Using Disposable Domain Names to Detect Online Card Transaction Fraud

Roy Laurens°, Hossein Rezaeighaleh¶, Cliff C. Zou∫, Jusak Jusak§

°§ Department of Computer Engineering, Institut Bisnis dan Informatika Stikom Surabaya, Surabaya, Indonesia
°¶∫Department of Computer Science, University of Central Florida, Orlando, FL, USA
°rlaurens@knights.ucf.edu, ¶rezaei@knights.ucf.edu, ∫czou@cs.ucf.edu, §jusak@stikom.edu

***Abstract* — Online card transaction fraud is one of the major threats to the bottom line of E-commerce merchants. In this paper, we propose a novel method for online merchants to utilize disposable ("one-time use") domain names to detect client IP spoofing by collecting client's DNS information during an E-commerce transaction, which in turn can help with transaction fraud detection. By inserting a dynamically generated unique hostname on the E-commerce transaction webpage, a client will issue an identifiable DNS query to the customized authoritative DNS server maintained by the online Merchant. In this way, the online Merchant is able to collect DNS configuration of the client and match it with the client's corresponding transaction in order to verify the consistency of the client's IP address. Any discrepancy can reveal proxy usage, which fraudsters commonly use to spoof their true origins. We have deployed our preliminary prototype system on a real online merchant and successfully collected clients DNS queries correlated with their web transactions; then we show some real instances of successful fraud detection using this method. We also address some concerns regarding the use of disposable domains.**

Keywords—Electronic Commerce; fraud detection; Disposable Domain Name; DNS; Authoritative Name Server; Proxy Detection; Security

I. INTRODUCTION

Establishing the integrity of client's IP address has a crucial security implication. This is especially true in the E-commerce environment, where many fraud detection methods, such as velocity checks [1], rely on client reporting its true IP address. On the other hand, many freely available tools allow clients to easily conceal their true IP addresses [2]. Online transaction fraudsters will often utilize these tools (such as proxy or VPN) to circumvent the velocity checks mentioned above. Thus, being able to verify the integrity of client's IP address is important in detecting fraudulent transaction attempts. Furthermore, it could also benefit other security applications as well (e.g., remote user login, geolocation-based access control, etc.).

In this paper, we propose a novel method for detecting IP address tampering in online card transactions by examining the DNS configuration of the client's machine using disposable domain names. It begins with inserting a uniquely generated "one-time use" hostname on the Merchant's webpage. This force the client to issue a DNS query for that hostname to its local DNS server, which will propagate to the Merchant's authoritative name server. The server can then match this query with the correct transaction for further analysis. For more information on DNS protocol and the various servers involved in its operation, please refer to [3][4].

To obtain real E-commerce transaction dataset and test the proposed system in practical environment, we collaborate with an E-commerce company that allows us to install our code on their production website and their authoritative DNS server for several months; the resulting dataset is used as the basis of our analysis. Since this is a real life E-commerce server, we are able to collect data from tens of thousands of transactions from thousands of client devices across hundreds of countries. The dataset also contains interesting observation on the prevalence and utilization statistics of public DNS servers, such as Google DNS or OpenDNS [5].

The structure of this paper is as follows: In the next section, we provide previous related works before modeling the adversary and the details of our approach in Section III. Next, Section IV shows the real life dataset that we collected from our E-commerce Merchant partner. Section V discusses various results we found from the dataset and our effort to address the concerns regarding the use of disposable domains. Finally, Section VI concludes the paper.

II. RELATED WORK

A. Fraudulent Transactions Detection methods

A vast body of literature has proposed various methods for collecting and identifying data and information of a website's client devices using various methods. Later analysis of this so-called fingerprinting information can detect anomalies commonly associated with fraud [6]. In general, there are two types of fingerprinting methods: active and passive fingerprints. An active fingerprinting acquires information by sending probes deliberately, while on the other hand, a passive fingerprinting obtains information by sniffing and monitoring traffic data silently even without making the targeted systems aware. In this sense, our proposed approach is more similar to passive fingerprinting as it can collect data non-obtrusively.

In [7], the author used DNS analysis techniques to detect domains that were engaged in malicious activities. The paper proposes extracting a large number of features from DNS traffic to detect wide variety of malicious domains automatically, which include activities such as botnet commands, phishing and scamming.

Authors in [8] introduced operating system (OS) fingerprinting based on DNS query characterization. Their paper showed that a specific OS sent queries with specific time-interval patterns. Hence, analysis of DNS queries received by DNS name servers can reveal information about the OS of the DNS resolver that sent these queries.

