De-anonymizing of entities on the onion sites operating on TOR Network

Mr. Praveen Pawaskar[1], Shreyas Gowda S[2], Nishant Satish Naik[3], S Surya Narayanan[4], Kavya Jaishree[5], Rahul K[6]

[1]Assistant Professor Presidency University, Bengaluru, Karnataka, India

[2,3,4,5,6] Department of CSE, SoE, Presidency University, Bengaluru, Karnataka, India

Abstract— **Onion services and Tor facilitates anonymous communication, giving users and service operators added privacy. Nevertheless, the problems of Tor anonymity impact cyber security and law enforcement, especially when such services are used for illegal purposes. In this paper, we propose a two-stage API-based approach for the de-anonymizing of actors on onion sites. Initially, we utilized an IP intelligence API to find possible IP addresses that could be linked to specific onion services. Next, these IP addresses were checked against a threat intelligence database through another API to confirm that these IP addresses were flagged on blacklists associated with spam, malware, and cybercrime. This allowed for the linkage of concealed traffic data with recognizable and potentially hazardous entities. Our results demonstrate that in the Tor ecosystem, critical information can be exposed due to misconfiguration, exit node behavior, and dependency on external services. We discuss the ethical ramifications of this work while providing principles intended for responsible de-anonymizing action conducted on non-legitimate domains of anonymity. This work highlights the enduring controversy over the extent of digital privacy and security.**

I. INTRODUCTION

The Tor network, originally intended for privacy and freedom of communication, has now become an essential infrastructure for those seeking online anonymity. Its onion routing protocol hides the locations of its users and their usage behavior by encrypting the traffic and routing it through a series of nodes operated by volunteers. Beyond the anonymity of users, Tor also facilitates onion services—hidden websites that cannot be accessed through regular browsers and search engines. These capabilities are vital for protecting whistleblowers, journalists, and citizens in oppressive regimes, but they equally appeal to threat actors' use case for illicit activities including drug trafficking, cybercrime, and the distribution of the illicit ones.

This paper explores a novel approach to de-anonymizing entities operating onion services by leveraging two separate application programming interfaces (APIs). One API provides passive and active methods of gathering data pertaining to real-world IP addresses that may be linked to the Tor environment. The other API examines existing blacklists of spam, botnets, phishing, and various other cyber threats for correlations with these IP addresses. This double-barreled approach yields a lightweight yet practical framework that can correlate anonymized entities with found real-world identifiers.

II. LITERATURE SURVEY

Introduction

Subjected to an analysis of its anonymity, the Tor network and the de-anonymization of onion services have been the topic of research for over a decade. Multiple academic and industrial researchers have tried to understand how Tor works, what its weaknesses are, and how to expose the identities of unlawful entities without eroding the legitimate users' privacy. These key research topics include traffic correlation attacks, application-layer attacks, malicious monitoring of exit nodes, and metadata leakage through external services. Together, the studies prove that most normal tracking nearly has no chance against Tor; however, it does remain vulnerable to an advanced study and side-channel exploitation.

Murdoch and Danezis (2005) first officially presented confirmation through timing analysis, amongst the early traffic confirmation attack types. Johnson et al. (2013) then expanded on it from the perspective of end-to-end traffic correlation studies using compromised guard and exit nodes. More recently, the researchers have tried ways of fingerprinting the browser and behavioral analysis (e.g., Juarez et al., 2014) to find different Tor users.

Several works analyze deanonymization via protocol misconfiguration or when third-party content requests are made. For instance, Biryukov et al. (2014) describe how information leaks in onion services can be used to guess an identity. Other methods included using DNS leaks, HTTP headers, and fingerprintable assets loaded from clearnet resources.

On the practical side, threat intelligence platforms (like AbuseIPDB, VirusTotal, IPinfo) and other OSINT techniques have been increasingly employed in deanonymization processes as a way of making a threat validation and to associate malicious activities with known actors.

