TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Software Science

Jessica Ai Truong 184564IVCM

EVALUATING THE DETECTION ACCURACY OF JA3 AND JA3S IN SECURITY MONITORING OF SSL COMMUNICATION

Master's Thesis

Supervisor: Hayretdin Bahsi
PhD

Tallinn 2019

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Jessica Ai Truong 184564IVCM

JA3 JA JA3S-I AVASTAMISE TÄPSUSE HINDAMINE TURVALISUSE JÄLGIMISEL SSL SUHTLUSES

Magistritöö

Juhendaja: Hayretdin Bahsi
PhD

Tallinn 2019

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Jessica Ai Truong

13.05.19

Abstract

Cybercriminals use encrypted communications such as SSL to hide their malicious activities. One of the issues in security monitoring is that the communication between an SSL client and server is encrypted, and to resolve this issue, TLS fingerprinting has been introduced. TLS fingerprinting in security monitoring is used to identify malicious communication without needing to decrypt the data or rely on the destination IP address. Reliability and usability of TLS fingerprinting remain an issue in security monitoring. This research concentrated on which would be more meaningful for security monitoring: utilizing the SSL fingerprint hashes or utilizing the original values of the fields that are used to generate these SSL fingerprints. It also focused on the detection of malicious activity in encrypted traffic. One way to do this is by using JA3 and JA3S which are fingerprints generated for the SSL client and server applications. JA3 was tested in real-world traffic with data taken from abuse.ch and Tallinn University of Technology. Malicious and normal traffic datasets were fed into Suricata and Bro to analyze the SSL communications and measure the false positives and false negative. A decision tree algorithm was used to calculate the accuracy of classification based on the dataset given. This study demonstrated that JA3 fingerprinting would not be useful in security monitoring but combining JA3-JA3S showed improvements in the accuracy of the detection. Machine learning was used as an instrument to demonstrate that sharing the original values of the fields would be more useful than hashing in security monitoring.

This thesis is written in English and is 68 pages long, including 6 chapters, 15 figures and 19 tables.

Annotatsioon

JA3 JA JA3S-I AVASTAMISE TÄPSUSE HINDAMINE TURVALISUSE JÄLGIMISEL SSL SUHTLUSES

Küberkurjadegijad kasutavad krüpteeritud suhtlust nagu SSL, et varjata oma pahatahtlikke tegevusi. SSL kliendi ja serveri vahelise suhtluse krüpteerimine tekitab probleeme turvalisuse jälgimisele. Selle probleemi lahendamiseks on kasutusele võetud TLS muster (*TLS fingerprinting*). Turvalisuse jälgimiseks kasutatakse TLS mustrit, et avastatada pahatahtlikku suhtlust, ilma seda lahti krüpteerimata. Samuti ei ole vaja teada sihtkoha IP aadressi. TLS mustri usaldusväärsus ja kasutatavus on probleemiks turvalisuse jälgimisel. Selle magistritööga sooviti välja selgitada, millist meetodit oleks kasulikum turvalisuse jälgimiseks kasutada: kas SSL mustri (*SSL fingerprint*) räside või SSL mustri loomiseks kasutatud väljade originaalväärtuste utiliseerimist ja jagamist. Samuti keskenduti antud töös krüpteeritud keskkonnast pahatahtliku tegevuse avastamisele. Üheks viisiks, kuidas seda teha, on kasutada JA3 ja JA3S-i. JA3-e testiti päriselu liiklusel. Andmed selleks saadi abuse.ch lehelt ja Tallinna Tehnikaülikoolilt. Pahatahtliku ja normaalse liikluse andmekogumeid töödeldi Suricata ja Bro-ga, et analüüsida SSL suhtlusi ja mõõta vale-positiivseid ning vale-negatiivseid vasteid. Andmekogumil kasutati otsustuspuu algoritmi, et arvutada kasutatud klassifitseerimise täpsus. Töö käigus leiti, et turvalisuse jälgimiseks ei ole JA3 mustri kasutamine üksi kasulik aga kombineerides JA3 ja JA3S oli pahatahtliku suhtluse avastamine täpsem. Masinõppega demonstreeriti, et turvalisuse jälgimiseks on väljade originaalväärtuste jagamine kasulikum kui räside kasutamine.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 68 leheküljel, 6 peatükki, 15 joonist, 19 tabelit.