Meanwhile, papers in [9] and [10] use the client's DNS activities to obtain behavioral. By exploiting DNS queries of particular users, the work in [9] reveals that it can detect the presence of an anonymized user in a new DNS stream with high accuracy based on the user's historical DNS activities. Similarly, the work in [10] attempted to capture the profile of users who had dynamic IP addresses by employing some pattern mining techniques to the DNS traffic.

As for checking the integrity of reported IP address, authors in [11] explore several proxy detection techniques to prevent fraudulent activities in real e-commerce services. The paper at the same time proposed a detection method based on the previous knowledge of cyber criminals who had gained access to the system and performed preventive action before the cyber criminals commit fraud.

There is also a well-established standard for protecting the integrity of DNS protocol itself, which is called DNS Security (DNSSEC) and is specified by Internet Engineering Task Force (IEFT) in [12]. However, it is designed to protect against attempts to hijack the protocol to redirect traffic to deceptive websites and does not address fraudulent client activities.

B. Disposable domain names usage

Most papers viewed disposable/one-time domain names as a misuse of DNS protocol. The literatures viewed this misuse as either malicious (such as botnet command and control [13] or NXDOMAIN hijacking [14]), or somewhat benign (performance improvement [15] [16], analytics and file reputation queries [17]). The objective of the papers itself are mostly on the detection of the misuse [18] and mitigating its impact on the various component of DNS protocol, in particular the DNS name servers [19].

C. Evaluation of the previous works

All of the above-mentioned existing works on fraud detection put emphasis on capturing the DNS traffic using the ISP's DNS Resolver as this server directly communicates with the client and therefore the client's data is more readily available. As a result, e-commerce merchants cannot use this approach because they are simply content providers and do not own or control the ISP used by the client. Our work, on the other hand, allows content providers to obtain the client's DNS configuration without any help from any ISP or intermediaries.

More importantly, compared to other proxy detection techniques, our work can directly correlate a DNS Query with its web session/transaction using simple static HTML image tag, which is impervious to most script manipulation, such as script editing [20] or JavaScript disabling [21]. Of course, a determined fraudster could attempt to block *any* image download from a webpage. However, such action will clearly raise the flag further as every single transaction has its own tracking and any missing DNS Query for a transaction can be easily detected and flagged as suspicious.

Concerning existing works on disposable domain names, our proposal is unique because it utilizes DNS protocol as a fraud detection method. Furthermore, our work also attempts to mitigate its on DNS infrastructure by minimizing traffic and reducing its cache burden.

III. DNS Configuration Profiling

A. Adversary Model

In this paper, the online merchant wants to detect if the IP address of the client is real or if there is indication of manipulation attempt to conceal the real IP address using proxy or similar technology.

Some specific characteristics of these concealment attempts are important for our model:

- The fraudsters behind this will want to monetize a stolen card as soon as possible – prior to the issuer blocking the card. As such, fraudsters must conduct fraudulent transactions within a short time period. This is the basis of the *velocity check* commonly performed by rule-based detection methods [1]. At its core, a velocity check detects if the same person (i.e., same email, same IP address, etc.) made numerous purchases within a short timeframe.
- To defeat velocity detection, fraudsters will attempt to conceal their IP addresses [22]. Nevertheless, they must perform this efficiently, due to the time constraints mentioned above. They might use freely available proxy, or use VPN Provider. However, it is very unlikely that they will install their own proxy or VPN server at a remote location. Hence, they will have very limited or no control at the entire proxy server [2] they utilize, including its DNS configuration.
- Another reason for an online fraudster to hide his real IP address is to deceive the merchant into thinking that the client's physical location is within the same geographical location as the stolen payment card's billing address. This is a security feature of the credit/debit card known as Address Verification System (AVS) [23]. As North America and Europe issue the majority of stolen credit/debit cards [24], most fraudsters will attempt to masquerade themselves as coming from these regions too, and hence we should focus our proxy IP addresses and detection attempt in these regions as well.

In summary, we want to analyze if there are any patterns of IP manipulation that we can discover by looking at the DNS configurations of suspicious transactions with IP addresses that are located in North America or Europe. We believe DNS configuration can help reveal these patterns and hence complement and enhance existing fraud detection methods.

B. Client DNS Configuration Data Gathering Methodology

To obtain client DNS configuration, we want to be able to:

1. **Make the client perform DNS query to E-commerce Merchant's own authoritative name server.** Forcing the client to perform DNS query seems very straightforward, as every client will need resolve the name of the hosts present in the webpages into their IP addresses. However, for efficiency DNS clients and local DNS servers have caching mechanism, which could prevent many DNS queries from going directly to the corresponding authoritative name servers [25]. In order to guarantee that checkout webpage for every online financial transaction will generate a DNS query that can be received and uniquely identified by the

online Merchant's authoritative name server, the Merchant webserver *dynamically generates a unique hostname* and embeds it as an HTML asset. In this way, the client's web browser will always have to resolve it from the authoritative name server since this unique hostname has never been resolved and cached in local DNS servers.

