

VPN Traffic Detection in SSL Protected Channel

Muhammad Zain ul Abideen¹, Shahzad Saleem², Madiha Ejaz³
School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology (NUST)

¹mabideen.msis18seecs@seecs.edu.pk

²shahzad.saleem@seecs.edu.pk

³madiha.ejaz311@gmail.com

Abstract

In recent times secure communication protocols over web such as HTTPS (Hyper Text Transfer Protocol Secure), are being widely used instead of plain web communication protocols like HTTP (Hyper Text Transfer Protocol). HTTPS provides end to end encryption between user and service. Now a days organizations use Network firewalls and/or Intrusion Detection and Prevention Systems (IDPS) to analyze the network traffic to detect and protect against attacks and vulnerabilities. Depending on the size of organization these devices may differ in their capabilities.

Simple Network Intrusion Detection System (IDS) and firewalls generally have no feature to inspect HTTPS or encrypted traffic, so they rely on unencrypted traffic to manage the encrypted payload of the network. Recent and powerful Next Generation Firewalls have Secure Sockets Layer(SSL) inspection feature [1] which are expensive and may not be suitable for every organizations. A Virtual Private Network (VPN) is a service which hides real traffic by creating SSL protected channel between the user and the server. Every internet activity is then performed under the established SSL tunnel.

User inside the network with malicious intent or to hide his activity from the network security administration of the organization may use VPN services. Any VPN service may be used by users to bypass the filters or signatures applied on network security devices. These services may be the source of new virus or worm injected inside the network or a gateway to facilitate information leakage.

In this paper we have proposed a novel approach to detect VPN activity inside the network. The proposed system analyzes the communication between user and the server to analyze and extract features from network, transport and application layer which are not encrypted and classify the incoming traffic as malicious i.e. VPN traffic or standard traffic. Network traffic is analyzed and classified using DNS (Domain Name System) packets and HTTPS (Hyper Text Transfer Protocol Secure)

based traffic. Once traffic is classified, the connection based on the server's IP, TCP port connected, domain name and server name inside the HTTPS connection are analyzed. This helps in verifying legitimate connection and flag the VPN based traffic.

We worked on top five freely available VPN services and analyzed their traffic patterns, the results show successful detection of the VPN activity performed by the user. We analyzed the activity of five users, using some sort of VPN service in their internet activity, inside the network. Out of total 729 connections made by different users, 329 connections were classified as legitimate activity, marking 400 remaining connections as VPN based connections. The proposed system is light weight enough to keep minimal overhead, both in network and resource utilization and requires no specialized hardware.

Keywords— Network Security, Network Forensics, SSL Analysis, Information Hiding, Insider threat Analysis, Network Analysis

1 Introduction

To enable the communication between the computers TCP/IP stack was implemented. The stack was implemented without the consideration of security of information being transferred in the communication [2]. This issue raised a lot of security concerns which are constantly managed by different security services [3]. Secure Socket Layer (SSL) is commonly used to provide authentication and encryption security service in TCP/IP stack [4].

The trend of encrypted traffic in the network has largely increased in last decade due to security concerns in general and privacy concerns in specific [5]. The encryption has provided a lot of benefits for the user ensuring end to end secrecy and data confidentiality. The need to inspect the traffic originating or destined for the organization's network has immensely increased for many security reasons. One of the reason may be to simply validate parties involved in the communication [6].

Simple firewalls are generally not equipped with SSL inspection or offloading which allows encrypted traffic to pass without any inspection [7]. This allows malicious traffic inside the network over covert channels that is not inspected by the firewall [8]. There is a dire need to detect legitimate and illegitimate traffic with minimal network overhead and overall system cost. This will allow any scale organization to better govern their organizational policies.

Virtual Private Network(VPN) service may be used to hide the real traffic in the network which may be otherwise not allowed or may be monitored [9]. A user using VPN service connects to a VPN server using normal Transport Layer Security (TLS) connection outside the network. Once connected, requests the website or service from the server [10, 11]. The VPN server originates the request on behalf of the user to the server requested. The encrypted response is sent to the the user on already established channel, as a result the whole activity passes any filters on the network firewall.

Such techniques may be used by the users which aim to hide from or deceive the organization of their internet activity [10]. This paper proposes a novel technique to detect VPN traffic inside a network. The proposed technique extracts the network traffic features and classifies the traffic to indicate if the traffic is legitimate or not. Key features are extracted from the network traffic and are compared against the already identified features of traffic found to be illegitimate or VPN traffic.

The system is also able to classify the traffic which is not following the pattern of normal traffic or normal user activity and flags that particular traffic stream to be invalid. We tested our system against five well known freely available web based VPN service providers, the proposed system was able to classify all of them correctly. More traffic characterizing features may be added to identify more applications.

2 Related Work and Comparison

Multiple VPN services like TOR [12], Hotspot Shield and other services have unique fingerprints and not all the services can be distinguished using a similar criteria. Yamada et al. discuss a technique that uses statistical analysis on the encrypted traffic [13]. The scheme discussed, uses data size of network packets and performs timing analysis on the received packets to detect malicious traffic inside an encrypted channel. This technique is very useful for Web service providers to analyze the traffic coming to their servers and detect any malicious activity coming from outside the network.

A study on android based applications which use VPN services [14] to show that these VPN services may use third party trackers to track user behavior and some may be used to bypass android sand-boxed environment. Once a malware or virus is delivered to the device inside the network, the whole network is vulnerable to attacks [15].

VPN clients inside the network act as a proxy, which connect to the respective VPN server. Once the connection is established, the VPN service provider is able to change or eavesdrop on the information and network traffic as required [16,17]. This attracts many third party advertisement or tracking entities [18,19]. Any malicious entity can read, save and/or modify your request and the related information to and from the destined service.

VPN services can change the data as they are in control of incoming and outgoing traffic from network to device. VPN services are also able to perform TLS interception [20] by using their own certificates which is trusted locally by the system, for VPN service to work properly. This leads to a more potentially risky situation when the device connected contains sensitive data [14,21]. One of the counter measure to this issue is certificate pinning [14,22]. So, detecting such VPN services inside your network can save you huge losses in terms of the information lost.

Goh et al. [23] proposes a man in the middle approach to detect VPN traffic in the network. The article put forwards a solution that uses secret-sharing scheme which involves a massive key management overhead using Public Key Infrastructure(PKI) technique. The paper assumes that the traffic coming to the system is unencrypted and the data is available in plain form for the system to analyze and detect VPN traffic. This is achieved by using application layer proxy

which generates the copy of unencrypted traffic against each connection which is then sent to the system for further analysis. This technique approximately doubles the network traffic and computational resources of existing system while increasing the memory requirements to decrypt and re-encrypt the web traffic.

Another solution that uses Deep Packet Inspection technique [24] uses multiple sensors through out the network to get the unencrypted traffic from the end hosts and send it back to SNORT [25] based IDS to detect unusual behavior in traffic. It increases the overall network traffic because a sensor is to be installed on each network machine to be able to detect any unusual activity. Another technique is to copy the entire connection traffic and use pre-shared secret to analyze any malicious traffic [26].

To identify applications being run inside the network, network analysis is used extensively. The work discussed by He et al. [27] uses basic yet one of the most effective and used techniques in network traffic analysis for traffic classification. Based on five tuple connection classification, the technique uses connection characteristics like packets size, their inter-arrival time and the direction and order of the packets to identify the network signature of any android application. The scheme provides basic understanding of traffic classification. However, network traffic generated by web based VPN services will have no major difference or identifying characteristics, different to a standard HTTPS connection.

The use of unencrypted traffic to manage, analyze and categorize encrypted traffic is an exciting concept, discussed by Niu et al. [28]. The schemes uses labelled DNS based data set to identify malicious Command and Control traffic and labels the traffic as suspicious or normal. The concept provides a unique prospective to analyze the network traffic beyond five tuple/ current connection technique discussed previously [27]. The Table 1 provides basic attributes of already discussed techniques. The techniques discussed pave the path of our proposed scheme.

Table 1: Attributes of Related Techniques

| Research Techniques | Strengths | Limitations |
|---------------------------------|--|--|
| NIDS based technique [23] | 1- Complete architecture to handle encrypted traffic based Intrusion detection. 2- Protection against Remote access and evasion techniques. | 1- Multiple devices to be added in the network. 2- Increased bandwidth inside the network due to traffic duplication. |
| DNS based technique [28] | 1- Introduces the concept of DNS scoring and analysis. Helpful in detecting malicious CNC based on DNS. | 1- All CNC may not use only DNS based implementation. |
| Connection based technique [27] | 1- Five tuple based connection management. Helpful in identifying different protocol and application behavior. | 1- Traffic generated by HTTPS based VPN will generally look like standard HTTPS streams. |

Our proposed system analyzes DNS records to identify malicious or illegitimate VPN Server

names. Connection features are extracted using five tuple approach. Five tuple approach classifies each new connection by five attributes listed below:

- Source IP
- Destination IP
- Protocol (TCP/UDP)
- Source Port
- Destination Port

DNS based traffic analysis and connection management using five tuple techniques, our proposed system goes a step further to analyze HTTPS handshake. This is done to verify the server name used in the connection with the DNS activity the user has generated by his network activity. Using this novel approach of managing a connection by using the activity preceding the current connection, we are able to detect and identify VPN traffic inside the network.

