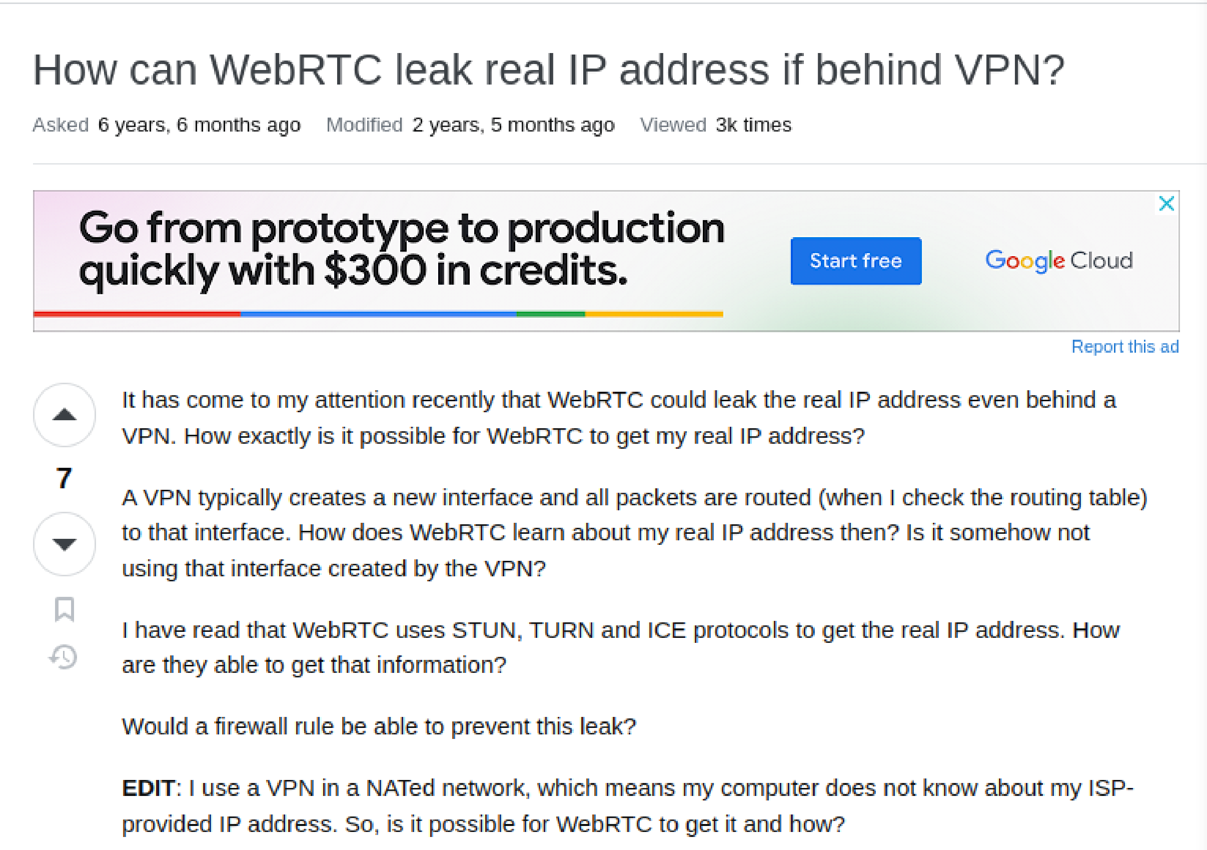
Working of WebRTC :


How WebRTC works
Signaling server
Signaling
Signaling
Peer
NAT
Media (audio, video, data)
NAT
Peer
STUN or TURN server
ICONS: FONIKUM/GETTY IMAGES (MOBILE AND LAPTOP), LUSHIK/GETTY IMAGES (FIREWALL), DA-VOODA/GETTY IMAGES (SERVER)
©2021 TECHTARGET. ALL RIGHTS RESERVED
TechTarget

STUN (Session Traversal Utilities for NAT)
TURN (Traversal Using Relays around NAT)

- WebRTC uses JavaScript, APIs and Hypertext Markup Language to embed communications technologies within web browsers. It is designed to make audio, video and data communication between browsers user-friendly and easy to implement. WebRTC works with most major web browsers.