List of abbreviations and terms

IoC	Indicator of Compromise
IDS	Intrusion Detection System
IP	Internet Protocol
SSL	Secure Sockets Layer
SID	Signature ID
SNI	Server Name Indication
TalTech	Tallinn University of Technology
TLS	Transport Layer Security

Table of Contents

List of Figures

List of Tables

1 Introduction

Most organizations use encryption and specifically Secure Socket Layer (SSL) and transport layer security (TLS) to protect their data [1]. Although encryption is a good security mechanism, it also presents many challenges to be able to monitor traffic and detect any threats. This is a growing concern because most of the security devices cannot inspect the encrypted data without severely impacting network performance. One issue with the encryption used by HTTPS is that it brings difficulty for an IDS to be able to examine the network traffic without decrypting it [2]. Of course, a solution would be to use a web proxy which will intercept the HTTPs connection and allow it to see the original data that was encrypted. However, this solution decreases the security level of communications and it also requires the proxy devices to have a higher performance [2]. This is actually a common approach used to detect malicious activity but is one that may not be beneficial in the future. It is important to note that when identifying threats in encrypted network traffic not to compromise the integrity of the encrypted data.

Encryption is a well-known method that cybercriminals use to hide their activity in the network traffic. Since cybercriminals are aware of the growth of encryption, they use it to their advantage to hide their existence and avoid detection whether that means transporting malware or exfiltrating data [1]. Cybercriminals are finding new tools and techniques they can use to make it difficult for analysts to detect malicious behaviour. They always want to be one step ahead to obfuscate their data. There are many kinds of research such as [3] and [4] that seek to detect encrypted data by using intrusion detection system (IDS). However, in recent years, the interest has shifted to focus more on the client and server side itself and not on the data that is being sent over the network. Studies have shown that it is especially important and beneficial to look at the SSL ClientHello packet fields as it will give information about the client application. The same can be seen from the SSL ServerHello packet. By understanding the fields of the client and server an assumption can be made about whether the traffic is malicious or normal. This technique may be useful in security monitoring by utilizing these fields to prevent malicious activity from happening immediately. Once a better understanding

of how these fields will help detect malicious traffic from normal traffic, it will aid in finding a solution to be able to monitor encrypted traffic and have high performance of security devices at all times.

1.1 Motivation

It has become common for malware to be located in encrypted traffic making it difficult for network administrators to be able to detect them. Cybercriminals are using this to their advantage to avoid being detected and to proceed with their malicious activities. Malware can be used to exfiltrate sensitive information to a cybercriminal's command and control server. Malware could also be a botnet waiting for the instructions from the attacker. Encrypted traffic has allowed Cybercriminals to encrypt their malicious data in the network and not to make noise in the traffic that would cause the network administrator to believe that suspicious activity exists. There is a chance that the malware goes undetected for many months. The security of the network becomes a challenge for the network administrators as it will be difficult for them to be able to identify suspicious activity. Statistically speaking, it takes "companies on average between 100 and 200 days to detect an attack because 80% of security systems do not recognize or prevent threats within SSL traffic" [5].

Intrusion detection systems (IDS) are devices typically used to monitor the network to keep a lookout for any malicious activities. IDSes can be classified into two categories: signature-based detection and anomaly-based detection. They are put in the network to examine all activity that is going on in the network and search for signs of violation of any policies that have been established. These devices are capable of detecting malicious traffic but cannot stop the traffic from reaching the destination. It is important to have an intrusion detection system in the network however an issue arises with encrypted traffic. A more detailed discussion about intrusion detection systems is discussed in section 2.17, but it is important to mention that the only way for network administrators to examine the traffic would be to decrypt the packet, but that would bring up confidentiality and integrity issues.

This thesis focused on testing out a methodology that may be useful in security monitoring to better detect and prevent malicious activity from occurring from the start. It is using JA3 and JA3S fingerprinting as a security monitoring instrument to improve the identification of malicious activity. The idea is to be able to identify whether the

client application is legitimate or malicious regardless of the data being sent to the destination and to find another technique that can be used to improve security monitoring.

1.2 Problem Statement and Contribution

In 2017 a method called JA3 was developed by John Althouse, Jeff Atkinson, and Josh Atkins. JA3 is a technique to generate SSL/TLS client fingerprints that can be shared for threat intelligence and be simple to produce [6]. This approach was created to distinguish between the SSL/TLS clients, and it was influenced by Lee Brotherston's previous research and TLS Fingerprinting tool (FingerprinTLS).

Encryption network traffic is becoming a concerned security problem as Cybercriminals are hiding their malicious communications in the SSL communication. They are using it to their advantage and making it difficult to detect without having to decrypt the packet. This thesis tested the detection accuracy of JA3 and JA3S fingerprinting in encrypted network traffic. These two techniques were tested against real network traffic to see their detection accuracy. This would help determine if these hashes would be useful as an additional feature to help improve security monitoring. The original values that were used to generate the JA3 and JA3S fingerprints were used to determine the classification accuracy as to whether the group of fields were malicious or normal. This study used the JA3 and JA3S technique to analyze the detection rate and if it is useful in security monitoring.