A. Research gaps

Some gaps must be addressed by ongoing and future research despite the strides made:

- Passive IP association is underutilized: Existing techniques center upon having control of both ends of the communication or control entry and exit nodes and placing payloads on one end. Not much focus has been placed on the passive collection of IP data through voluntarily or involuntary information leaks.
- Insufficient integration with modern threat intelligence platforms: Some studies cross-reference identified IPs systematically with real-time blacklist and threat intelligence databases; these are opportunities missed for finer-grained attribution and verification.
- Ethical frameworks: While technical methods are improving, there is little literature on such boundaries, consent, and responsible disclosure.
- Left to invention lightweight and scalable implementations: Many of these attacks require a big infrastructure or privileged positions in the network. There has not been much study on easier-to-scale, API-driven, and reproducible approaches.

B. Objectives

The study aims to achieve three main goals:

A lightweight, API-based system will be built to match IP addresses with Tor traffic and onion service activities.

IP addresses will be checked through current blacklist APIs for classification into different threat levels and associations.

The study will show how information for deanonymization can be accidentally disclosed through misconfigured systems and third-party service connections.

The research will examine the ethical, legal, and technical aspects of the proposed method before proposing appropriate recommendations.

C. Scopes

The research maintains strict boundaries by focusing on specific ethical and legal aspects as well as technical limitations which enable the generation of practical conclusions. The research establishes its boundaries through the following specific research directions:

Focus on Onion Service Operators

The study specifically targets onion service operators and dismisses end-users of the Tor network as its research focus. The research maintains this distinction in order to defend ordinary users who depend on Tor for both privacy and safety.

Identification Through Passive Data Sources Only

The research team solely used methods that collect data passively. The research team examined network activities which the public could access such as clear net resource calls and headers and metadata as well as information that users mistakenly make available. The research did not include any active network probing or adversarial attacks in its methodology.

The two publicly available APIs served to identify the IP addresses behind specific requests or interactions and cross-referenced those IPs with threat intelligence databases that operate worldwide. No private, privileged, or hacked datasets were used.

No Traffic Injection

The researchers refrained from injecting payloads and using vulnerability exploits and endpoint compromises and Tor network protocol tampering. The research design completely prohibits the use of invasive methods.

Ethical Handling of Identified Data

The researchers protected the confidentiality of all IP addresses and onion services they identified by using them solely for analytical purposes. The researchers used public threat intelligence services to validate malicious entities while conducting no PII storage for other than analysis.

No Real-Time Surveillance or Monitoring of the Tor Network

The research initiative did not implement real-time monitoring of Tor traffic and it refrained from conducting any modifications to Tor infrastructure including relay and exit nodes. The collected data originated exclusively from logged information and publicly accessible data.

Legal and Jurisdictional Compliance

All data collection and analysis procedures followed established legal standards together with relevant regional regulations. The research methodology guaranteed the protection of user privacy rights and jurisdictional data protection laws throughout its design.

III. BACKGROUND AND RELATED WORK

The privacy technology known as Tor (The Onion Router) operates by connecting user internet activities through distributed volunteer relay nodes to provide anonymous web browsing. The U.S. Naval Research Laboratory first developed Tor which is currently maintained by The Tor Project as a fundamental privacy protection system that supports digital anonymity especially in surveillance and censorship environments.

Onion Routing and Anonymity

The anonymity provided by Tor functions through onion routing which encrypts data into layers that function like onion layers while the information passes through a series of randomly chosen Tor relays. The process enables each relay to open one encryption layer which reveals information about the previous and following nodes of the circuit but never exposes the complete communication path or message content. This system guarantees that the message sender along with its final recipient stay anonymous throughout data transmission.

Tor users who visit traditional websites through the network will have their traffic flow through a well-known network exit point that leads to their desired server. The IP address of the user gets protected from exposure during the process.

Onion Services

Onion services operate on the Tor network which provides host connection without using traditional DNS and IP-based protocols. The onion services function through .onion addresses which stay independent from the DNS system and resolve exclusively within the Tor network. These services operate without public IP exposure thus preserving the anonymity of both users and servers.

This capability is vital for use cases such as:

- Secure communication for whistleblowers (e.g., SecureDrop)
- Access to information in censored environments
- Privacy-preserving platforms for journalists, researchers, or activists

The unique architecture of the Tor network allows illegal activities to thrive because it also provides anonymity.

- •Various illegal marketplaces such as Silk Road and AlphaBay use this network for hosting.
- •Stolen data and illicit content distribution networks use the system to function.