  In most cases, WebRTC connects users by transferring real-time audio, video and data from device to device using P2P communications.



How can WebRTC leak real IP address if behind VPN?

Asked 6 years, 6 months ago Modified 2 years, 5 months ago Viewed 3k times

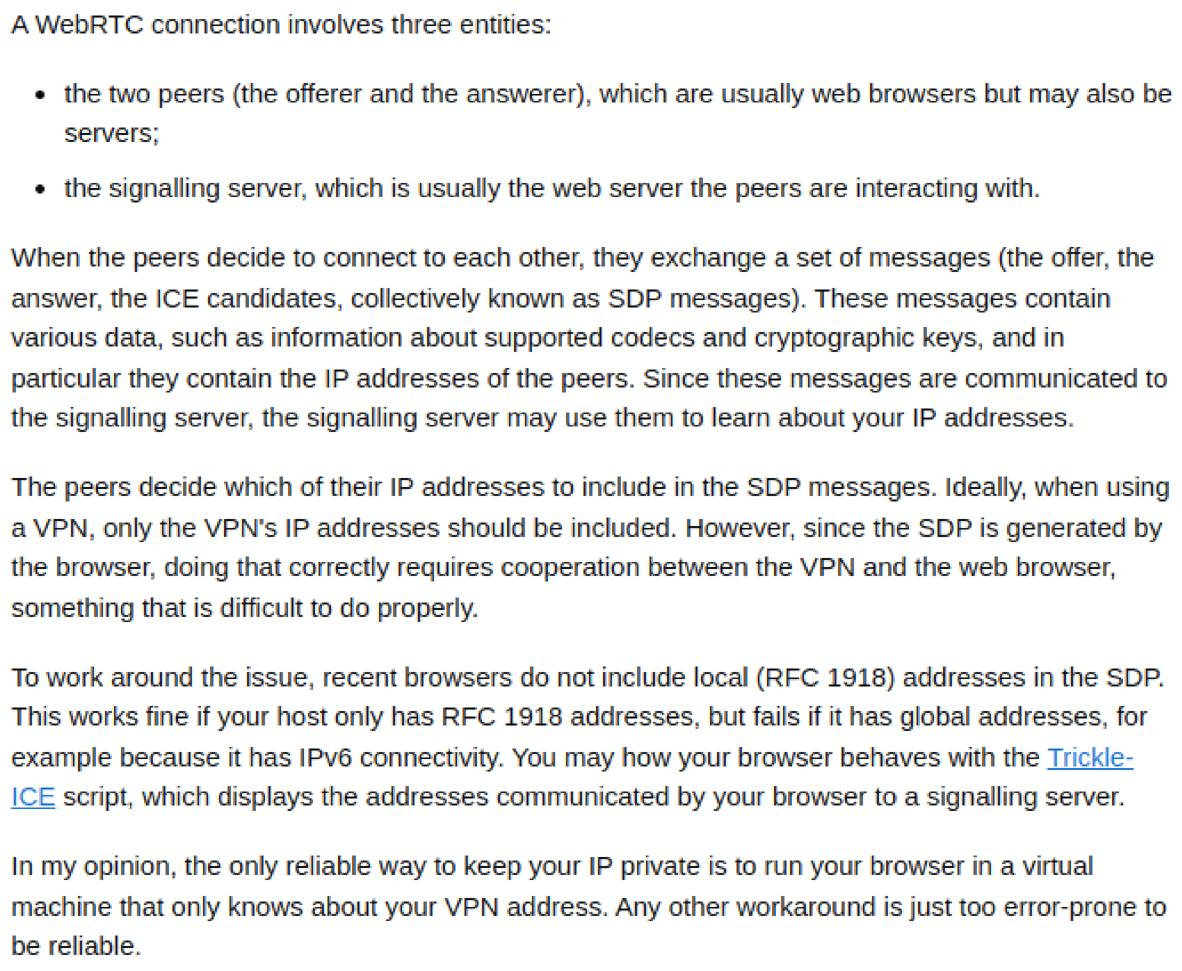
It has come to my attention recently that WebRTC could leak the real IP address even behind a VPN. How exactly is it possible for WebRTC to get my real IP address?