As JA3 has been fairly recent and may interest organizations to use this method, the following question is posed:

What is the detection accuracy of the JA3 fingerprinting in SSL communication? How does the detection rate differ when using JA3 and JA3S together than only just JA3? What is the detection accuracy of using the original values of the JA3 and JA3S in machine learning models?

1.3 Scope

A fairly recent method called JA3 fingerprinting was developed to aid in detecting malicious activities within a network. The scope of this research looked at the accuracy

of JA3 and JA3S fingerprints in detecting malicious activity versus the accuracy of preserving the original values of the JA3 and JA3S fingerprints.

2 Background

This chapter gives an overview of security monitoring in encrypted network traffic. It provides an introduction to JA3 and JA3S fingerprinting and how it could be utilized in security monitoring which is the main focus of this paper.

2.1 Cyber Threat Intelligence vs. Security Monitoring

Cyber threat intelligence is “what cyber threat information becomes once it has been collected, evaluated in the context of its source and reliability, and analyzed through rigorous and structured tradecraft techniques by those with substantive expertise and access to all-source information” [7]. It is existing threat information from multiple data sources that organizations use to understand the current threats that exist within the organization, possible vulnerabilities, and what cybercriminals are interested in targeting. Cyber threat intelligence can help organizations stay up to date with the most recent threats and vulnerabilities. The information is disseminated on multiple platforms where organizations can go to learn and keep up to date about any new cyber threats.

Security monitoring “collects and analyzes information to detect suspicious behaviour or unauthorized system changes on your network, defining which types of behaviour should trigger alerts, and taking action on alerts as needed” [8]. In most organizations, security monitoring is performed by using intrusion detection system (IDS)/intrusion prevention system (IPS) to monitor the incoming and outgoing traffic. These systems allow the network administrators to monitor the ongoing activity in the network and discover any abnormalities. Besides examining the network, security monitoring can also be used to record events of each machine connected to the network [9]. They can log the login and logouts, authentication attempts, and more [9].

Cyber threat intelligence can be used in security monitoring to aid an organization stay up to date with the latest cyber threats and vulnerabilities. TLS fingerprinting could be used as an element of cyber threat intelligence. The fingerprints may be shared on

multiple platforms (publicly or privately) to allow organizations to use in their security monitoring.

2.2 Transport Layer Security

The Transport Layer Security protocol's main purpose is to provide data confidentiality and integrity between two transmitting applications [10]. This protocol is composed of two layers, the first is the TLS Record Protocol and the second is the TLS Handshake Protocol. The TLS Record Protocol has two major priorities: private connection and reliable connection [10]. The protocol uses the security arguments constructed by the handshake protocol to compact and encode the above layer data [11]. This protocol is used to encase numerous higher-level protocols, one of them being the TLS Handshake Protocol [10]. This TLS Handshake Protocol permits the client and server to validate each other and to make a negotiation on an encryption algorithm and cryptographic keys before data can be sent or received [10]. This protocol contains three properties. The first is that "the peer's identity can be authenticated using asymmetric, or public key, cryptography" [10]. The second is the negotiation of a shared secret cannot be acquired or be available to external listeners. Even if an attacker tried to place itself in the middle of the communication the data would still be unattainable [10]. Finally, the negotiation between the client and server is reliable which meant that cybercriminals could not alter the agreement made without being detected by the individuals of the communication [10]. As of today, the TLS protocol is used to secure network protocols such as HTTP, FTP, SMTP, Voice over Internet Protocol (VoIP) and Virtual Private Network protocol (VPN) [12].

2.3 TLS Handshake

For the purpose of this research, the most critical part of the TLS handshake is the ClientHello message. This is the first message that is sent to the server to initialize communication with the server. Figure 1 shows the full process of the TLS handshake.