2. **Associate the DNS query with the transaction/session that generates that query.** In order to match the query with the transaction, we must embed the data into the DNS query message itself. Unfortunately, the only field on the message that a webpage or webserver can control is the hostname that is used to reference an HTML asset (i.e., image, css, etc.). Hence, we must embed the transaction id into the hostname so this identifier will propagate back into our authoritative name server, which the server will match with the corresponding transaction.

In order to successfully implement this technique, we need to update and construct two components:

- **A webpage containing an asset (i.e., image, css, etc.) referenced with the unique hostname**. A server-side or client-side scripting can generate the hostname. Server-side is more secure but it might not be practical for websites that uses static pages. Client-side scripting is less secure, but we can use obfuscation techniques to hide the script and the hostname to hinder their tampering. Merchants can put this page anywhere in the checkout process of an online credit/debit card transaction.
- **A custom-made DNS authoritative name server**. As every single transaction will generate a unique hostname, we need custom-made DNS authoritative name servers that will answer/respond to all hostname queries. These servers will answer all hostname queries that follow our format and encoding. Furthermore, they will also parse the hostname to recover the transaction ID and update our transaction database with the DNS query that it receives.

Figure 1 illustrates the diagram of this proposed technique.

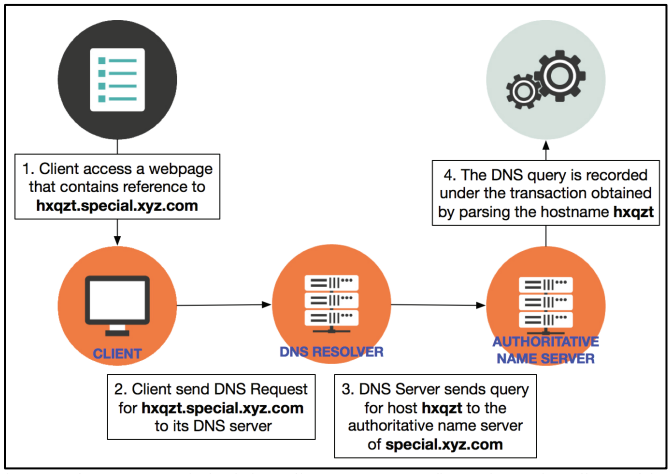

Fig. 1. The process of obtaining client's DNS configuration using disposable domain. By embedding the client's transaction ID into the dynamically-generated unique hostname, the online Merchant can correlate client IP with its DNS Resolver.

C. *Method Implementation*

Since we are deploying our technique on a merchant server that runs live transaction, we have to ensure compatibility with the merchant's current system. The merchant uses static pages hosted on Amazon's Content Delivery Network for its E-commerce website. However, the checkout page itself is using Amazon's Cloud Compute, hence we choose to use server-side scripting (PHP) to generate the unique hostname. We obtain the final hostname by combining two-character random salt with the 10-digit transaction id, encrypting them with simple PHP **mcrypt** function and converting the result to hexadecimal representation. This is not the most efficient use of character space, but some DNS servers randomize the capitalization of hostname and do not support special characters. Therefore, to ensure robust transmission of hostname, we choose hexadecimal representation. The final html tag on the webpage will look like this:

```
<img src='https://a1b2c3d4.subdom.mybiz.com/a.gif'>
```

For coding the custom-made authoritative name server, we modify the freely available PHP DNS server [26]. Although PHP is not the most efficient language for implementing DNS server, our collaborative Merchant uses PHP/MySQL for their transaction processing, so by also using PHP we can ensure compatibility with their existing transaction database access. It is also easy to recover the transaction ID back from the hostname as we simply reverse the process using similar PHP functions. Furthermore, their transaction activity is light – hence there is no requirement for high performance DNS authoritative name server. We configure the name server so it responds only to queries that follow our format. This security measure is necessary as our server receives malicious DNS queries within *minutes* of coming online.

To facilitate system deployment, and impose minimum change to an online Merchant's existing E-commerce system, the customized authoritative name server will be an added component without touching the original authoritative name server of an online Merchant. To achieve this goal, we add a subdomain record in the Merchant's DNS zone so that our special name server will act as the name server only for all name queries in the specified subdomain. For example, suppose the Merchant domain name is `mybiz.com`, and the subdomain for our DNS profiling is `subdom.mybiz.com`, then our customized authoritative name server will respond to all hostname queries in that subdomain (i.e., `*.subdom.mybiz.com`).