3 Forensic Analysis of VPN Services Client

To detect the network activity of VPN services we carried out the forensic analysis of VPN services. For this purpose we choose top five freely available web based VPN services listed below:

- Tor Browser
- Hotspot Shield Free
- Browsec VPN
- ZenMate VPN
- Hoxx VPN

For each of these VPN services we analyzed the network traffic generated by their clients installed on a user PC. The initial analysis was performed using Wireshark [29] and Network Miner [30]. Detailed analysis of each VPN service is discussed below:

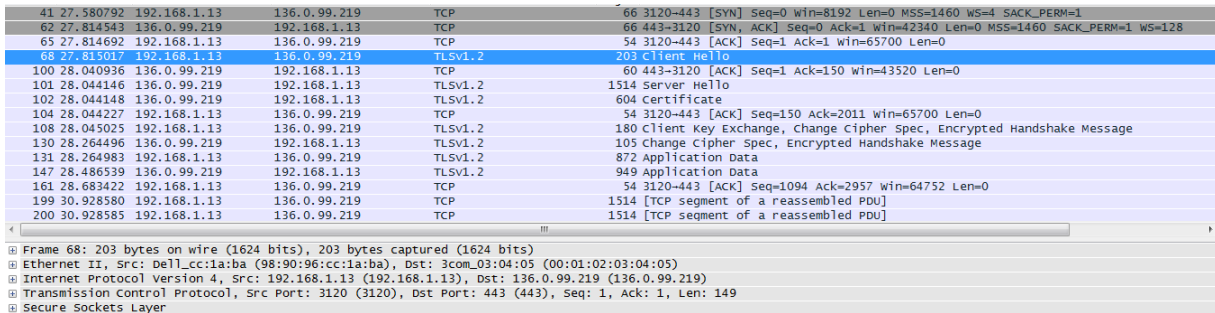
3.1 Hotspot Shield

Hotspot shield [31] developed by AnchorFree is one of the leading free VPN services used. We tested it's two versions:

- Client Application for Windows Desktop
- Firefox Add-on

3.1.1 Client Application for Windows Desktop

In client version of the above mentioned VPN service it was observed that once enabled, the service uses standard port 443 for HTTPS connections but generally connects to only one server. All the traffic may it be multi-site traffic uses the same active connection. Figure 1 shows the connection details for current user activity against Hotspot Shield. Hotspot shield uses fake well-known server name in SSL certificate to by pass the traffic from server name based filters over the network, if any, as shown in figure 2 below.



| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-----------|--------------|--------------|----------|--------|--|
| 41 | 27.580792 | 192.168.1.13 | 136.0.99.219 | TCP | 66 | 3120->443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1 |
| 62 | 27.814543 | 136.0.99.219 | 192.168.1.13 | TCP | 66 | 443->3120 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1460 SACK_PERM=1 WS=128 |
| 65 | 27.814692 | 192.168.1.13 | 136.0.99.219 | TCP | 54 | 3120->443 [ACK] Seq=1 Ack=1 Win=65700 Len=0 |
| 68 | 27.815017 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 203 | Client Hello |
| 100 | 28.040936 | 136.0.99.219 | 192.168.1.13 | TCP | 60 | 443->3120 [ACK] Seq=1 Ack=150 Win=43520 Len=0 |
| 101 | 28.044146 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 1514 | Server Hello |
| 102 | 28.044148 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 604 | Certificate |
| 104 | 28.044227 | 192.168.1.13 | 136.0.99.219 | TCP | 54 | 3120->443 [ACK] Seq=150 Ack=2011 Win=65700 Len=0 |
| 108 | 28.045025 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 180 | Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message |
| 130 | 28.264496 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 105 | Change Cipher Spec, Encrypted Handshake Message |
| 131 | 28.264983 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 872 | Application Data |
| 147 | 28.486539 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 949 | Application Data |
| 161 | 28.683422 | 192.168.1.13 | 136.0.99.219 | TCP | 54 | 3120->443 [ACK] Seq=1094 Ack=2957 Win=64752 Len=0 |
| 199 | 30.928580 | 192.168.1.13 | 136.0.99.219 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 200 | 30.928585 | 192.168.1.13 | 136.0.99.219 | TCP | 1514 | [TCP segment of a reassembled PDU] |

Figure 1: Wireshark: Hotspot Shield Client

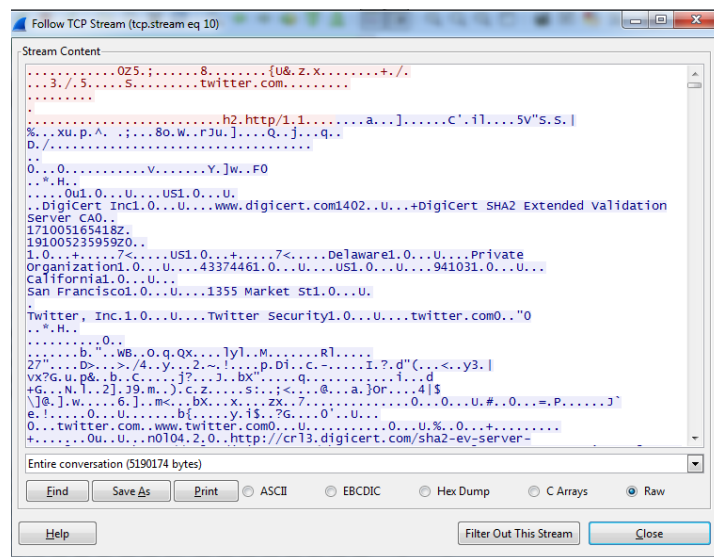


Figure 2: Wireshark: Hotspot Shield TCP Stream

It can be seen that used server name is twitter.com. It does not generate any DNS entry for such server name. The Network miner tool shows us the connection details in figure 3. We can see that eight unique connections were made, in this case it generally means eight unique web pages were open. Requests of all these web pages were managed by the server whose IP is 136.0.99.219. Certificate Details can also be seen against this server IP which were received. Total 20708 packets were sent in this activity and 11684 packets were received.

Figure 4 shows that no DNS activity for such host name was found during the communication. We can see all the DNS generated by the user while using Hotspot Shield Client.

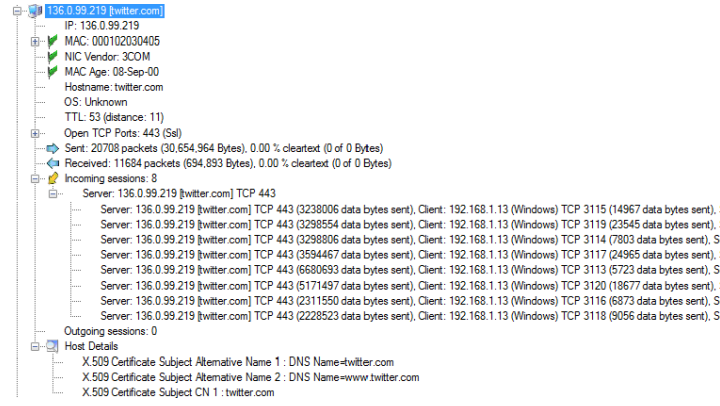


Figure 3: Network Miner: Hotspot Shield Connection Details

| Frame nr. | Timestamp | Client | Client Port | Server | Server Port | IP TTL | DNS TTL (time) | Transaction ID | Type | DNS Query | DNS Answer |
|-----------|-------------------------|--------|-------------|--------|-------------|--------|----------------|----------------|-----------------------|-----------------------------|----------------------|
| 3 | 2018-09-26 10:14:28 UTC | 192... | 49283 | 192... | 53 | 61 | 00:01:00 | 0x025A | 0x0005 (CNAME) | geo.hotspotshield.com | us.hotspotshield.com |
| 3 | 2018-09-26 10:14:28 UTC | 192... | 49283 | 192... | 53 | 61 | 00:01:00 | 0x025A | 0x0001 (Host Address) | us.hotspotshield.com | 74.115.0.53 |
| 32524 | 2018-09-26 10:15:31 UTC | 192... | 51671 | 192... | 53 | 61 | 00:01:00 | 0x0337 | 0x0001 (Host Address) | dh90c9110an8.cloudfront.net | 143.204.208.138 |
| 32524 | 2018-09-26 10:15:31 UTC | 192... | 51671 | 192... | 53 | 61 | 00:01:00 | 0x0337 | 0x0001 (Host Address) | dh90c9110an8.cloudfront.net | 143.204.208.6 |
| 32524 | 2018-09-26 10:15:31 UTC | 192... | 51671 | 192... | 53 | 61 | 00:01:00 | 0x0337 | 0x0001 (Host Address) | dh90c9110an8.cloudfront.net | 143.204.208.122 |
| 32524 | 2018-09-26 10:15:31 UTC | 192... | 51671 | 192... | 53 | 61 | 00:01:00 | 0x0337 | 0x0001 (Host Address) | dh90c9110an8.cloudfront.net | 143.204.208.156 |

Figure 4: Network Miner: No DNS information Found for 136.0.99.219

3.1.2 Firefox Add-on

Hotspot Shield in add-on uses standard https port along with standard DNS queries. The only way to detect hotspot shield inside the network is to identify the domain names used by hotspot shield. Shown below in the figure 5, is the network traffic generated by Hotspot shield captured using wireshark. It can be seen in the figure 6 shown below that the domain name is

| | | | | | |
|-----|-----------|--------------|--------------|---------|--|
| 41 | 27.580792 | 192.168.1.13 | 136.0.99.219 | TCP | 66 3120-443 [SYN, Seq=0 win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1 |
| 62 | 27.814543 | 136.0.99.219 | 192.168.1.13 | TCP | 66 443-3120 [SYN, ACK] Seq=0 Ack=1 win=42340 Len=0 MSS=1460 SACK_PERM=1 WS=128 |
| 65 | 27.814692 | 192.168.1.13 | 136.0.99.219 | TCP | 54 3120-443 [ACK] Seq=1 Ack=1 win=65700 Len=0 |
| 68 | 27.815017 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 203 Client Hello |
| 100 | 28.040936 | 136.0.99.219 | 192.168.1.13 | TCP | 60 443-3120 [ACK] Seq=1 Ack=150 win=43520 Len=0 |
| 101 | 28.044146 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 1514 Server Hello |
| 102 | 28.044148 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 604 Certificate |
| 104 | 28.044227 | 192.168.1.13 | 136.0.99.219 | TCP | 54 3120-443 [ACK] Seq=150 Ack=2011 win=65700 Len=0 |
| 108 | 28.045025 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 180 Client Key Exchange, change cipher Spec, Encrypted Handshake Message |
| 130 | 28.264496 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 105 Change Cipher Spec, Encrypted Handshake Message |
| 131 | 28.264983 | 192.168.1.13 | 136.0.99.219 | TLSv1.2 | 872 Application Data |
| 147 | 28.486539 | 136.0.99.219 | 192.168.1.13 | TLSv1.2 | 949 Application Data |
| 161 | 28.683422 | 192.168.1.13 | 136.0.99.219 | TCP | 54 3120-443 [ACK] Seq=1094 Ack=2957 win=64752 Len=0 |
| 199 | 30.928580 | 192.168.1.13 | 136.0.99.219 | TCP | 1514 [TCP segment of a reassembled PDU] |
| 200 | 30.928585 | 192.168.1.13 | 136.0.99.219 | TCP | 1514 [TCP segment of a reassembled PDU] |