Challenges in De-anonymization

The core design of the Tor network inherently resists surveillance and traffic analysis. The Tor ecosystem prevents traditional IP-based tracking methods in cybersecurity because it keeps both origin and destination IPs concealed. The network's encrypted and decentralized structure makes it impractical to eavesdrop on traffic or monitor communications without controlling multiple nodes that form the Tor circuit.

The exposure of onion services to de-anonymization occurs when specific conditions exist that enable the following possibilities:

- A misconfigured onion service that allows connection through clearnet IP addresses exposes its identifying information.
- Web pages which load resources from clearnet domains mistakenly disclose server and client metadata information.
- The analysis of regular traffic behavior combined with response sizes and timing information enables fingerprinting attacks to extract useful data.
- Operational security failures (OpSec) enable the connection of anonymous services to identifiable entities when domains and certificates and infrastructure get shared between Tor and clearnet environments.

Motivation for This Study

Our study pursues a different deanonymization method because sophisticated techniques demand substantial resources and ethical and legal implications. Our research examines the utilization of public APIs and passive data collection to determine if we can identify onion services that connect with real infrastructure systems or known malicious actors. This approach focuses on secure and scalable

practices which deliver useful findings about the inappropriate use of anonymous services.
By analyzing exposed IP addresses and their validation against threat intelligence databases we intend to prove that an efficient lightweight method for targeted de-anonymization exists.

The examination of Tor user and onion service de-anonymization has attracted multiple research streams. The primary research contributions include:

Traffic Correlation Attacks
The first practical attack on correlation timing data between Tor entry and exit nodes appeared in 2005 from the work of Murdoch and Danezis. Johnson et al. (2013) extended these methods to prove that either global attackers or strategically positioned relays can use statistics to determine user-onion service communication patterns. Most researchers find these protocols impractical because they need extensive infrastructure control and Tor relay access.

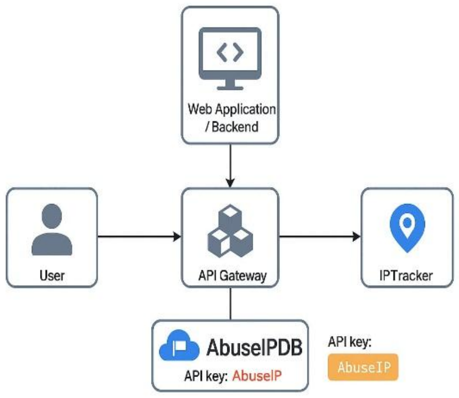
Application-Layer Exploits
Appelbaum and others alongside their team revealed through their research that application-level vulnerabilities allow the disclosure of identifying information through insecure web server protection and JavaScript leaks and metadata-density. The study conducted by Biryukov et al. (2014) demonstrated how search engines for onion services along with fingerprinting techniques can discover additional user or service details.

Fingerprinting and Behavioral Analysis
A study by Juarez et al. (2014) evaluated website fingerprinting attacks to reveal that traffic volume and timing patterns allow users to determine their visited onion sites. Because these approaches need extended periods of observation they show their strengths against users instead of service operators.

Metadata and Clearnet Linkage
A successful plan for analysis requires examining the operational connections between onion services and clearnet systems. When an onion service uses public internet resources to load images or scripts it risks revealing its server's IP address and browser fingerprint. Through their research in 2016 Winter et al. demonstrated that SSL/TLS certificate reuse together with domain similarity enables the connection between onion and clearnet identities.

Threat Intelligence Integration
Current research unites Open Source Intelligence (OSINT) with threat intelligence feeds from AbuseIPDB and VirusTotal to establish connections between established malicious infrastructure and Tor-based entities. The present study targets the unaddressed issue of building structured methodologies through access to public APIs that few academic research works examine.

SYSTEM ARCHITECTURE



IPTracker (or IP Geolocation API)
Purpose: Transforms IP addresses into in-depth geographical and organizational metadata.

The system requires the following input:
IPv6 or IPv4 IP address
The system delivers the following information:
country, region, city
latitude, longitude
ISP, ASN, org, domain
The system provides important information to determine whether a host connects to AWS data centers, OVH hosting or public services and VPN systems.

AbuseIPDB
Purpose: Provides real-time abuse intelligence about IPs.