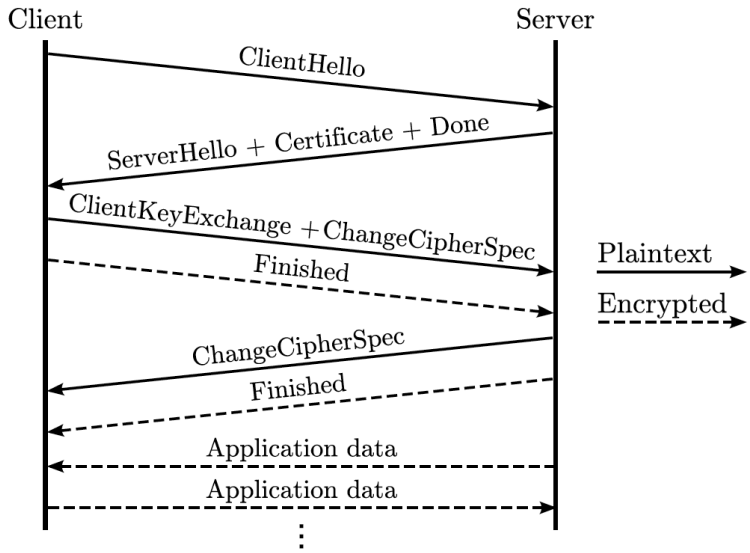

Figure 1. SSL/TLS Handshake. Source copied from [5]

So much information can be acquired just from gathering information from the TLS Handshake. Focusing on client-based features will aid to properly label a malicious agent connecting to google.com compared to a benign agent connecting to google.com [13]. The client's cryptographic arguments can be used to differentiate between malicious and benign events [13]. Looking at the server-side features will help understand the destination that the client is trying to connect to. In the next section, a brief overview of the Bestafera malware is discussed and how it is possible to determine the client of the user agent based on the TLS ClientHello message.

2.4 Case Study: Bestafera

Bestafera is a malware that is mainly known for "keylogging and data exfiltration" [14]. The behavioural pattern of this malware usually contains a self-signed certificate, data exfiltration and a C2 message. When the TLS handshake begins, information about the client that is needed to identify the client type. Figure 2 below display the details of the TLS ClientHello message mapped to the possible clients. The fields that have been boxed in yellow indicate the SSL fields of the client. In this case machine learning was used by taking the fields boxed in yellow and deducing the client based off of the given information. This case study looked at the sequence of packet lengths which provided a behavioural profile of the application and the extensions and cipher suites from the SSL ClientHello packet suggested that the client was using TLS to communicate with the

server [14]. This thesis briefly touched on using the decision tree machine learning algorithm to obtain the accuracy of classification in 4.4 and 5.3.

Figure 2. Detail display of the ClientHello message. Source copied from [7]

2.5 Encrypted Network Traffic

The first step that should be taken before analzying the network traffic to identify the client is to understand the network traffic. After comprehending the network traffic then it is possible to proceed onto client identification and detection of malicious activity [15]. It is necessary to observe network traffic to understand the typical traffic patterns that exist on a specific network. One can obtain a record of many different pattern types from an encrypted network traffic [15]. By monitoring the network traffic one can get detailed visibility of the information available. Information such as the senders, receivers, the conversations, their domains, applications, protocols, incoming and outgoing traffic and utilization, and the bandwidth usage by host and group can be obtained from looking observing the network [16]. It is valuable to determine if the encrypted network traffic is normal or malicious in a way that maintains the purity of encryption [13].

2.6 TLS Connection Establishment

A TLS connection is established by the client and server agreeing on the parameters of the connection by using the Client Hello and Server Hello messages [17]. These messages are not encrypted and allow passive observation. When sending the

ClientHello message to the server, the client will list the highest protocol version it can support along with the cipher suites and elliptic curve points [17]. The server will then choose their favoured options based off of the information proposed by the client and returns a ServerHello message back to the client with its choices [17]. In the end, the server is the one who decides the ultimate protocol version, cipher suite and other arguments that are used in the protected channel for communication [17].

2.7 ClientHello Packet

The idea of using a ClientHello message to help identify a client application was not a new idea. There has been ongoing research since 2009 and in 2015, Lee Brotherston released FingerprinTLS. After this release, it was brought to attention that SSL fingerprinting could be a solution that could work on already existing tools like if it was a network security monitoring device or a load balancer [18]. TLS fingerprinting is an approach where identification of SSL/TLS clients is possible [19]. This technique is built based on the order of information that has been advertised in the ClientHello message, which is the very first message to be sent in the TLS handshake [19]. The message during the handshake is unencrypted. Ciphersuite is the union of the key exchange, encryption, list of ciphers, the preferences, and version [19]. Depending on which cipher suite definitions is used, the server is required to provide the appropriate RSA certificate for the key exchange [20]. Table 1 displays the Ciphersuite definitions.