Figure 5: Wireshark: Hotspot Shield Add-on

ext-mi-ex-nl-ams-pr-p-1.northghost.com for which the connection is established. We observed that hotspot shield domain name consists of two main parts:

- Server Identifier
- Domain Name

This can also be seen in certificate details in figure 7 below analyzed by Network Miner tool:

It is clearly observed that domain name is *.northghost.com and the other part is some server identifier as it may change once you re-initiate the connection. It can be seen that the connections for hotspot shield were established against only one server with IP address 216.162.47.67. Total

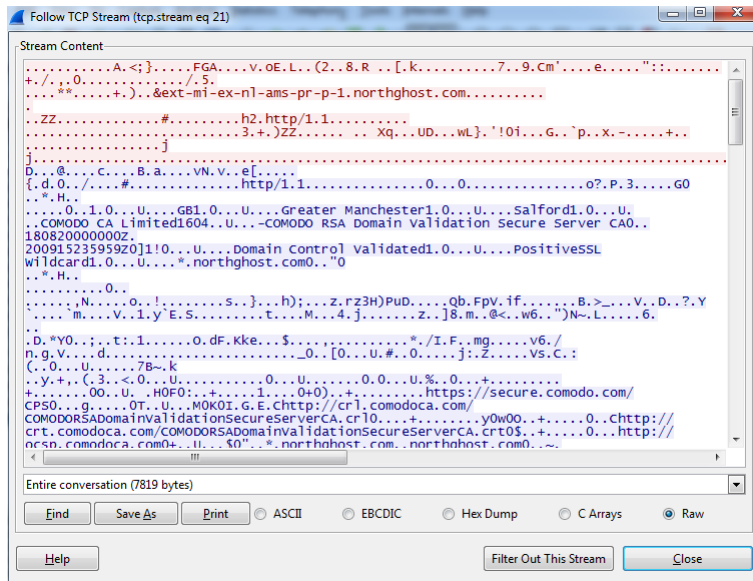


Figure 6: Wireshark: Hotspot Shield Add-on TCP Stream

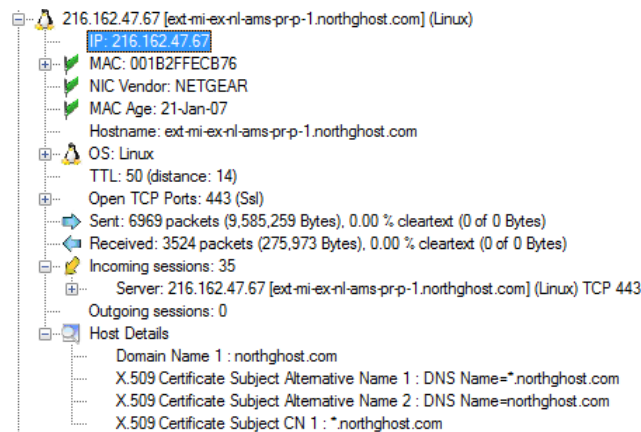


Figure 7: Network Miner: Hotspot Shield Add-on Connection Details

| | | | | | | | | | | | |
|-------|-------------------------|--------|-------|--------|----|----|----------|---------|-----------------------|--|-----------------------------|
| 2983 | 2019-05-13 07:13:26 UTC | 192... | 60552 | 192... | 53 | 64 | 00:04:17 | 0x35... | 0x0005 (CNAME) | fonts.gstatic.com | gstaticadsdl.google.com |
| 2983 | 2019-05-13 07:13:26 UTC | 192... | 60552 | 192... | 53 | 64 | 00:04:17 | 0x35... | 0x0001 (Host Address) | gstaticadsdl.google.com | 172.217.19.3 |
| 3201 | 2019-05-13 07:13:26 UTC | 192... | 62215 | 192... | 53 | 64 | 02:49:29 | 0x3A... | 0x0005 (CNAME) | www.google-analytics.com | www.google-analytics.com |
| 3201 | 2019-05-13 07:13:26 UTC | 192... | 62215 | 192... | 53 | 64 | 00:04:39 | 0x3A... | 0x0001 (Host Address) | www.google-analytics.com | 172.217.19.14 |
| 5200 | 2019-05-13 07:13:30 UTC | 192... | 61388 | 192... | 53 | 64 | 00:05:00 | 0xE9... | 0x0001 (Host Address) | ext-mi-ex-nl-ams-pr-p-1.northghost.com | 216.162.47.67 |
| 14297 | 2019-05-13 07:13:43 UTC | 192... | 59456 | 192... | 53 | 64 | 00:24:24 | 0x25... | 0x0005 (CNAME) | edge-chat.facebook.com | star.c10r.facebook.com |
| 14297 | 2019-05-13 07:13:43 UTC | 192... | 59456 | 192... | 53 | 64 | 00:00:07 | 0x25... | 0x0001 (Host Address) | star.c10r.facebook.com | 157.240.24.20 |
| 14927 | 2019-05-13 07:13:44 UTC | 192... | 51326 | 192... | 53 | 64 | 00:00:47 | 0xA2... | 0x0001 (Host Address) | star-mini.c10r.facebook.com | 157.240.24.35 |
| 14927 | 2019-05-13 07:13:44 UTC | 192... | 51326 | 192... | 53 | 64 | 00:04:26 | 0xA2... | 0x0005 (CNAME) | www.facebook.com | star-mini.c10r.facebook.com |

Figure 8: Network Miner: DNS information for 136.0.99.219

connections established were 35 and total 20708 packets were sent in this activity and 11684 packets were received.

The add-on also generates standard DNS activity as shown in figure 8.

Changing the VPN locations from add-on's option has no effect on the server being connected by the client as the server identifier in the same activity does not change.

3.2 ZenMate

Zenmate [32] developed by ZenGuard is also very popular free VPN service used. We analyzed the chrome based add-on of ZenMate. It uses standard https port along with standard DNS queries. The only way to detect Zenmate inside the network is to identify the domain names used by Zenmate VPN. Shown below in the figure 9, is the network traffic generated by Zenmate VPN captured using wireshark.

| | | | | | |
|-----|-----------|---------------|---------------|---------|--|
| 508 | 44.562205 | 192.168.100.3 | 193.176.86.50 | TCP | 66 3921-443 [SYN] Seq=0 Win=17520 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 509 | 44.723596 | 193.176.86.50 | 192.168.100.3 | TCP | 66 443-3921 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1412 SACK_PERM=1 WS=512 |
| 510 | 44.723755 | 192.168.100.3 | 193.176.86.50 | TCP | 54 3921-443 [ACK] Seq=1 Ack=1 Win=17408 Len=0 |
| 511 | 44.724659 | 192.168.100.3 | 193.176.86.50 | TLSv1.2 | 571 Client Hello |
| 513 | 44.884046 | 193.176.86.50 | 192.168.100.3 | TCP | 54 443-3921 [ACK] Seq=1 Ack=518 Win=29696 Len=0 |
| 514 | 44.896032 | 193.176.86.50 | 192.168.100.3 | TLSv1.2 | 1466 Server Hello |
| 515 | 44.897054 | 193.176.86.50 | 192.168.100.3 | TCP | 1466 [TCP segment of a reassembled PDU] |
| 516 | 44.897060 | 193.176.86.50 | 192.168.100.3 | TLSv1.2 | 1466 Certificate |
| 517 | 44.897079 | 193.176.86.50 | 192.168.100.3 | TLSv1.2 | 160 Server Key Exchange |
| 518 | 44.897184 | 192.168.100.3 | 193.176.86.50 | TCP | 54 3921-443 [ACK] Seq=518 Ack=4343 Win=17408 Len=0 |
| 519 | 44.915463 | 192.168.100.3 | 193.176.86.50 | TLSv1.2 | 180 Client Key Exchange, Change Cipher Spec, Hello Request, Hello Request |
| 523 | 45.074344 | 193.176.86.50 | 192.168.100.3 | TLSv1.2 | 296 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message |
| 524 | 45.078336 | 192.168.100.3 | 193.176.86.50 | TLSv1.2 | 308 Application Data |
| 532 | 45.246899 | 193.176.86.50 | 192.168.100.3 | TLSv1.2 | 102 Application Data |
| 533 | 45.248158 | 192.168.100.3 | 193.176.86.50 | TLSv1.2 | 660 Application Data |

Frame 511: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits) on interface 0
 Ethernet II, Src: IntelCor_77:5c:13 (34:e6:ad:77:5c:13), Dst: 48:8e:ef:b4:66:d7 (48:8e:ef:b4:66:d7)
 Internet Protocol Version 4, Src: 192.168.100.3 (192.168.100.3), Dst: 193.176.86.50 (193.176.86.50)
 Transmission Control Protocol, Src Port: 3921 (3921), Dst Port: 443 (443), Seq: 1, Ack: 1, Len: 517
 Secure Sockets Layer

Figure 9: Wireshark: ZenMate Add-on

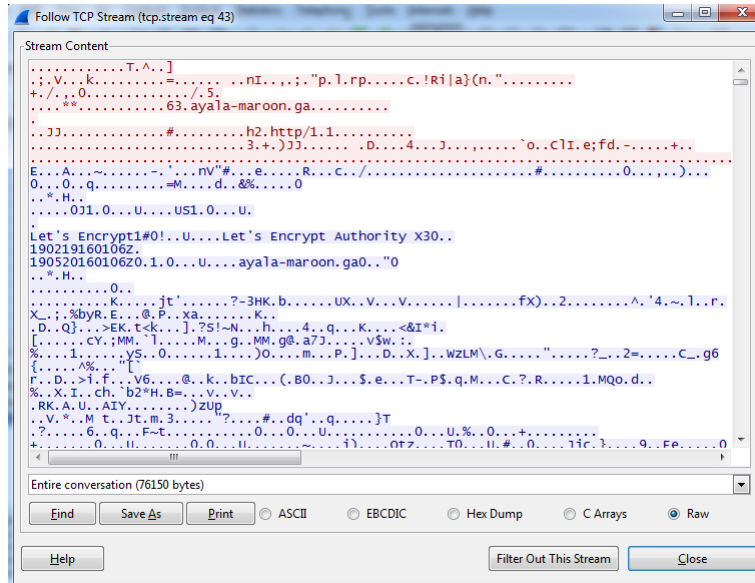


Figure 10: Wireshark: ZenMate Add-on TCP Stream

It can be seen in the figure 10 shown below that the domain name is 63.ayala-maroon.ga for which the connection is established.