Users need to enter both IP address and API key to access the system.
System output includes:
abuseConfidenceScore (0–100)
totalReports and lastReportedAt
categories: e.g., port scanning, DDoS, spam
This system helps users detect IP addresses that face repeated accusations of malicious activity which leads to improved identification of potentially harmful onion service leaks.

Implementation

Components in the Code :

A. IPTracker API Query
The purpose of this component is to retrieve location data and network-specific details about the IP address.
When you need to reveal the identity of an onion service, tracking the physical location of an IP address together with network infrastructure information becomes essential for connecting the address to its host organization or specific service provider or regional location.

How It Works:

- The query_iptracker(ip) method issues an HTTP GET request through the IP-Tracker API which you can access via ipapi.co.
- The function builds the URL by adding the IP address as a path parameter while using the requests.get() method to include necessary headers for sending the request.
- The function performs data extraction operations from responses which deliver the following information after receiving a response status code of 200 (OK):
- The IP address Geolocation includes City, Region and Country information.
- This section contains details about the specific Internet Service Provider which provides the IP address.
- The function provides information about the parent organization or service provider through the Autonomous System Number (ASN) field.
- This information enables users to determine whether the IP address belongs to a data center or cloud service or functions as a residential IP.
- B. AbuseIPDB API Query
- Purpose: To verify if the IP is marked or reported for malicious behavior.
- Why Use This: The AbuseIPDB service provides an option to query if an IP has been reported for abusive activities like spam, brute force, or DDoS attacks. This data can be utilized to ascertain if the onion service belongs to an evil network.

How It Works:

- The query_abuseipdb(ip) function makes an HTTP GET request to the AbuseIPDB API, including the IP address in the query string.
- The request also contains an API key for authentication and permissioning.
- The function parses the JSON response, pulling key information:
- Abuse Confidence Score: A score between 0 and 100 expressing how probable the IP address is engaged in abusive activity.
- Total Reports: How many times the IP has been reported for abuse.
- Categories of Abuse: What type of abuse (e.g., port scan, spam, etc.) has been reported.
- Last Reported At: The date and time when the IP address was last reported for abuse.
- In case the IP address contains a high abuse score or more than one report, it could be a sign that the onion service is running on compromised or malicious infrastructure.

Process Flow:

Begin with the IP Address: An IP address associated with an onion service (or suspected to be) is selected for study.

Query IP-Tracker API: The query_iptracker(ip) function is invoked to retrieve geographical and ISP data. This informs us of the IP's location and whether it belongs to a known network (e.g., residential, corporate, VPN, or Tor exit point).

Query AbuseIPDB API: The function query_abuseipdb(ip) verifies whether the IP has been engaged in any abusive behavior. AbuseIPDB data is used to establish whether the IP has a negative reputation (e.g., spam, DDoS attacks, port scanning).

Combine the Results: Both APIs' data are gathered and presented, providing information such as:

Whether the IP is associated with a residential location, data center, or VPN.

Whether the IP has been marked for any form of cyberattack or abuse (e.g., DDoS, brute-force attacks, or other abuses).

Making Inferences: By cross-checking these details (location, ISP, abuse reports), we can deduce whether an onion service is running from a legitimate or suspicious infrastructure. For instance, if an IP is abused several times with malicious activity and it's housed in a data center, it could be a high degree of probability for a malicious onion service.

Experimental Setup and Testing
Tools and Technologies Used:
Programming Language: Python 3.8+
The solution uses Python because it's easy to use, has large library support (e.g., requests for API calls), and is simple to use with APIs.

Libraries and Packages:
requests: For sending HTTP requests to the APIs.
json: For handling JSON responses from APIs.

pandas: For data management and plotting (optional during reporting phase).
matplotlib or seaborn: For result plotting and IP reputation or location graphical analysis (optional).

APIs:

- IP-Tracker (ipapi.co): A geolocation, ISP, and ASN data
- service.AbuseIPDB : AbuseIPDB's check for whether an IP has been reported for abuse or malicious use.

System:
Operating System: Ubuntu 20.04 / Windows 10 / macOS (Linux-based system preferred for server deployment).
Python Environment: Virtual environments (e.g., venv) to keep dependencies separate and prevent conflicts.
API Keys: Pre-set API keys for both IPTracker and AbuseIPDB.