Table 1. Ciphersuite definitions. Source copied from [20]

CipherSuite TLS_RSA_WITH_NULL_MD5 = { 0x00,0x01 };
CipherSuite TLS_RSA_WITH_NULL_SHA = { 0x00,0x02 };
CipherSuite TLS_RSA_WITH_NULL_SHA256 = { 0x00,0x3B };
CipherSuite TLS_RSA_WITH_RC4_128_MD5 ={ 0x00,0x04 };
CipherSuite TLS_RSA_WITH_RC4_128_SHA= { 0x00,0x05 };
CipherSuite TLS_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0A };
CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA = { 0x00,0x2F };
CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA = { 0x00,0x35 };
CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA256 = { 0x00,0x3C };
CipherSuite TLS_RSA_WITH_AES_256_CBC_SHA256 = { 0x00,0x3D };

The TLS extensions depend on what the client supports and shown in a specific order in the message. This relied upon the SSL library type and version [19]. As for the elliptic curves, there are approximately twenty-five curve types that have been registered by the Internet Assigned Numbers Authority (IANA) [19]. This number and preference will vary depending on the client [19]. Once these fields have been gathered from the ClientHello message from the TLS handshake, JA3 will generate the fingerprint.

2.8 JA3 Fingerprint

JA3 was developed by three Salesforce members, John B. Althouse, Jeff Atkinson and Josh Atkins. It is a technique used to generate SSL fingerprints based on the ClientHello packet in order to identify the client that established an encrypted connection [21]. This gives clarity from the start as to whether or not the client is malicious. In the best case, JA3 can be used to recognize malware and botnet command and control traffic that uses SSL/TLS [22]. The JA3 hash represents an SSL/TLS client application that has been detected by a device or network sensor [23]. JA3 enables organizations to be able to detect malware, applications, and more, regardless of the destination, the Command and Control IP addresses or the SSL certificates [23]. Each server or client based software may use various TLS configurations within themselves [24]. Each TLS configuration can be used to identify the type of software, libraries, and specific TLS settings (ciphersuites, extensions, etc.) used in the software [24].

2.9 JA3 Fingerprint Generation

The process of generating JA3 fingerprints is to separate multiple fields within the TLS ClientHello packet that is sent during the SSL handshake[25]. These fields are then concatenated using a comma to delimit each field as can be seen later in this section [23]. Initially, some believed just to hash the entire packet. However, this was not going to work “as there’s a random string within the packet as well as certain SSL extensions which can hold destination specific information like the ‘Server_Name’ extension which holds the destination domain” [18]. If no SSL extensions exist within the ClientHello packet, then the fields can be left empty [23].

The fields that are essential to generate a fingerprint is the SSL version, ciphers, extensions, elliptic curves, and elliptic curve point formats. The ciphers, extensions and elliptical curves help identify the client. The SSL version is the TLS protocol version

that the client requests to communicate with during the session [20]. This should be the latest version that is supported by the client. The cipher suites consist of a key exchange algorithm, a bulk encryption algorithm and a message authentication code (MAC) algorithm. It contains a mixture of cryptographic algorithms that the client supports in order of its preference [26]. The key exchange is the key that is exchanged between the two devices. Bulk encryption is the process of encrypting the data and the MAC algorithm provides data integrity, making sure that the data being sent has not been altered. If all three fields are placed together, there is a high chance that it will decrease the number of collisions [19]. The long string is converted from decimal values to an MD5 hash to create an easy to use 32-bit character fingerprint, which is called a JA3 SSL Fingerprint [23]. The reasoning behind using MD5 hashes is so that JA3 can be easily incorporated into current products. JA3 signatures will work for all TLS/SSL communications and JA3 will be able to associate various attacks to the same client. [24].

2.10 Benefit of JA3

The benefit of utilizing JA3 is that it has many advantages as an indicator of compromise (IoC) in comparison to IP addresses or domain names [21]. These almost one of a kind fingerprints can be used to improve conventional cyber security approaches like blacklisting, whitelisting, and search for IoCs [27]. JA3 hashes could be used as an input signature to typical cyber security solutions to identify and block malware [27]. JA3 hashes could also be investigated in datasets to find other traffic communications that use the same JA3 hash [27]. JA3 makes it possible to detect malware based on the communication and not on what the client connects to [23]. An interesting note worth mentioning is that if a cybercriminal creates their own executable malware it will likely have a unique fingerprint. These fingerprints can be used to help detect a popular technique that is used today to exfiltrate data and circumvent Internet censorship, which will be briefly discussed later in this paper.

The five fields of TLS communication are enough information to identify a client [28]. Again, when speaking about identifying the client this means looking at the type of client application to know immediately if the client is legitimate or malicious. The rationale behind this method is that tools are much harder to change than IP addresses or domain names [28]. However, this method is in the field of signatures and is extremely

dependent on the blacklists and whitelists that are available Regardless, this method can be used to find deviations in the network [28].