Like hotspot shield, zenmate's domain name also consists of two main parts:

- Server Identifier
- Domain Name

This can also be seen in certificate details in figure 11 below analyzed by Network Miner tool:

It is clearly observed that domain name is *.ayala-maroon.ga and the number part is some server identifier. Zenmate is unique from other VPN service as it constantly changes the servers

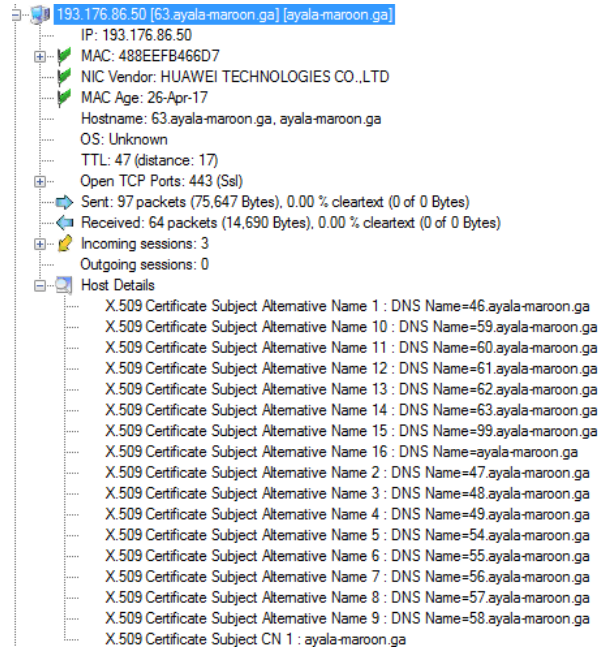


Figure 11: Network Miner: ZenMate VPN Add-on Connection Details

being connected by a user. So any suspicious or long activity with one server cannot be identified by automated tools. As seen in figure 11 multiple host names against the same domains are listed in SSL certificate provided by the VPN server. These servers/hosts may be used randomly to request multiple resources over the internet. It clearly shows in the figure that number of connections against this server are only five, which is less than other VPN server's connection discussed in the paper.

Another unique feature that zenmate offer is that, it changes the domain name as well once the location of the VPN server is changed from the settings of add-on. As shown in figure 12 the server name is changed to 34.lutz-obrien-olive.ga once the user has changed the location.

ZenMate changes domain names against region selected by the user but for same region the server identifier of domain name may change but domain remains the same. If a user is constantly changing the locations after some time when all locations available are exhausted the domains for each location could be identified. As shown is figure 13 multiple domains for Zenmate service used by this user were:

- lutz-obrien-olive.ga
- ayala-maroon.ga
- hall-silver.ga
- young-purple.ga

This information can now be used to prepare a filter to identify ZenMate VPN inside the network. One can also notice that last part of domain is always a color and ends with .ga. So, if we received DNS request or response and the domain name ends with .ga with "-"(dash) in the query it could be separated on "-". Once separated if last string contains any well known

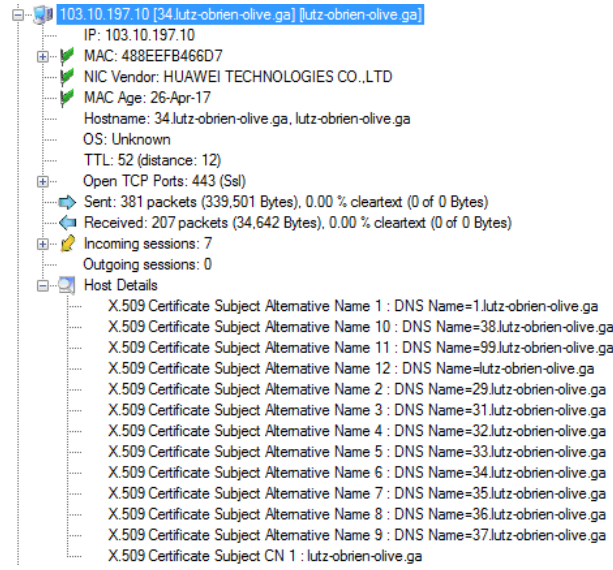


Figure 12: Network Miner: ZenMate VPN Add-on Connection Details - Changed Location

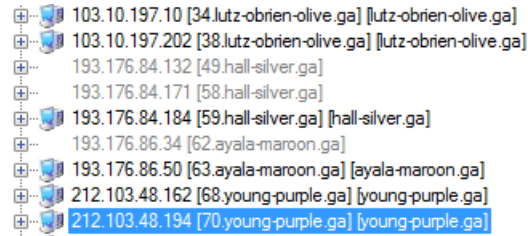


Figure 13: Network Miner: ZenMate VPN Add-on Connection Details - All Locations

color name we can classify it as ZenMate DNS server. As shown in figure 14 the domain name analysis done by Network Miner, we can see the same pattern discussed above.

| | | | | | | | | | | | |
|------|-------------------------|--------|-------|--------|----|----|----------|--------|-----------------------|-------------------------|----------------|
| 5438 | 2019-04-01 19:32:38 UTC | 192... | 53872 | 192... | 53 | 64 | 00:01:08 | 0x5F28 | 0x0001 (Host Address) | 34.lutz-obrien-olive.ga | 103.10.197.10 |
| 634 | 2019-04-01 19:30:33 UTC | 192... | 62279 | 192... | 53 | 64 | 00:00:19 | 0xC61C | 0x0001 (Host Address) | 38.lutz-obrien-olive.ga | 103.10.197.202 |
| 4705 | 2019-04-01 19:31:16 UTC | 192... | 63787 | 192... | 53 | 64 | 00:05:00 | 0x1FCC | 0x0001 (Host Address) | 49.hall-silver.ga | 193.176.84.132 |
| 4592 | 2019-04-01 19:31:00 UTC | 192... | 60082 | 192... | 53 | 64 | 00:05:00 | 0x743A | 0x0001 (Host Address) | 58.hall-silver.ga | 193.176.84.171 |
| 4980 | 2019-04-01 19:31:53 UTC | 192... | 55896 | 192... | 53 | 64 | 00:05:00 | 0xCE24 | 0x0001 (Host Address) | 59.hall-silver.ga | 193.176.84.184 |
| 356 | 2019-04-01 19:30:06 UTC | 192... | 61995 | 192... | 53 | 64 | 00:04:39 | 0xE773 | 0x0001 (Host Address) | 62.ayala-maroon.ga | 193.176.86.34 |
| 507 | 2019-04-01 19:30:30 UTC | 192... | 49346 | 192... | 53 | 64 | 00:04:21 | 0xF496 | 0x0001 (Host Address) | 63.ayala-maroon.ga | 193.176.86.50 |
| 5270 | 2019-04-01 19:32:18 UTC | 192... | 60309 | 192... | 53 | 64 | 00:03:43 | 0xC40C | 0x0001 (Host Address) | 68.young-purple.ga | 212.103.48.162 |
| 5153 | 2019-04-01 19:32:07 UTC | 192... | 65171 | 192... | 53 | 64 | 00:00:09 | 0x713C | 0x0001 (Host Address) | 70.young-purple.ga | 212.103.48.194 |

Figure 14: Network Miner: DNS information for ZenMate Servers

3.3 Tor Browser

Tor Browser [12] is used generally by users to hide their internet activity and to access resources on dark web. Tor browser uses a concept of onion routing to hide user's activity. We installed tor browser to analyze the network traffic generated by the browser. It uses a non-standard port for communication over internet. It uses HTTPS over 9001 TCP Port initially for circuit connection. After the circuit connection is established TOR may use 443 for normal internet or any other port as configured. Tor will generally not generate any DNS traffic. A normal TOR stream viewed in wireshark is shown in figure 15.

Opening of each web site may create new connection to server and server name along with

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-----------|------------|-------------|----------|--------|---|
| 850 | 9.961776 | 172.16.0.6 | 5.9.42.230 | TCP | 74 | 41744 → 9001 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |
| 851 | 10.111696 | 5.9.42.230 | 172.16.0.6 | TCP | 82 | 9001 → 41744 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 |
| 852 | 10.111732 | 172.16.0.6 | 5.9.42.230 | TCP | 66 | 41744 → 9001 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSv |
| 853 | 10.111875 | 172.16.0.6 | 5.9.42.230 | TLSv1.2 | 260 | Client Hello |
| 860 | 10.260446 | 5.9.42.230 | 172.16.0.6 | TCP | 74 | 9001 → 41744 [ACK] Seq=1 Ack=195 Win=30080 Len=0 |
| 861 | 10.264325 | 5.9.42.230 | 172.16.0.6 | TLSv1.2 | 1079 | Server Hello, Certificate, Server Key Exchange, Se |
| 862 | 10.264349 | 172.16.0.6 | 5.9.42.230 | TCP | 66 | 41744 → 9001 [ACK] Seq=195 Ack=1006 Win=32128 Len= |
| 863 | 10.265052 | 172.16.0.6 | 5.9.42.230 | TLSv1.2 | 192 | Client Key Exchange, Change Cipher Spec, Encrypted |
| 869 | 10.414827 | 5.9.42.230 | 172.16.0.6 | TLSv1.2 | 125 | Change Cipher Spec, Encrypted Handshake Message [T |
| 870 | 10.415039 | 172.16.0.6 | 5.9.42.230 | TLSv1.2 | 186 | Application Data |
| 871 | 10.507976 | 5.9.42.230 | 172.16.0.6 | TLSv1.2 | 879 | [TCP Previous Segment not captured], Ignored Unk |
| 872 | 10.507694 | 172.16.0.6 | 5.9.42.230 | TCP | 78 | [TCP Window Update] 41744 → 9001 [ACK] Seq=361 Ack |
| 873 | 10.508823 | 5.9.42.230 | 172.16.0.6 | TCP | 1522 | [TCP Out-Of-Order] 9001 → 41744 [ACK] Seq=1057 Ad |

<

> Frame 851: 82 bytes on wire (656 bits), 82 bytes captured (656 bits)
 > Ethernet II, Src: Netgear_fe8cb:76 (08:1b:2f:fe:cb:76), Dst: Tp-LinkT_1c:2a:63 (60:e3:27:1c:2a:63)
 > Internet Protocol Version 4, Src: 5.9.42.230, Dst: 172.16.0.6
 > Transmission Control Protocol, Src Port: 9001, Dst Port: 41744, Seq: 0, Ack: 1, Len: 0

Figure 15: Wireshark: Tor Traffic

their IP addresses are communicated to TOR browser during circuit establishment process and is encrypted. Figure 16 shows a TOR based TCP stream analyzed in wireshark.