D. *Benefits Provided by the DNS Configuration Profiling*

Our method offers several advantages compared to existing fingerprinting/tracking/proxy detection techniques:

- Fraudsters cannot disable or modify this tracking. The tracking hostname can host real asset needed by the webpage (image, stylesheet, etc.), so disabling it means rendering the webpage non-functional, which can be easily detected and marked as suspicious. In comparison, JavaScript-based tracking code, such as the methods proposed in [27], is prone to modification or deactivation. Our dataset supports this assessment, as we are able to obtain and correlate DNS data for 100% of our transactions.

- It is difficult or impossible for fraudsters to manipulate the data, as modification requires direct control of the proxy server, which is seldom possible. Furthermore, *any* modification must be consistent with data collected for *all* other users of the same proxy server or even servers on the same subnet, otherwise we consider the discrepancies as suspicious.
- For caching, security monitoring, or even censorship purposes, numerous network devices in the path from a client to its used proxy server can perform their own *additional* DNS Query for all hostnames that pass through their system [28] [29]. For example, some ISPs, even WiFi Routers, autonomously send DNS Query for unknown hostnames contained in the network traffic passing through them. If this happens, the authoritative name server in our system will receive multiple DNS queries for the same hostname, matching to a single online card transaction. These additional queries will inadvertently disclose the Client's real location or at least part of its network path to the proxy.
- Finally, our method only runs codes on the Merchant's own server and does not require any coordination and cooperation from other parties. Online merchant can deploy this independently as it leverages Merchant's existing infrastructure and, assuming the Merchant already run its own authoritative name server, it does not incur any additional cost.

IV. Dataset Evaluation

To obtain real E-commerce transaction dataset and test the proposed system in practical environment, we collaborate with an E-commerce company that allows us to install our code on their production website and, as we don't want to disrupt its existing authoritative DNS, an additional authoritative DNS server. We deployed our code on the Merchant's server from November 2017 until March 2018. During that time, we collected DNS data from 18,974 transactions.

In addition, since we want to analyze abnormalities in DNS queries originated from the same IP or IPs within the same subnet, we group together transactions based on the subnet of their reported IP addresses. In order to simplify our analysis, we assume a class C (/24) subnet. Next, we write a script to report subnets where the majority (> 75%) of all transactions of that subnet is considered suspicious or fraudulent, as reported by our merchant's fraud team. We choose lower threshold than 100% because we assume that our Merchant could have missed detecting some fraudulent cases. Finally, we manually look at each of these reported subnets and analyze the DNS address patterns that we saw.

In order to obtain the geolocation and organization of a client IP address, we use freely available web tools such as `iplocator.net`.

As mentioned earlier, our DNS data gathering method is very robust as we got 100% of the transactions correlated with their DNS data. In other words, no transaction failed to report its client's DNS configuration. In comparison, JavaScript fingerprinting code [27] has up to 5% failure in collecting data on the same transactions dataset. This is because disabling JavaScript is easier and often does not break the webpage; it is also commonly existed for some legitimate users. Whereas disabling image, etc. to disable our proposed DNS data gathering will usually render the webpage unusable.

A. Normal DNS Behavior

There are two types of DNS queries: iterative and recursive. *Iterative* queries will be resolved without querying other DNS servers, even if it means the answer is not definitive. *Recursive* queries will attempt to query other DNS servers until it obtains a definitive answer.

Common desktop and mobile Operating System, such as Windows, Android or Apple's IOS, runs a simplified DNS Resolver called stub resolver [3]. This minimal resolver might contain caching functionality but it depends on its ISP's DNS server to resolve unknown hostname and will use recursive query to do so. The ISP's DNS servers are the ones that send DNS queries to our customized authoritative name server and their configuration is the one that we record. Therefore, clients from the same ISP (same subnet) will usually share the same one or two local DNS Server address, all of which belongs to the same organization/ISP. For example, client with IP address 73.21.251.160 (Geolocation: US, ISP: Comcast) has DNS Server address of 69.252.68.139 (Geolocation: US, ISP: Comcast).

However, after looking at both the transactions and their corresponding DNS Resolver addresses for high-risk subnets, we could identify several anomalies in their DNS configuration.

B. Identical IP and DNS Address

A transaction from these subnets has the same address for both its IP address and its DNS Resolver address. This means that the same device has a fully functioning DNS Resolver. However, since ordinary buyer's device should only have a stub resolver [3], it means that the device used here is not a common desktop/mobile device, but potentially a server itself: it probably runs a proxy service – including DNS Server. Table 1 shows several examples belonging to this abnormal category.