2.11 Limitation of JA3

A limitation of JA3 fingerprinting is that there is a possibility that two client applications have the same JA3 fingerprint. Although the client applications themselves are different from each other they may be using the same libraries or OS sockets for communication [29]. Therefore, the JA3 fingerprint would not be useful for detection and identifying whether the communication is legitimate or malicious. This is when JA3S can be used alongside with JA3 to better assist in identification. The number of possible JA3 and JA3S fingerprint hashes are limited due to the number of fields used to generate the fingerprints and due to a finite number of values that can be used in each field.

2.12 JA3S

JA3S is for the server side of the SSL/TLS communication. The generation of the JA3S is similar to JA3 except fewer fields are needed. It uses only the SSLVersion, Cipher, SSLExtension. JA3S cannot generate a JA3S fingerprint based on the ServerHello message. JA3S makes it possible to fingerprint “the entire cryptographic negotiation between a client and it’s server by combining JA3 + JA3S” [6]. This is possible because a server will always reply to the same client the same way and differently to different clients [6].

2.13 JA3-JA3S Pair

Althouse mentioned in one of his Twitter conversations that JA3S, “cannot be used alone for detection or blacklisting, as it only holds value when combined with the JA3 of the client” [30]. The combination of JA3 with JA3 is likely to be more effective than JA3 alone as it would reduce the number of false positives.

2.14 JA3 Usage

JA3 fingerprints can be useful in detecting malicious circumvention techniques that are discussed in the following sections. Although this research does not work with these

topics, it is important to note how useful JA3 and JA3S can be in detecting malicious activity in security monitoring.

2.14.1 Domain Fronting

Domain fronting is one circumvention technique that "hides the endpoint of a connection using HTTPS while communicating with censored hosts" [31]. It is using HTTPS to communicate with a forbidden host to make it appear that it is communicating with a host that is permitted by the censor [32]. It is a circumvention technique used to disguise the true destination of the client's message by rerouting them through a content delivery network that hosts many websites [33]. From the firewall's perspective, the HTTPS request looks like it is going to a legitimate website when in reality it is going to another malicious site that would usually be blocked. In a deeper sense, different domain names are being used at different layers of communication and it works on the application layer.

Today, most cybercriminals are "increasingly using cyber squatted domains to register a domain that is very similar to the victim/target's corporate domain" [34]. This is often done to avoid suspicion and any human analysis. It also makes it difficult for the blue team members to determine if these new domains have been registered by legitimate developers or for any new products/services [34]. As modern command and control channels typically use encryption it becomes more difficult and/or nearly impossible to detect SSL traffic without using SSL termination, but there are some privacy and GDPR concerns with this [34]. The concern that arises with SSL termination is that the data must be decrypted and inspected. This raises the matter of confidentiality and integrity of the data. Therefore, domain fronting still exists and remains undetected. If the IP address of the client is frequently changing, it would not matter because the JA3 fingerprint hash would remain the same since fingerprints rely on the SSL ClientHello fields. It would be possible to identify the type of client and where it is trying to communicate regardless of the encrypted data being sent.

2.14.2 Detecting Domain Fronting using JA3 and JA3S

JA3 and JA3S can be used to detect domain fronting activity in a network. The fingerprints created for the SSL/TLS ClientHello and ServerHello and will remain the same regardless if the IP address of the client or server changes. These fingerprints can be used to determine the type of application (browser, email programs, software, etc.) before an SSL connection has been established. In domain fronting, the actual

destination address is hidden in the Server Name Indication (SNI), but that does not matter because JA3 only cares about the SSL fields from the ClientHello and ServerHello packets.

Unsupervised machine learning can be combined with JA3 to identify clients that have an unusual JA3 as mentioned here [27]. There are many blog posts that discuss how unsupervised machine learning "makes the detection of domain fronting without having to break up encrypted traffic possible by combining unusual JA3 detection with other anomalies such as beaconing" [34].

2.15 Data Exfiltration

Data Exfiltration is a popular technique amongst attackers that when used could harm an organization's reputation and is a significant issue organizations are facing [35]. It is the unpermitted transfer of data from systems such as a user's system or IT servers [35]. These transfers can be performed manually or automatically via a malicious program being sent over a network [35]. This method is used to exfiltrate sensitive information from an organization. Examples of sensitive data include personal data, customer data, internal tools and software. A typical data exfiltration scenario would "involve the attacker establishing a Command and Control (C2) connection to remotely control the compromised machine" [36]. In order to gain control, the attacker must first penetrate a host system by exploiting the security vulnerabilities in the organization [36]. Once the host machine has been infected a backdoor or Remote Access Tool must be installed. A backdoor is part of a malicious code that is placed on a compromised system to aid in command and control and data exfiltration [36]. Remote access tools and backdoors are types of malware that uses command and control to exfiltrate data out of an organization [36]. Data exfiltration requires a passageway to allow data to be transferred elsewhere [37].