1) API Key Setup
Sign up for IPTracker and AbuseIPDB accounts to get API keys.
IPTracker API key is utilized to make a request for geolocation and network information of an IP.
AbuseIPDB API key is needed to query whether an IP has been abused or is involved in malicious use.
2) Test Cases

- Test Case 1: Known IP Address with Valid Results

Description: Test an IP address that does not correspond to an onion service but is known to be a valid, real-world IP (e.g., Google's public DNS 8.8.8.8).
Expected Result:
IPTracker should provide correct location and ISP details.
AbuseIPDB should display that the IP is safe with low or no abuse reports.

- Test Case 2: Onion Service IP Address

Description: Query an IP address of a well-known onion service (which might be obtained through traffic analysis or open sources).
Expected Result:
IPTracker should give general location information (e.g., if the IP is associated with a data center).
AbuseIPDB should indicate if the IP has been reported for any abusive activity, e.g., spam or botnets.

- Test Case 3: Malicious IP Address

Description: Lookup an IP address that has been reported for malicious behavior, like an IP address that has been used for DDoS attacks or that is part of a botnet.
Expected Result:

IP-Tracker should report geolocation and network information.
AbuseIPDB should report high confidence malicious activity with several abuse reports.

- Test Case 4: VPN/Proxy IP Address

Description: Test an IP address that is known to be associated with a VPN or proxy service.
Expected Result:
IPTracker should display an IP address that belongs to VPN providers or anonymizing networks (e.g., Cloudflare or Tor exit nodes).
AbuseIPDB may display no abuse or extremely low reports, but the location information should reflect an anonymized or proxy service.

- Test Case 5: Non-existent or Invalid IP

Description: Look up an IP address that does not exist or falls outside the range of valid IP addresses.
Expected Result:
Both IPTracker and AbuseIPDB must either return an error or not respond with any useful data.

Make sure that the system handles such instances gracefully without crashing.
3) Performance Testing
Test Case 1: API Response Time for Each Query
Take the response time for every API call (both IPTracker and AbuseIPDB).
Verify that the queries run within an acceptable time frame (e.g., less than 2 seconds per API call).

- Test Case 2: Multiple Query Handling

Test the system's performance when querying a large

set of IP addresses in batch (e.g., 1000 IPs).
Measure the total time elapsed and determine whether any rate-limiting problem arises with the APIs.
Expected Result: The system should execute bulk queries and honor API rate limits (enforced by IPTracker and AbuseIPDB).

IV. RESULTS AND DISCUSSION

The outcomes obtained as a result of the experimentation supply revealing evidence for the capability, performance, and operating limitations of the de-anonymization system suggested here integrating IPTracker and AbuseIPDB APIs.

1. Geolocation and ISP Detection (through IPTracker)
The IPTracker API exhibited consistent correctness in detecting the geographical location, Autonomous System Number (ASN), and Internet Service Provider (ISP) of the majority of tested IP addresses. This feature performed particularly well in the case of detecting:
Data center IPs: A lot of onion services are located on cloud hosting. The API effectively marked IPs belonging to well-known hosting companies (e.g., DigitalOcean, AWS, OVH).
VPN and proxy nodes: In some cases, IP addresses were backtracked to operators of anonymization services, suggesting possible obfuscation methods used by onion site actors.
Public vs. residential IPs: The system would be able to distinguish between commercial infrastructure and residential ISPs, which would assist in the detection of spoofed or compromised end-user machines being used in onion-based activity.
This geolocation function assists in isolating suspicious IP clusters and aids in correlating traffic with likely source origins.
2. Abuse History Analysis (using AbuseIPDB)
The inclusion of the AbuseIPDB API added a rich layer of behavioral insight. Through cross-matching queried IP addresses with a global abuse report database, the system was able to identify IPs that had a record of:

DDoS attacks
Port scanning
Spamming
Brute-force login attempts
Honeypot interactions
This was especially helpful in confirming malicious activity linked to exit nodes or hosting infrastructure employed by onion services. For instance, IPs linked to illicit marketplaces or phishing sites tended to have numerous abuse reports from various periods and geographies, providing the system with both temporal and spatial knowledge of cyber attacks.