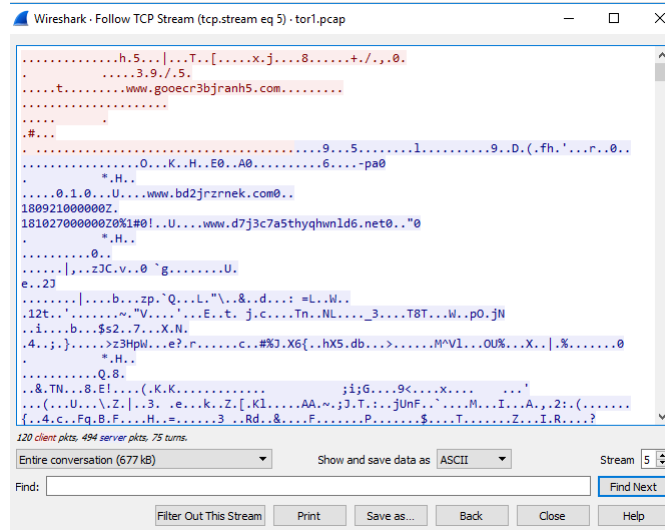


Figure 16: Wireshark: Tor Browser TCP Stream

| | | | | |
|----------------|-----------------------------------|------------------------------|-----------------------------|---------|
| 5.9.42.230 | [www.goocr3bjranh5.com] | [www.d73c7a5thyqhwld6.net] | [www.b6enz3fioufmp7twa.com] | (Linux) |
| 95.130.12.119 | [www.e6c3r3ntbd4sfrenypdt7o.com] | [www.j72ed2gr6vpzytcx.net] | | (Linux) |
| 109.236.90.209 | [www.77bahgmj.com] | [www.2u7hg4dwgcg2ykbdk5.net] | | |
| 164.132.77.175 | | | | |
| 172.16.0.1 | | | | |
| 172.16.0.6 | (Linux) | | | |
| 185.13.39.197 | [www.x5o244h62yix23cdfvcuhso.com] | [www.hbafmp5y3bdww.net] | | (Linux) |
| 195.228.75.149 | [www.lehip.com] | [www.uocck7z3eae.net] | | (Linux) |

Figure 17: Network Miner: TOR Browser User Activity Details

Connection details of a TOR connections analyzed by Network Miner are shown in figure 18. It shows that against server IP 5.9.42.230 total 639 packets were sent and 586 packets were received by the user. Complete activity of the user for the session being discussed is also shown in figure 17. It is interesting to mention here that no DNS activity was found for TOR browser.

3.4 Browsec VPN

Browsec VPN [33] is another freely available VPN. We used it as firefox add-on. It uses standard HTTPS port along with standard DNS queries. The only way to detect Browsec VPN inside

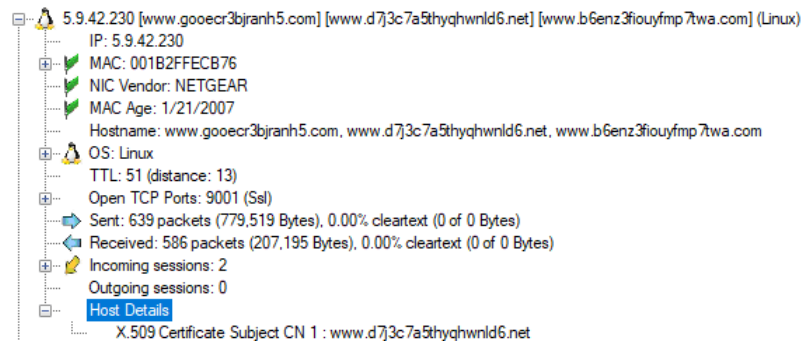


Figure 18: Network Miner: TOR Browser Connection Details

the network is to identify the domain names used by it. Shown below in the figure 19, is the network traffic generated by Browsec VPN captured using wireshark.

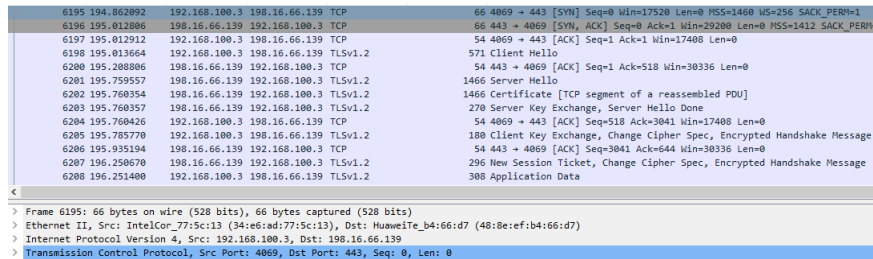


Figure 19: Wireshark: Browsec VPN Add-on

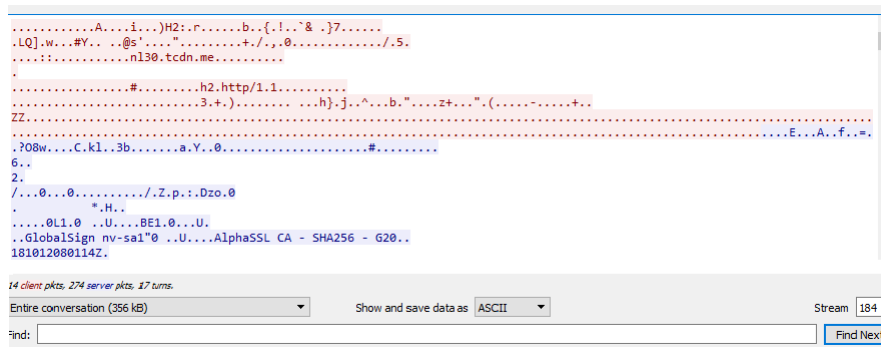


Figure 20: Wireshark: Browsec VPN TCP Stream

It can be seen in figure 20 that the domain name is nl30.tcdn.me for which the connection was established. Like other VPN services the domain name of Browsec VPN can also be further divided for better analysis. It consists of three main parts, it can also be seen in certificate details in figure 21 analyzed by Network Miner tool:

- Country Code
- Server Identifier
- Domain Name

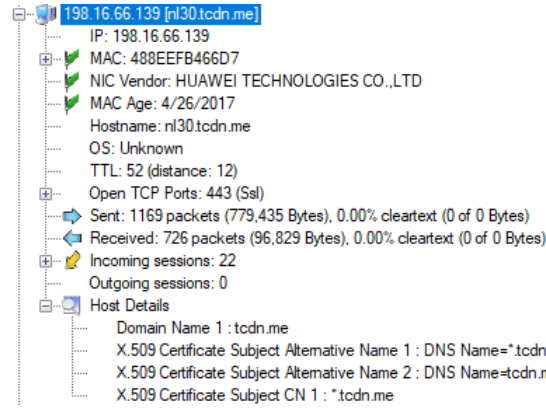


Figure 21: Network Miner: Browsec VPN Connection Details-NL



Figure 22: Network Miner: Browsec VPN Connection Details-UK

It is clearly observed that domain name is *.tcdn.me and the other part consists of some server identifier and location identifier. In figure 21 the location identifier is nl, which means Netherlands and in figure 22 we can see the country is United Kingdom.

Like Zenmate VPN, Browsec VPN also changes it's DNS information when changing the location but unlike zenmate the domain name is not changed rather only server qualifier is changed. The figure 23 shows the DNS traffic generated by user's activity.