IP = DNS	
Client IP	**DNS**
108.62.5.130	108.62.5.130
108.62.5.36	108.62.5.36
38.132.120.66	38.132.120.66
173.239.240.159	173.239.240.159
185.153.176.2	185.153.176.2

Table 1. Sample list of transactions where the DNS Resolver IP address is the same as the client's IP address

C. DNS Geolocation Differs from Client's IP

The DNS Resolver from these transactions is located on a different country from where the corresponding client IP is, which clearly is abnormal since a legitimate client's own ISP usually has its DNS Servers located nearby to speed up the name resolving process. The reasonable explanation for this abnormal behavior is that the client is hiding his IP address by using proxy and his ISP (or in some cases, the client's home router) is intercepting the packet destined to the proxy and performing its

own DNS query. When this happens, the extraneous query pierces the confidentiality of the proxy and reveals the client's true location to us.

We show sample transactions belonging to this category in Table 2. In the Indonesian and Turkey DNS server case, we investigated it further and found that Indonesian and Turkey ISPs resolved unknown hostname automatically as part of their traffic monitoring and censorship actions [28] [29].

This explanation does not work in the other direction, as it is common for high-risk country devices to use American or other public DNS server to avoid their country's Internet censorship.

IP Geo ≠ DNS Geo			
Client IP	**IP Geo**	**DNS**	**DNS Geo**
77.234.46.224	USA	178.18.201.113	Turkey
77.234.46.194	USA	202.152.254.245	Indonesia
138.197.174.117	Canada	180.251.20.148	Indonesia
138.197.174.154	Canada	41.226.16.50	Tunisia

Table 2. Sample list of transactions where the DNS Resolver geolocation is from a high-risk country that differs from the corresponding client's IP geolocation

D. *Various DNS Resolvers in the Same Subnet*

In this case, we found subnets where clients that have IP addresses in close proximity (sometimes even contiguous) use numerous different DNS Resolvers and these Resolvers come from different organizations. Obviously, this does not make sense. In a normal situation, clients DNS configuration comes from their ISP, so IP addresses from the same subnet from the same ISP should have the same DNS configuration. However, if the client is hiding its IP address and is using HTTP-only proxy [30], for example, the client IP observed by online Merchant will be the IP of the proxy. In such a scenario, the DNS Resolver of the same IP or same subnet could be totally different as the proxy could be used by many different clients that are originated from different ISPs, each with its own DNS configuration. Table 3 shows several transaction examples belonging to this category.

This also shows our method's strength against tampering. A fraudster might change his DNS configuration to try to match his proxy persona. However, he won't be able to change the configuration of other user in the same subnet, and hence this inconsistency will reveal his manipulation attempt.

Same Client IP subnet, too many different DNS subnet		
Client IP	**DNS**	**DNS Org**
172.98.87.112	86.51.29.38	Bayanat Al-Oula
172.98.87.113	37.107.255.149	SaudiNet
172.98.87.223	216.87.131.212	Verisign
172.98.87.246	66.249.84.58	Google

Table 3. Sample list of transactions where clients from the same subnet (172.98.87.0/24) use DNS Resolvers from many different organizations

E. *Sharing Subnets between IP Address and DNS Server*

In this case, many clients' IP addresses are located within the same subnet as their DNS server farm. The presence of numerous DNS servers on the same subnet is not abnormal, as this might simply be an indication of a DNS farm for load balancing. However, it also means that other addresses in the same subnet are servers as well, as it is very insecure (and therefore very unlikely) for ISPs to have clients and public servers sharing the same subnet [31]. Hence, client transactions that use this subnet are suspicious as it is very likely that they originate from servers (i.e., proxy) and not from a real client machine. Table 4 shows several transaction examples belonging to this category.

Client IPs in the same subnet with DNS farm	
Client IP	**DNS**
5.62.59.11	5.62.59.212
5.62.59.13	5.62.59.194
5.62.59.17	5.62.59.195
5.62.59.21	5.62.59.196
5.62.59.26	5.62.59.197
5.62.59.29	5.62.59.198
5.62.59.37	5.62.59.200
5.62.59.45	5.62.59.203

Table 4. Sample list of transactions where clients uses identical subnet for both IP address and DNS server farm (5.62.59.X)

F. *Summary: comparison to other fraud detection methods*

We summarize our findings below. As the usage of disposable domain names is meant to supplement existing fraud detection methods but not to replace them, we are only interested in precision (true positive fraction of all suspected transactions) and not recall. We use the Merchant's existing fraud detection result as the ground truth.