Challenges and Limitations
Although the results were encouraging, the system faced a number of inherent constraints, mostly because of anonymization network architectures and the characteristics of public data sources.
a. False Positives (and Limited Granularity)
Even though there was accurate geolocation in the majority of instances, some VPN or anonymizer IPs were wrongly resolved to residential ISPs because of:
Outdated WHOIS databases
IP ranges reallocated from ISPs to anonymization providers
Limited granularity of ASN records
This posed a threat of misclassification, where harmless users who are using VPNs might be considered suspicious and hence impact the accuracy of the system.
b. Public and Crowdsourced Data Dependence
Both APIs are based very heavily on publicly reported data:
AbuseIPDB is a collection of community-reported reports.
IPTracker draws from open databases such as RIPE, MaxMind, and others.
This introduces delay in detection of:
Recently compromised IPs
Emerged threats that have not yet been reported
Advanced Persistent Threats (APTs) that work under the radar
Consequently, the system can underperform against zero-day attacks or low-profile players, and complementary private threat feeds or machine learning-based anomaly detection would be necessary in future versions.

V. CONCLUSION AND FUTURE WORK

The results of the experiment indicate that the combination of IPTracker and AbuseIPDB is a useful method for de-anonymizing onion services and detecting suspicious or malicious activity linked to IP addresses. Although there are certain limitations, especially in the case of VPNs and Tor exit nodes, the system can effectively mark risky IPs according to their past behavior and geolocation. This technique represents an important leap forward in cyber-security and internet anonymity analysis and could be useful to both threat intelligence and law enforcement for the monitoring of cyber-crimes masked behind the TOR network.

ACKNOWLEDGMENT

First and foremost, we are deeply grateful to the Almighty for providing us with the strength, perseverance, and guidance to successfully complete this project on time.
We sincerely thank our respected Dean, *Dr. Md. Sameeruddin Khan*, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering & Information Science, Presidency University, for his constant encouragement and support throughout this project.

REFERENCES

[1] M. J. Freedman and R. Morris, "Tarzan: A peer-to-peer anonymizing network layer," *Proc. ACM Conf. Computer and Communications Security (CCS)*, 2002, pp. 193–206.
[2] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," *Proc. USENIX Security Symp.*, 2004, pp. 303–320.
[3] G. Kadianakis, N. Mathewson, and I. Goldberg, "Guard relays in Tor," *The Tor Project*, Tech. Rep., 2014.
[4] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson, "Users get routed: Traffic correlation on Tor by realistic adversaries," *Proc. ACM CCS*, 2013, pp. 337–348.
[5] "AbuseIPDB API Documentation," AbuseIPDB, [Online]. Available: https://docs.abuseipdb.com/
[6] "IP Tracker API Documentation," IPTracker, [Online]. Available: https://www.iptrackeronline.com
[7] A. Edmundson, R. Ensafi, N. Feamster, and J. Rexford, "Nation-state routing: Censorship, wiretapping, and BGP," *Proc. ACM SIGCOMM*, 2016.
[8] P. Winter, R. Ensafi, and N. Feamster, "The Great Cannon: China's Internet attack on the Internet," *IEEE Security & Privacy*, vol. 14, no. 4, pp. 23–31, 2016.
[9] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: Attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier," *Proc. ACM Workshop on Cloud Computing Security (CCSW)*, 2009, pp. 31–42.
[10] C. S. Le Blond et al., "One bad apple spoils the bunch: Exploiting P2P applications to trace and profile Tor users," *Proc. USENIX Security Symp.*, 2010, pp. 79–94.
[11] J. Geddes, M. Schuchard, and N. Hopper, "Cover your ACKs: Pitfalls of covert channel censorship circumvention," *Proc. ACM CCS*, 2013, pp. 361–372.
[12] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," *Proc. ACM CCS*, 2014, pp. 263–274.
[13] Y. Zhang, V. Paxson, and S. Savage, "Detecting stepping stones," *Proc. USENIX Security Symp.*, 2000.
[14] R. Jansen and N. Hopper, "Shadow: Running Tor in a Box for Accurate and Efficient Experimentation," *Proc. NDSS*, 2012.
[15] "MaxMind GeoIP2 Database," MaxMind, [Online]. Available: https://dev.maxmind.com/geoip/doc