3.5 Hoxx VPN

Hoxx VPN [34] is another freely available VPN. We used it as firefox add-on. It uses standard HTTPS port along with standard DNS queries. We can detect Hoxx VPN inside the network by identifying the domain names used by the VPN service. Shown below in the figure 24, is the

| Frame nr. | Timestamp | Client | Client Port | Ser... | Server Port | IP TTL | DNS TTL (time) | Transaction ID | Type | DNS Query | DNS Answer |
|-----------|-------------------------|--------|-------------|--------|-------------|--------|----------------|----------------|-----------------------|--------------|----------------|
| 6194 | 2019-04-01 19:33:01 UTC | 192... | 50050 | 192... | 53 | 64 | 17:24:48 | 0x3D78 | 0x0001 (Host Address) | nl30.tcdn.me | 198.16.66.139 |
| 7425 | 2019-04-01 19:33:15 UTC | 192... | 59982 | 192... | 53 | 64 | 17:20:27 | 0xDA3C | 0x0001 (Host Address) | nl10.tcdn.me | 198.16.66.123 |
| 8497 | 2019-04-01 19:33:39 UTC | 192... | 53446 | 192... | 53 | 64 | 11:18:17 | 0x7234 | 0x0001 (Host Address) | sg17.tcdn.me | 178.128.57.177 |
| 8770 | 2019-04-01 19:34:00 UTC | 192... | 61604 | 192... | 53 | 64 | 12:19:49 | 0x7957 | 0x0001 (Host Address) | sg25.tcdn.me | 178.128.117.77 |
| 8980 | 2019-04-01 19:34:02 UTC | 192... | 64325 | 192... | 53 | 64 | 17:14:48 | 0x173F | 0x0001 (Host Address) | uk1.tcdn.me | 178.62.34.82 |
| 9282 | 2019-04-01 19:34:23 UTC | 192... | 56276 | 192... | 53 | 64 | 17:19:22 | 0x1984 | 0x0001 (Host Address) | uk9.tcdn.me | 46.101.16.229 |
| 9762 | 2019-04-01 19:34:44 UTC | 192... | 49190 | 192... | 53 | 64 | 17:24:24 | 0x86B9 | 0x0001 (Host Address) | uk12.tcdn.me | 178.62.6.233 |

Figure 23: Network Miner: DNS information for Browsec VPN

network traffic generated by Hoxx VPN captured using wireshark.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-----------|---------------|---------------|----------|--------|--|
| 252 | 28.811041 | 192.168.1.2 | 149.28.168.15 | TCP | 66 | 9687 → 443 [SYN] Seq=0 Win=17520 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 260 | 29.010811 | 149.28.168.15 | 192.168.1.2 | TCP | 66 | 443 → 9687 [SYN, ACK] Seq=0 Ack=1 Win=29280 Len=0 MSS=1460 SACK_PERM=1 |
| 261 | 29.010891 | 192.168.1.2 | 149.28.168.15 | TCP | 54 | 9687 → 443 [ACK] Seq=1 Ack=1 Win=17488 Len=0 |
| 262 | 29.013443 | 192.168.1.2 | 149.28.168.15 | TLV1.2 | 571 | Client Hello |
| 266 | 29.214909 | 149.28.168.15 | 192.168.1.2 | TCP | 54 | 443 → 9687 [ACK] Seq=1 Ack=518 Win=30336 Len=0 |
| 267 | 29.218401 | 149.28.168.15 | 192.168.1.2 | TLV1.2 | 1514 | Server Hello |
| 268 | 29.219246 | 149.28.168.15 | 192.168.1.2 | TLV1.2 | 1514 | Certificate [TCP segment of a reassembled PDU] |
| 269 | 29.219250 | 149.28.168.15 | 192.168.1.2 | TLV1.2 | 137 | Server Key Exchange, Server Hello Done |
| 270 | 29.219319 | 192.168.1.2 | 149.28.168.15 | TCP | 54 | 9687 → 443 [ACK] Seq=518 Ack=3004 Win=17488 Len=0 |
| 271 | 29.232412 | 192.168.1.2 | 149.28.168.15 | TLV1.2 | 180 | Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message |
| 284 | 29.428043 | 149.28.168.15 | 192.168.1.2 | TLV1.2 | 105 | Change Cipher Spec, Encrypted Handshake Message |
| 287 | 29.439411 | 192.168.1.2 | 149.28.168.15 | TLV1.2 | 354 | Application Data |
| 289 | 29.645919 | 149.28.168.15 | 192.168.1.2 | TCP | 1514 | 443 → 9687 [ACK] Seq=3055 Ack=944 Win=31360 Len=1460 [TCP segment of a |

< Frame 252: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
 > Ethernet II, Src: IntelCor_77:5c:13 (34:e6:ad:77:5c:13), Dst: Netgear_fe:cb:76 (08:1b:2f:fe:cb:76)
 > Internet Protocol Version 4, Src: 192.168.1.2, Dst: 149.28.168.15
 > Transmission Control Protocol, Src Port: 9687, Dst Port: 443, Seq: 0, Len: 0

Figure 24: Wireshark: Hoxx VPN Add-on

| | | | | | | | | | | | |
|-------|-------------------------|---------|-------|---------|----|----|------------|--------|------------------|---|-----------------------------------|
| 250 | 2019-05-13 07:37:01 UTC | 192.... | 51103 | 192.... | 53 | 64 | 1.00:00:00 | 0xC6D8 | 0x0001 (Host ... | dyn-149-28-168-15-c0f8-1b377a.klalive.com | 149.28.168.15 |
| 251 | 2019-05-13 07:37:01 UTC | 192.... | 51103 | 192.... | 53 | 64 | 1.00:00:00 | 0xC6D8 | 0x0001 (Host ... | dyn-149-28-168-15-c0f8-1b377a.klalive.com | 149.28.168.15 |
| 254 | 2019-05-13 07:37:01 UTC | 192.... | 62867 | 192.... | 53 | 64 | 1.00:00:00 | 0xC6BD | 0x0001 (Host ... | dyn-149-28-168-15-c0f8-1b377a.klalive.com | 149.28.168.15 |
| 263 | 2019-05-13 07:37:01 UTC | 192.... | 61758 | 192.... | 53 | 64 | 00:00:00 | 0x621C | 0x0000 | dyn-149-28-168-15-c0f8-1b377a.klalive.com | No error condition (flags 0x8180) |
| 264 | 2019-05-13 07:37:01 UTC | 192.... | 61758 | 192.... | 53 | 64 | 00:00:00 | 0x621C | 0x0000 | dyn-149-28-168-15-c0f8-1b377a.klalive.com | No error condition (flags 0x8180) |
| 895 | 2019-05-13 07:37:14 UTC | 192.... | 64210 | 192.... | 53 | 64 | 00:00:00 | 0x6FAE | 0x0000 | dyn-149-28-168-15-c0f8-1b377a.klalive.com | No error condition (flags 0x8180) |
| 18562 | 2019-05-13 07:37:48 UTC | 192.... | 54745 | 192.... | 53 | 64 | 1.00:00:00 | 0xCE67 | 0x0001 (Host ... | dyn-146-185-141-219-5871-1b377a.klalive.com | 146.185.141.219 |
| 18563 | 2019-05-13 07:37:48 UTC | 192.... | 54745 | 192.... | 53 | 64 | 1.00:00:00 | 0xCE67 | 0x0001 (Host ... | dyn-146-185-141-219-5871-1b377a.klalive.com | 146.185.141.219 |
| 18572 | 2019-05-13 07:37:48 UTC | 192.... | 50164 | 192.... | 53 | 64 | 23:59:59 | 0x5EB6 | 0x0001 (Host ... | dyn-146-185-141-219-5871-1b377a.klalive.com | 146.185.141.219 |
| 19170 | 2019-05-13 07:37:49 UTC | 192.... | 55650 | 192.... | 53 | 64 | 00:00:00 | 0xCC57 | 0x0000 | dyn-146-185-141-219-5871-1b377a.klalive.com | No error condition (flags 0x8180) |
| 19173 | 2019-05-13 07:37:49 UTC | 192.... | 55650 | 192.... | 53 | 64 | 00:00:00 | 0xCC57 | 0x0000 | dyn-146-185-141-219-5871-1b377a.klalive.com | No error condition (flags 0x8180) |

Figure 25: Network Miner: DNS information for Hoxx VPN

It can be seen in the figure 26 that the domain name is dyn-146-185-141-219-5871-1b377a.klalive.com for which the connection is established. Like other VPN services the domain name of Hoxx VPN server can also be further divided for better analysis. It consists of two main parts:

- Server Identifier
- Domain Name

This division can also be seen in certificate details in figure 27 analyzed by Network Miner tool. It is clearly observed that domain name is *.klalive.com and the other part consists of some server identifier. The figure 25 shows the DNS traffic generated by user's activity.

4 Proposed System

The proposed system distinguishes the normal flow of an internet activity or session from an abnormal one. Normally, when a user wants to connect to a website a DNS request is made to translate the web name to IP address [35]. After successful name resolution, against the IP a TCP (Transmission Control Protocol) session is initiated and required security associations are established. This behavior may be used to monitor and analyze different feature of network traffic. [36–38].

The proposed system classifies any incoming data into multiple categories depending on the current state of connection, in addition to that, internet activity preceding the connection is also monitored to identify the traffic as VPN or simple internet traffic. The process of detecting any illegitimate traffic is further classified into two main processes:

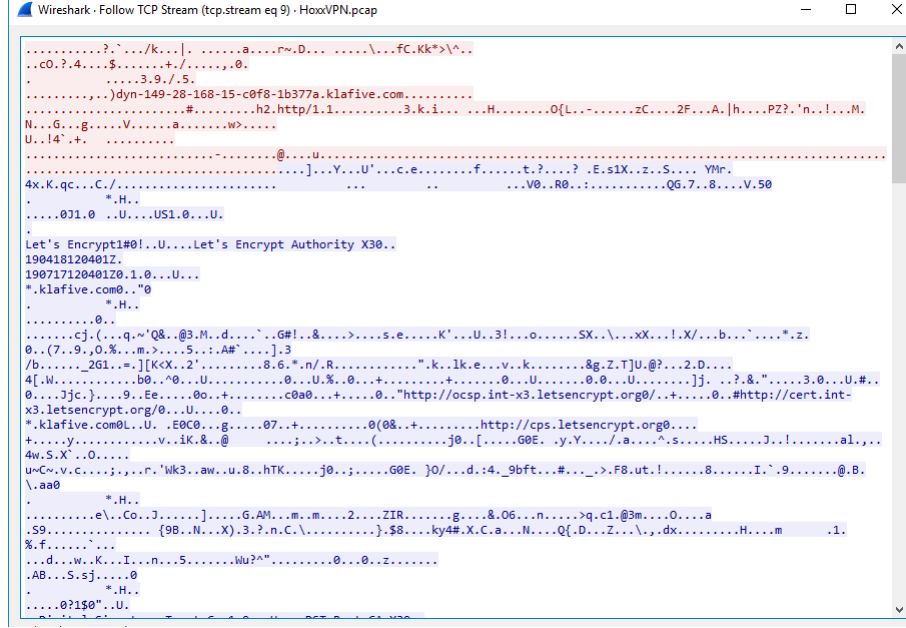


Figure 26: Wireshark: Hoxx VPN TCP Stream

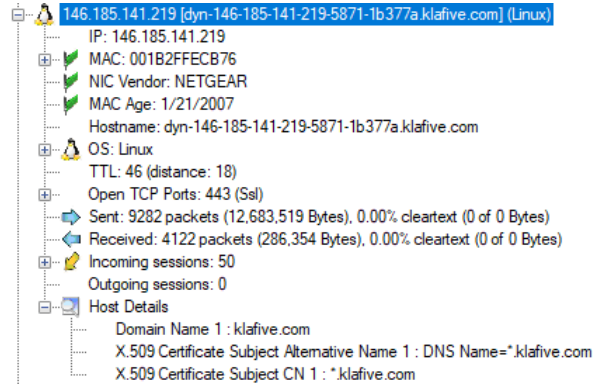


Figure 27: Network Miner: Hoxx VPN Connection Details

- Feature Extraction
- Traffic Classification

4.1 Feature Extraction

To classify traffic as normal or VPN we have to extract different traits of the network traffic. Now most of these traits can be found in current traffic stream while some of them are collected before the actual stream starts. Figure 28 shows the basic flow of network traffic feature extraction module of the system. The analyzer extracts following information to be used for traffic categorization

- **Basic Feature Extraction** Server IP of the server and user is extracted at the first step. This information is extracted from IPv4 Protocol fields Source IP and Destination IP [39]. Depending upon the transport layer protocol the source port and destination ports are also extracted [40].