2.15.1 Detecting Data Exfiltration using JA3

When alerts are generated by the IDS engine it is possible to see the source IP address and the destination IP address along with some other details. If the client is trying to connect to a suspicious destination, then further investigation can be completed. Even though the information is encrypted it is still possible to determine whether or not the client is connecting to a command and control server. The JA3 fingerprint of the client can be seen as to whether it is a browser, malware, etc. this could be an advantage in the

early detection of possible attackers. JA3 fingerprinting can be thought of as an additional component that could signal the use of unauthorized or unwanted software at the beginning stages of a malware communication [34].

2.16 Network Monitoring

Organizations monitor their network to be aware of what is happening in their internal environment. Network monitoring is used to monitor and analyze network infrastructure to identify issues that could affect the performance [38]. Typically network monitoring solutions generate alerts to notify the IT administrators of issues within the network [38]. These alerts will allow the organization to find the issue and resolve it before it gets worse. The system will also generate a detailed report to help companies understand the network infrastructure [38]. These reports can help answer questions that companies may have. Various types of logging and traffic monitoring tools could help network administrators to get a clear understanding of all the devices in the network [39]. This may be achieved by using different tools such as firewalls, IDS/IPS, or packet capturing [39].

2.17 Intrusion Detection System

Organizations should monitor their network traffic for malicious activities and conduct additional analysis to differentiate between the malicious and legitimate user activities to secure their network [40]. In order to detect malicious activity an intrusion detection systems (IDS) must be implemented in the network. However, IDSes are only beneficial if they have superior detection capabilities [40]. It is extremely critical that the IDS detection mechanism is detailed enough to be able to tell the difference between malicious and legitimate traffic [40].

2.18 Importance

Organizations constantly aim to keep the continuous operation of their networks to prevent any downtime [41]. If a network was down for the slightest amount of time the productivity within a company would decrease and public service departments would be limited in providing crucial services [41]. For this reason, network administrators should monitor the traffic and performance of the network to double-check that security breaches do not take place within the network [41]. Network monitoring can guide an

organization to discover suspicious entryways that have been placed intentionally by cybercriminals or unintentionally such as an unused service that has been enabled [39]. Traffic monitoring tools can help distinguish between malicious traffic and normal traffic [39]. Network administrators "usually deploy the monitoring systems in the intra-domain Internet to capture, analyse, and decide whether an instance is normal or anomalous" [25, p.45].

2.19 Types of Network Monitoring

There are many types of network monitoring solutions. These systems could be firmware or software and they can be carried out on site or handled externally via a service provider [38]. Traffic monitoring tools help organizations to be familiar with their normal traffic in order to be able to recognize suspicious traffic that is passing through the network [39]. Companies may also outsource network monitoring requirements to third-party vendors to closely keep an eye of the network infrastructure via a personalized dashboard on the Internet [38]. Some types of traffic monitoring tools include bandwidth monitoring, packet sniffing, and network-based IDS. A couple of these tools include Suricata, Snort and Bro, however, in this research, only Bro and Suricata will be used.

2.19.1 Snort

Snort is "'application aware'" meaning that it only looks at the traffic that matches its rules and then takes actions such as alert, drop, etc. when a match has been found [43]. Snort does not support multithreading, therefore it does not "matter how many cores a CPU contains, only a single core or thread will be used by Snort" [43]. One disadvantage of Snort is that it is an extremely old tool and was invented to run on an older infrastructure [44]. One reason why the improvement of a similar open source system was created was because of its performance as it is limited to a single thread architecture [45]. A disadvantage to snort is that although the rules are simple to write it becomes complicated to compose rules for complex conditions [46].

2.19.2 Bro

Bro is better considered as an "'anomaly detection system'" and it uses scripts to inspect the traffic [43]. One major advantage of Bro is that the scripts allow for better quality automated workflows between different systems. This is an approach that allows for

more detailed decisions than the typical pass or drop actions [43]. Users frequently use Bro to document the network behaviour [44]. Bro accumulates the network metadata that it records much nicer than packet capture. This means that the data can be searched, indexed, queried and reported in a manner that was formerly non-existent [44]. Bro supports JA3S while Suricata does not currently support it.

2.20 Classifying Network Traffic

Classifying network traffic is beneficial to an organization. When an organization classifies network traffic, they are able to see what types of traffic exist, organize the traffic into different categories and give specific treatments to certain traffic [47]. Identifying and organizing the traffic will help the employees allocate the proper network resources to deliver excellent performance for various types of traffic [47]. For example, vital network traffic or any traffic that matches a specific criteria can easily be pointed out for appropriate handling to attain maximum application performance [47].