A VPN typically creates a new interface and all packets are routed (when I check the routing table) to that interface. How does WebRTC learn about my real IP address then? Is it somehow not using that interface created by the VPN?

I have read that WebRTC uses STUN, TURN and ICE protocols to get the real IP address. How are they able to get that information?

Would a firewall rule be able to prevent this leak?

EDIT: I use a VPN in a NATed network, which means my computer does not know about my ISP-provided IP address. So, is it possible for WebRTC to get it and how?



A WebRTC connection involves three entities:

- the two peers (the offerer and the answerer), which are usually web browsers but may also be servers;
- the signalling server, which is usually the web server the peers are interacting with.

When the peers decide to connect to each other, they exchange a set of messages (the offer, the answer, the ICE candidates, collectively known as SDP messages). These messages contain various data, such as information about supported codecs and cryptographic keys, and in particular they contain the IP addresses of the peers. Since these messages are communicated to the signalling server, the signalling server may use them to learn about your IP addresses.

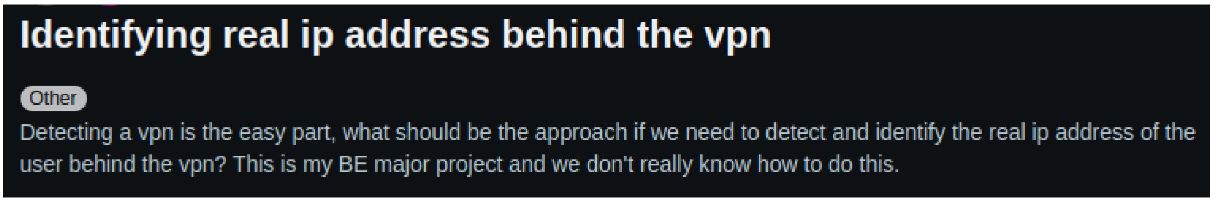
The peers decide which of their IP addresses to include in the SDP messages. Ideally, when using a VPN, only the VPN's IP addresses should be included. However, since the SDP is generated by the browser, doing that correctly requires cooperation between the VPN and the web browser, something that is difficult to do properly.

To work around the issue, recent browsers do not include local (RFC 1918) addresses in the SDP. This works fine if your host only has RFC 1918 addresses, but fails if it has global addresses, for example because it has IPv6 connectivity. You may how your browser behaves with the Trickle-ICE script, which displays the addresses communicated by your browser to a signalling server.

In my opinion, the only reliable way to keep your IP private is to run your browser in a virtual machine that only knows about your VPN address. Any other workaround is just too error-prone to be reliable.

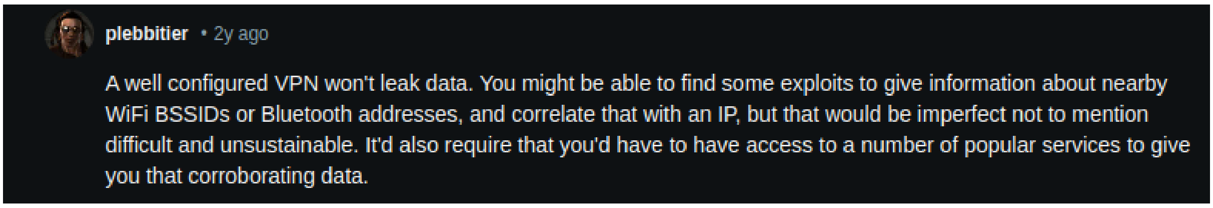
Database validation

- IP address databases are a cornerstone in the arsenal of methods for detecting VPN use in browsers. These databases contain information about IP addresses, including their affiliation with known VPN or proxy services. Cross-matching a user's IP address with these databases can determine whether the user is associated with a VPN or Proxy.



Identifying real ip address behind the vpn

Detecting a vpn is the easy part, what should be the approach if we need to detect and identify the real ip address of the user behind the vpn? This is my BE major project and we don't really know how to do this.



plebbitier • 2y ago

A well configured VPN won't leak data. You might be able to find some exploits to give information about nearby WiFi BSSIDs or Bluetooth addresses, and correlate that with an IP, but that would be imperfect not to mention difficult and unsustainable. It'd also require that you'd have to have access to a number of popular services to give you that corroborating data.

removal of the data

verification