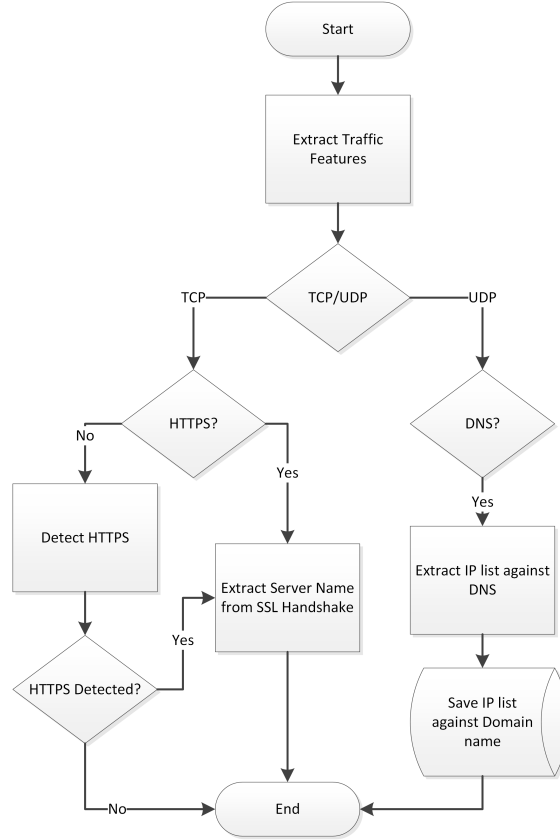


Figure 28: Feature Extraction

- **Domain Name Server Analysis** Unencrypted traffic information is as important in traffic characterization and behavior analysis of users as the encrypted traffic. For any web request, generated by a user, a DNS request is initiated by the user's browser to request the IP information of the server name. A response is sent to the user from DNS server containing IP information of the server [35]. This information is stored by our system to verify the DNS server name vs HTTPS certificate's server name to see for any inconsistencies.
- **HTTPS Protocol Detection** Incoming traffic is then passed to HTTPS Detection Module. System looks for HTTPS other than port 443. This is done by looking for HTTPS headers on streams which are TCP based connections but server port number is other than 443. A lot of applications and services use the technique to change the server port. This allows them to pass through network firewall and are not labelled as encrypted payload.
- **SSL Analysis** The proposed system decodes SSL certificates [41] once HTTPS is detected. There are 4 basic type of messages in SSL:
 - Handshake
 - Change Cipher Spec
 - Application Data
 - Alert

From the Handshake messages we extract the server information such as name of the server to which the connection is made. This is used to verify or detect the DNS activity versus server name.

These features once extracted are used by traffic classifier to classify each connection to VPN or normal traffic.

4.2 Traffic Classification

After features are extracted we can classify the incoming traffic as normal traffic or VPN traffic only for the TCP based connections. TCP connection states are stored for every new connection. Once the connection is established, it is classified as legitimate or VPN traffic based on extracted features of previous network traffic and new connection. This classification may be as legitimate traffic or VPN traffic. Proposed scheme classifies the incoming connections as shown in Figure 29 and discussed below.

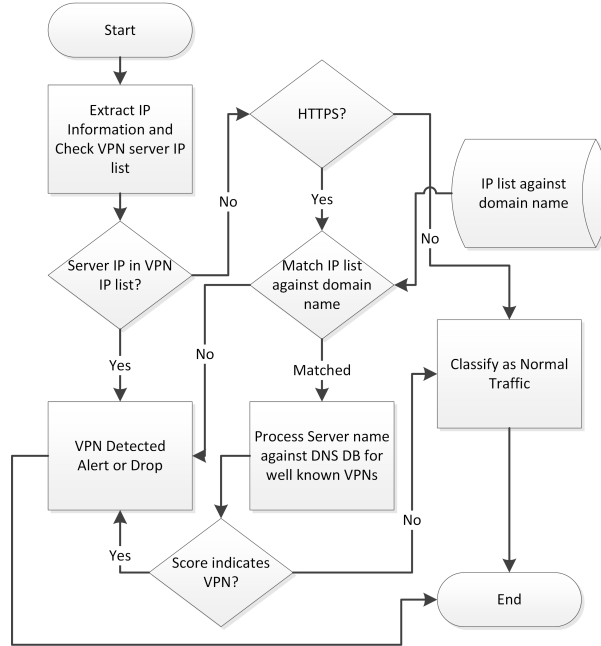


Figure 29: Traffic Classification

- **IP Based Classification** Server IP of each new connection is looked up in an already populated IP based Hash table. This hash table contains the IP list of Tor's [12] exit nodes along with the server IP that were previously classified by the system as VPN servers. This is done to minimize the resource utilization against already classified VPN Server. If server IP of the current connection is found in this IP based hash then the traffic is classified as VPN traffic.
- **Server Name Based Classification** If the connection is not classified by VPN IP based hash table, the server name specified in HTTPS Client Hello message is used to classify the connection. In a normal TCP/IP based communication whenever a service or website

needs to be accessed, first its domain name is converted into IP address. This is done as to access the resources over the internet [42]. An IP address at a given time is bound to specific domain. Using this technique we classify the normal domains against the domains responsible for VPN Services. This classification can be further divided in two steps:

- **No Server Name Analysis** Against the current server name extracted from the connection we lookup our self-maintained DNS list, populated by network traffic. If no DNS entry is present for that server name in the list or the server IP of the connection is not associated against the given server name, such traffic is classified as VPN traffic. Mostly inside the initial connection to VPN server, these IPs against DNS are shared with the client’s application in SSL protected channel as to avoid any DNS based filtering.
- **Server Name Analysis** The server name or the domain name of the current connection is looked up against the well-known VPN server’s domain names. The list is maintained to lookup the server name, if found the connection is classified as VPN based connection. The list is generated by the traffic analysis of these VPN servers and some unique strings are extracted specific to that VPN service as discussed previously in section 3.

5 System Evaluation

The deployment of our proposed solution if used only for detection can be passive as well. Passive deployment will result in lower latency as the traffic is being mirrored by the switch or gateway itself. Provided all the traffic destined for outside the network and DNS traffic passes through the tapped interface as shown in figure 30.

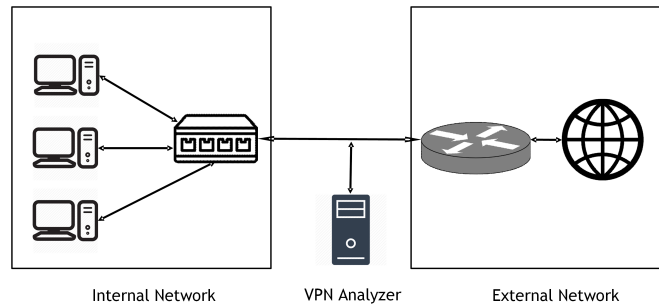


Figure 30: Deployment Model

We analyzed the traffic pattern of well known available VPN services which use HTTPS protocol for communication. These servers are listed below:

- Tor Browser
- Hotspot Shield Free
- Browsec VPN

- ZenMate VPN
- Hoxx VPN

Table 2: Forensic Analysis of freely Available VPN Services

| VPN Services | Classifiers for Forensic Analysis | | | |
|---------------------|-----------------------------------|-----------|--------------------|--------------|
| | IP | Host Name | Non-Standard HTTPS | DNS Activity |
| TOR Browser | ✓ | × | ✓ | ✓ |
| Hotspot Shield Free | × | ✓ | ✓ | ✓ |
| Browsec VPN | × | ✓ | × | × |
| ZenMate VPN | × | ✓ | × | × |
| Hoxx VPN | × | ✓ | × | × |

The traffic of these VPN services was analyzed and a selection criteria was build based on the pattern emerging from the analysis. The key features for each VPN service are shown in Table 2. In case of TOR we see non-standard HTTPS behavior which means that it may not be on default port 443. We can also detect TOR by TOR nodes list populated and updated by community.

In case of Hotspot Shield we tested two variants of its client. One was the add-on of Firefox web browser and the other client was desktop application. In case of web browser extension or add-on, Hotspot Shield uses special Domain names which are used to uniquely classify the service. In case of desktop application the client uses non-standard port for HTTPS with no DNS activity. Browsec and Hoxx VPNs both were tested as add on to the browser and they are uniquely classified using the domain names the servers use.

All three services discussed above use same type of domain names across multiple geo-locations e.g. any traffic may be classified as traffic of Hoxx VPN if its domain name contains *.klafive.com. This is not the case for ZenMate VPN. It changes domain names with respect to geo locations chosen by the user. The list of these domain names is communicated during initial connection setup and is updated frequently. This allows VPN services like ZenMate and others to work over a network which uses DNS based filters, if these filters are not updated frequently.