Type	Precision	Nbr of txn
IP = DNS	100%	34
IP Geo <> DNS Geo	28.18%	1,661
Same subnet, different DNS	97.82%	275
IP/DNS share subnet	100%	208

Table 5. Precision of disposable domain method in detecting fraud

The disposable domain names method shows a high degree of precision, except for the case of difference in geolocation between the client's IP and its DNS. However, as mentioned previously, there are legitimate reasons to use different DNS than the one allocated by the client's ISP. Hence, the majority of these cases are benign and it shows in the result.

G. *General Observation on Third-party DNS*

Out of the 18,974 transactions collected in our dataset, there are 1,661 (8.7%) that use third-party (i.e., the DNS IP address is registered to entity other than the client's ISP). We grouped the entities that own these DNS servers and figure 2 shows the result.

The vast majority of these DNS are well-known public DNS servers: Google and OpenDNS [32] [33]. Together, they count for more than 80% of the data. The remaining third parties are not too well known, with some of them actually come from suspicious transactions mentioned earlier.

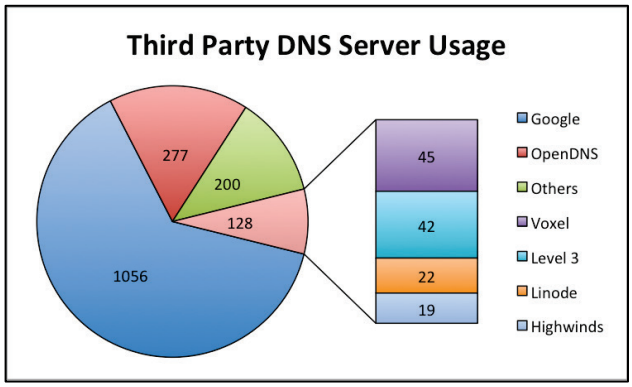

Fig. 2. The statistics of third party DNS server usage by entity.

As Google is by far the largest entity, we further analyze Google DNS data by looking into geolocation, as shown in figure 3.

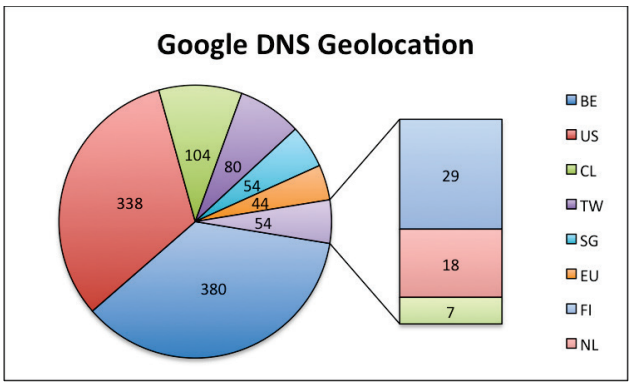

Fig. 3. Google public DNS geolocation referenced by ISO country code.

As mentioned in the overview of their public DNS server [34], Google DNS host its data centers worldwide, and it uses anycast routing to send users to the geographically closest data center [35]. As the majority of our collaborative Merchant's customers originate from North America, Europe and South America, the statistics reflects this by showing that the majority of Google DNS' geolocation is from US, Belgium and Chile, respectively.

H. Access Time on Client's Browser

We measure the impact of our additional DNS lookup from various locations in the world as shown in Table 6. USA location has the best time since the name server is located in Virginia. However, even in the worst case, the impact to customer's experience is minimal.

US	Asia	Europe	South America
57.30	282.6	142.4	248.6

Table 6. Average access time from around the world (ms)

V. Discussion

A. Broader usage

Although our focus is on analyzing DNS profile for suspicious online debit/credit card transactions, it is also applicable to other applications where integrity of client's IP address is crucial. Several contemporary examples are geolocation-restricted content from streaming service such as Netflix, or even simple remote user web login that has geolocation-based restriction.

As our approach uses domain owner / content provider's own web / authoritative name server and therefore does not incur additional costs, it can be deployed even in the most mundane applications. In fact, we can use it on any web interaction that has a session associated with it. We believe this proposed method has the potential of greatly increasing the security of website interaction and Internet in general.

B. Limitation

Although our approach is successful in using DNS data to detect IP spoofing activity, sometime a client has a legitimate reason to hide its identity, as in the case to circumvent monitoring and censorship by its local government. Fortunately, most frauds originate from IP addresses that purport to come from developed countries – as this is where most of the stolen cards originated from – hence we can focus on these countries instead [24].