5.1 Traffic Generation

Across multiple systems inside the network multiple clients of the above mentioned VPN services were installed and configured. These clients were enabled and network activity was generated by surfing the internet. The activity was monitored by VPN Detector and alerts were generated once the VPN activity was detected.

5.2 Traffic Classification Alert

The alerts generated above for different VPN services were of different types depending upon the activities performed by the users. The generated alerts by five of these users are shown in Table 3

Table 3: Alerts Generated for the user activity

| User Details | Alerts Classification (Connection Based) | | | | |
|--------------|--|---------------------|--------------|---------------|--------|
| | Total | Legitimate Activity | IP based VPN | DNS based VPN | NO DNS |
| User 1 | 178 | 59 | 4 | 109 | 6 |
| User 2 | 85 | 50 | 0 | 35 | 0 |
| User 3 | 250 | 114 | 0 | 135 | 1 |
| User 4 | 71 | 24 | 2 | 41 | 4 |
| User 5 | 145 | 82 | 0 | 63 | 0 |

The alerts shown in table 3 shows the traffic classification of each type of VPN service used with respect to it's unique characteristics as discussed in Table 2. Mostly VPNs may be classified with the help of DNS activity which enable the user to access such services.

The results shown in table 3, show that the system classified 400 out of 729 active connections as potential VPN connections. Once the system is deployed, any new connection activity in the network is monitored. Each system connected to internet manages it's on DNS cache to reuse DNS information. If a new connection is made and no DNS activity is present in the system for the server, the system will flag it as potential VPN traffic. To improve system's precision, the system ignores the already established connections.

VPN classification based on IP and DNS activity may need periodic updates to the lists maintained by the system. Updating this information will increase the overall accuracy of the system and result in less false positives and negatives. Our test shows that in case of TOR IP analysis the IP information should be populated in real time to get better results.

6 Conclusion

A VPN service inside an organization may generally be used by an individual to hide the real communication. This communication may be harmful or damaging for the organization and organization may not allow such communication over it's monitored network. An organization may not be able to invest heavily on SSL based proxies to manage its network. This paper proposes a light weight approach to detect and block unwanted VPN clients inside the organizational network responsible for some illegitimate activity.

Our proposed technique focuses on the information available in plain, which means there is no need to decrypt or decode any network communication. This helps in low resource utilization. The proposed solution not only focuses on the current connection but also keeps track of the network activity responsible for this communication i.e. DNS activity. Such mapping of DNS with its next stream helps identify the normal behavior of the TCP/IP network stack. If no Domain Name information is available for current connection it may not be normal traffic flow. The scheme also analyzes non-standard use of HTTPS and detect this anomaly as it is largely used to hide such communication from HTTPS based filters in firewall.

Results show that our proposed system is able to identify and classify such trends in network traffic and classify the network traffic. The analysis of the VPN services discussed in

table 2 is crucial to detect these services. These service providers keep changing the traffic characteristics for their service. Active analysis of these services must be carried out to keep VPN Detector up to date with latest traffic trends.

Conflict of Interest Declaration

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

- [1] Kishan Neupane, Rami Haddad, and Lei Chen. Next generation firewall for network security: A survey. In SoutheastCon 2018, pages 1–6. IEEE, 2018.
- [2] B Harris and R Hunt. Tcp/ip security threats and attack methods. Computer Communications, 22:885–897, 06 1999.
- [3] Xin Li, Minmei Wang, Huazhe Wang, Ye Yu, and Chen Qian. Toward secure and efficient communication for the internet of things. IEEE/ACM Transactions on Networking, 2019.
- [4] Eric Rescorla. SSL and TLS: designing and building secure systems, volume 1. Addison-Wesley Reading, 2001.
- [5] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. Measuring HTTPS adoption on the web. In 26th USENIX Security Symposium (USENIX Security 17), pages 1323–1338, Vancouver, BC, 2017. USENIX Association.
- [6] Jeremy Clark and Paul C Van Oorschot. Sok: Ssl and https: Revisiting past challenges and evaluating certificate trust model enhancements. In 2013 IEEE Symposium on Security and Privacy, pages 511–525. IEEE, 2013.
- [7] Cem Paya and Opher Dubrovsky. Inspecting encrypted communications with end-to-end integrity, July 14 2009. US Patent 7,562,211.
- [8] Vladimir Lifliand and Avraham Michael Ben-Menahem. Encrypted network traffic interception and inspection, November 5 2013. US Patent 8,578,486.
- [9] Neal Leavitt. Anonymization technology takes a high profile. Computer, 42(11):15–18, 2009.
- [10] Zhang Zhipeng, Sonali Chandel, Sun Jingyao, Yan Shilin, Yu Yunnan, and Zang Jingji. Vpn: a boon or trap?: A comparative study of mpls, ipsec, and ssl virtual private networks. In 2018 Second International Conference on Computing Methodologies and Communication (ICCMC), pages 510–515. IEEE, 2018.
- [11] K Karuna Jyothi and B Indira Reddy. Study on virtual private network (vpn), vpn’s protocols and security. 2018.

- [12] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.
- [13] A. Yamada, Y. Miyake, K. Takemori, A. Studer, and A. Perrig. Intrusion detection for encrypted web accesses. In 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), volume 1, pages 569–576, May 2007.
- [14] Muhammad Ikram, Narseo Vallina-Rodriguez, Suranga Seneviratne, Mohamed Ali Kaafar, and Vern Paxson. An analysis of the privacy and security risks of android vpn permission-enabled apps. In Proceedings of the 2016 Internet Measurement Conference, pages 349–364. ACM, 2016.
- [15] Suhizaz Sudin, R Badlishah Ahmad, and Syed Zulkarnain Syed Idrus. A model of virus infection dynamics in mobile personal area network. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 10(2-4):197–201, 2018.
- [16] Nicholas Weaver, Christian Kreibich, Martin Dam, and Vern Paxson. Here be web proxies. In International Conference on Passive and Active Network Measurement, pages 183–192. Springer, 2014.
- [17] Charles Reis, Steven D Gribble, Tadayoshi Kohno, and Nicholas C Weaver. Detecting in-flight page changes with web tripwires. In NSDI, volume 8, pages 31–44, 2008.
- [18] Narseo Vallina-Rodriguez, Srikanth Sundaresan, Christian Kreibich, and Vern Paxson. Header enrichment or isp enrichment?: Emerging privacy threats in mobile networks. In Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization, pages 25–30. ACM, 2015.
- [19] Nicholas Weaver, Christian Kreibich, and Vern Paxson. Redirecting dns for ads and profit. FOCI, 2:2–3, 2011.
- [20] Narseo Vallina-Rodriguez, Johanna Amann, Christian Kreibich, Nicholas Weaver, and Vern Paxson. A tangled mass: The android root certificate stores. In Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies, pages 141–148. ACM, 2014.
- [21] Yihang Song and Urs Hengartner. Privacyguard: A vpn-based platform to detect information leakage on android devices. In Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices, pages 15–26. ACM, 2015.
- [22] Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. Why eve and mallory love android: An analysis of android ssl (in) security. In Proceedings of the 2012 ACM conference on Computer and communications security, pages 50–61. ACM, 2012.
- [23] V. T. Goh, J. Zimmermann, and M. Looi. Towards intrusion detection for encrypted networks. In 2009 International Conference on Availability, Reliability and Security, pages 540–545, March 2009.

- [24] A. A. Abimbola, J. M. Munoz, and W. J. Buchanan. Nethost-sensor: Investigating the capture of end-to-end encrypted intrusive data. Comput. Secur., 25(6):445–451, September 2006.
- [25] Martin Roesch. Snort - lightweight intrusion detection for networks. In Proceedings of the 13th USENIX Conference on System Administration, LISA '99, pages 229–238, Berkeley, CA, USA, 1999. USENIX Association.
- [26] Xiaoning Li, Karanvir S Grewal, Geoffrey H Cooper, and John R Guzik. Encrypted data inspection in a network environment, November 3 2015. US Patent 9,176,838.
- [27] Gaofeng He, Bingfeng Xu, and Haiting Zhu. Appfa: A novel approach to detect malicious android applications on the network. Security and Communication Networks, 2018, 2018.
- [28] Weina Niu, Xiaosong Zhang, GuoWu Yang, Jianan Zhu, and Zhongwei Ren. Identifying apt malware domain based on mobile dns logging. Mathematical Problems in Engineering, 2017, 2017.
- [29] Anish Nath. Packet Analysis with Wireshark. Packt Publishing Ltd, 2015.
- [30] Netresec. Network miner.
- [31] AnchorFree. Hoptspot shield vpn.
- [32] ZenGuard. Zenmate vpn.
- [33] Browsec LLC. Browsec vpn your personal privacy and security online.
- [34] VPN1.com. Lightning fast vpn service | hoxx vpn | vpn service for everyone.
- [35] Paul V Mockapetris. Domain names: Implementation specification. Technical report, 1983.
- [36] Luca Deri, Rocco Carbone, and Stefano Suin. Monitoring networks using ntop. In 2001 IEEE/IFIP International Symposium on Integrated Network Management Proceedings. Integrated Network Management VII. Integrated Management Strategies for the New Millennium (Cat. No. 01EX470), pages 199–212. IEEE, 2001.
- [37] Paul Barford, Jeffery Kline, David Plonka, and Amos Ron. A signal analysis of network traffic anomalies. In Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment, pages 71–82. ACM, 2002.
- [38] Mohammed Abdul Qadeer, Arshad Iqbal, Mohammad Zahid, and Misbahur Rahman Siddiqui. Network traffic analysis and intrusion detection using packet sniffer. In 2010 Second International Conference on Communication Software and Networks, pages 313–317. IEEE, 2010.
- [39] Jon Postel. Internet protocol. Technical report, DARPA, 1981.
- [40] Jon Postel. Transmission control protocol. Technical report, DARPA, 1981.

- [41] Tim Dierks and Eric Rescorla. The transport layer security (tls) protocol version 1.2. Technical report, 2008.
- [42] Behrouz A. Forouzan. TCP/IP Protocol Suite. McGraw-Hill, Inc., New York, NY, USA, 2 edition, 2002.