Another observation is that we can achieve 100% data-gathering rate. We believe this is because our DNS profiling method is new and therefore the fraudsters do not expect this and thus have not implemented any countermeasures. Nevertheless, if a webpage asset download is disabled, we can resort to incorporate our profiling hostname into the checkout server hostname itself. That is, we make unique payment server hostname for every single transaction, for example: https://a1b2c3.payment.mybiz.com. However, the downside is that we must use SSL wildcard certificate and we cannot use the more secure SSL Extended Validation (EV) certificate [36]. This is because EV certificate must associate with only one domain name and it does not work for multiple domain names such as in wildcard certificate.

C. Public DNS Proliferation

There is a possibility that ISPs might start relegating their DNS service to Google or other public DNS for cost-saving reason. Although it seems that this would render our method ineffective, in reality it would just reduce our accuracy as Google, for example, would have to increase their geographic coverage and we would still be able to get some geolocation data albeit with lower precision.

More importantly, however, is that Google DNS recently supports EDNS Client Subnet (ECS) [37], which is a DNS extension that provides client subnet as part of the query [38]. This is even better from our profiling perspective, as we will be able to directly obtain the source address of the client that initiates the DNS query. This is an interesting development and we plan to perform further research in this area.

D. Reducing burden on DNS Protocol

We implement several techniques to reduce the impact of our approach to the DNS protocol and its components. First, to reduce the impact on DNS resolver, we use small TTL value (30 seconds) and we only use second level subdomains as our unique hostname. Furthermore, we only implement our code on the checkout page, which means each transaction only generates one extra DNS request.

VI. Conclusion

In this paper, we have shown how a carefully crafted webpage asset reveals client's DNS configuration, which in turn helps detect IP concealment attempt commonly associated with online fraud. By analyzing the dataset collected from a real online Merchant, we show how analysis of DNS profile can reveal various IP concealment attempts. Furthermore, we demonstrate how our approach is robust and difficult to tamper with. Finally, we show several statistics for public DNS servers based on our collected dataset, and how we can still utilize our method even under this circumstance.

Acknowledgement

This work is supported by the National Science Foundation under grant DGE-1723587, and the grant from US Army PEO STRI.

References

[1] K. R. Seeja, Masoumeh Zareapoor, "FraudMiner: A Novel Credit Card Fraud Detection Model Based on Frequent Itemset Mining", *The Scientific World Journal*, September 2014.

[2] R.-M. Lin, Y.-C. Chou and K.-T. Chen, "Stepping Stone Detection at The Server Side", *2011 IEEE Conference*, Shanghai, 2011.

[3] P. Mockapetris, "Domain names – concepts and facilities," *Internet Request for Comments (RFC 1034)*, November 1987.

[4] P. Mockapetris, "Domain names – implementation and specification," *Internet Request for Comments (RFC 1035)*, November 1987.

[5] A. Kesavan, "Comparing the performance of popular public DNS providers", https://www.networkworld.com/article/3194890, *Network World*, May 2017.

[6] R. Laurens, J. Jusak, and C.C. Zou, "Invariant diversity as proactive fraud detection mechanism for online merchants," in *Proc. IEEE Globecom 2017*, Singapore, Dec. 2017.

[7] L. Bilge, E. Kirda, C. Kruegel, M. Balduzzi, "EXPOSURE: finding malicious domains using passive DNS analysis," *ACM Transaction on Information and System Security*, Vol. 16., No. 4, Apr. 2014.

[8] T. Matsunaka, A. Yamada, A. Kubota, "Passive OS fingerprinting by DNS traffic analysis," *IEEE 27th International Conference on Advanced Information Networking and Applications*, Barcelona, Spain, Mar. 2013.

[9] D.W. Kim, J. Zhang, "Deriving and measuring DNS-based fingerprints," *International Journal of Information Security and Applications*, Vol. 36, pp. 32-42, 2017.

[10] D. Herrmann, C. Banse, H. Federrath, "Behavior-based tracking: exploiting characteristic patterns in DNS traffic," *Computers and Security*, Vol. 39, pp. 17-33, 2013.

[11] M. Pannu, B. Gill, R. Bird, K. Yang, B, Farrel, "Exploring proxy detection methodology," *IEEE International Conference on Cybercrime and Computer Forensic*, Vancouver, Canada, June 2016.

[12] R. Arends, et.al., "DNS Security Introduction and Requirements," *Request for Comments (RFC 4033)*, Mar. 2005.

[13] S. Yadav, A. K. K. Reddy, A. L. N. Reddy, S. Ranjan, "Detecting algorithmically generated domain-flux attacks with DNS traffic analysis", *IEEE/ACM Trans. Netw.*, Oct. 2012.

[14] P. Vixie, "What dns is not, " *ACMQueue, no. 10*, Nov. 2009.

[15] U. Goel, M. Steiner, M. Wittie, S. Ludin, and M. Flack, "Domain-Sharding for Faster HTTP/2 in Lossy Cellular Networks", *arXiv*, 2017.

[16] J. Otto , M. Sánchez, J. Rula , F. Bustamante, "Content delivery and the natural evolution of DNS", ACM conference on Internet measurement conference, November 2012.

[17] McAfee, "Faqs for global threat intelligence file reputation", https://kc.mcafee.com/corporate/index?page=content&id=KB53735, 2013.

[18] D. Chiba, T. Yagi, M. Akiyama, T. Shibahara, T, Yada, T. Mori, S. Goto, "DomainProfiler: Discovering Domain Names Abused in Future", *Dependable Systems and Networks (DSN) 2016 46th Annual IEEE/IFIP International Conference on*, 2016.

[19] Y. Chen, M. Antonakakis, R. Perdisci, Y. Nadji, D. Dagon, W. Lee, "DNS noise: Measuring the pervasiveness of disposable domains in modern DNS traffic", *IEEE Dependable Systems and Networks (DSN) Conference*, 2014.

[20] R. Johari, P. Sharma, "A Survey on Web Application Vulnerabilities (SQLIA XSS) Exploitation and Security Engine for SQL Injection", *International Conference on Communication Systems and Network Technologies (CSNT)*, May 2012.

[21] Panopticlick, "Is your browser safe against tracking?", https://panopticlick.eff.org/self-defense, *Electronic Frontier Foundation*, 2010.

[22] ThreatMetrix, "Device Fingerprinting", https://www.threatmetrix.com/wp-content/uploads/2010/03/ThreatMetrix_Datasheet.pdf, 2010.

[23] VISA, "Card-Not-Present Security: A Multi-layered Approach to Payment Card Security", http://www.visa.ca, December 2010.

[24] U.S. Payments Forum, "Card-Not-Present Fraud around the World", https://www.uspaymentsforum.org/wp-content/uploads/2017/03/CNP-Fraud-Around-the-World-WP-FINAL-Mar-2017.pdf, March 2017.

[25] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, "DNS Performance and the Effectiveness of Caching", *IEEE/ACM Transactions on Networking*, 2002.

[26] Yswery, "An Authoritative DNS Server written purely in PHP", https://github.com/yswery/PHP-DNS-SERVER, Oct 2017.

[27] R. Upathilake, Y. Li, A. Matrawy, "A classification of web browser fingerprinting techniques", *New Technologies Mobility and Security (NTMS) 2015 7th International Conference on*, 2015.

[28] "Internet Sehat dan Aman (INSAN)", https://kominfo.go.id/content/detail/3303/internet-sehat-dan-aman-insan/0/internet_sehat, Kementrian Komunikasi dan Informatika Republik Indonesia, 2013.

[29] A. Di Florio, N.V. Verde, A. Villani, D. Vitali, L.V. Mancini, "Bypassing censorship: a proven tool against the recent internet censorship in Turkey", *2014 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 2014.

[30] "Squid 3 proxy", http://www.squid-cache.org/Intro/

[31] Cisco Systems, "Cisco on Cisco Best Practices; Cisco IP Addressing Policy", https://www.cisco.com/c/dam/en_us/about/ciscoitatwork/downloads/ciscoitatwork/pdf/Cisco_IT_IP_Addressing_Best_Practices.pdf, 2009.

[32] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing DNS resolvers in the wild", *Proceeding of lMC*, 2010.

[33] C. Huang, D. A. Maltz, A. Greenberg, and J. Li, "Public DNS system and global traffic management", *Proceeding of lEEE lNFOCOM*, 20ll.

[34] Google, "Introduction to Google Public DNS", https://developers.google.com/speed/public-dns/docs/intro, 2016.

[35] Google, "Distributing serving clusters for wide geographical coverage", https://developers.google.com/speed/public-dns/docs/performance#geography, 2016.

[36] "Guidelines For The Issuance And Management Of Extended Validation Certificates", https://cabforum.org/wp-content/uploads/CA-Browser-Forum-EV-Guidelines-v1.6.8.pdf, 2018.

[37] Google, "EDNS Client Subnet (ECS) Guidelines", https://developers.google.com/speed/public-dns/docs/ecs, 2016.

[38] C. Contavalli, et. al., "Client Subnet in DNS Queries," *Internet Request for Comments (RFC 7871)*, May 2016.