2.21 Suricata

Suricata is an open source network threat detection engine that was developed by the OISF. It is a rule-based intrusion detection/intrusion prevention system engine that uses rulesets that have been established externally to monitor the network traffic and notifies the system administrator if there are any suspicious events [48]. Suricata can be used to examine network traffic by using its rules and signatures. This engine has the capability to administer large amounts of traffic and send the alerts via emails [49]. It supports application-layer detection rules and has the capability to recognize HTTP or SSH traffic, for example, on abnormal ports based on protocols [43]. It improves the visibility of traffic which will increase the detection of malicious content. It should be noted that it really depends on what an organization is looking for, and which system does the job right in terms of detection. Suricata also supports multithreading, which is one of its main benefits. Using a multi-threaded architecture will allow Suricata to instantly collect and unravel the packets, as shown in Figure 3. below [40].

Suricata was the chosen intrusion detection system in this thesis because it has been refined in recent years and has much better features than Snort. It also supports JA3 fingerprinting.

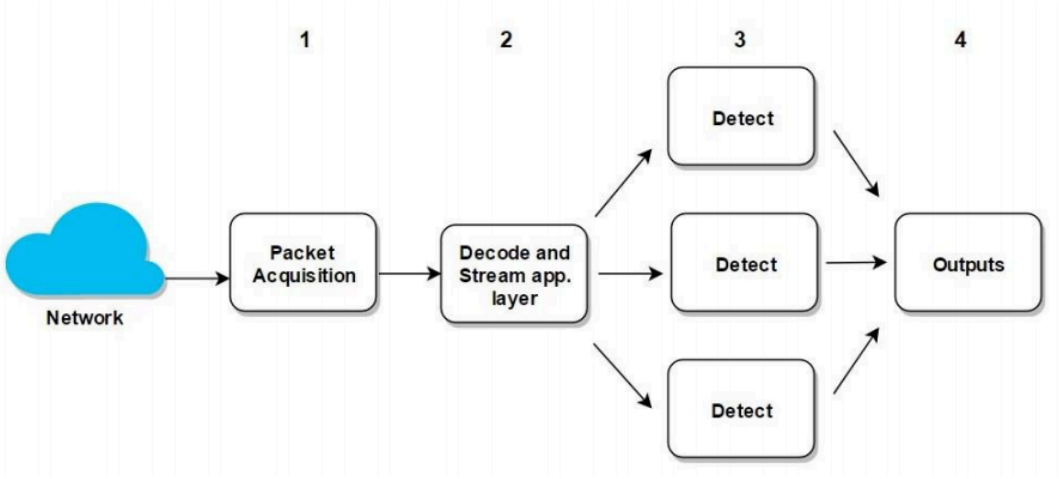

Figure 3. Suricata's multi-threaded architecture. Source copied from [24]

This engine allows for greater traffic inspection performance which means that more rules can be operated against large amount of traffic [46]. There are other IDS solutions like SNORT and Bro, but Suricata is an improved version of Snort. It should be stated that a rule-based solution like Suricata or Snort is much more efficient in monitoring elementary alerts [46]. In Day and Burns research, the analysis of the results displays that Suricata has a greater certainty than Snort but this comes with a price of placing a high demand on the CPU [50].

2.22 False Positives and False Negatives

False positive means that the IDS has triggered an alert thinking that there was a malicious activity in the network traffic when there was actually none and false negative is when no alert has been triggered by the IDS when there was actually malicious activity going on in the network [51]. False positive and false negative are two metrics that are important in measuring the accuracy of Suricata. The possibility of having false positives and false negatives may occur and there are a number of ways that this can occur. One way would be to have incorrect rules meaning such as invalid information in the signature or incorrect rule generation, but this way is quite rare [52]. Another issue is that the rules may be inaccurate because there is not a signature for that specific attack [52]. Other false positives and false negatives may occur due to performance issues with the detection engine [52]. False positives may cause security analysts to put unnecessary effort into an alert that was mistakenly identified as malicious. Therefore, it is important to pay attention to the false positives and false negatives to aid in the accuracy of detection.

In regards to JA3 the fingerprint that has been generated for an SSL client or server, these fingerprints are not unique to only one client [29]. There may be other SSL client applications that consequently have the same fingerprint as one another. It is imperative to mention that JA3 can be thought of as an extra part of information that can be used to help in detection.

3 Related Work

There are a few studies that exist but do not focus specifically on JA3 but on TLS fingerprinting. There is one study which addressed JA3, but the paper was not focused on testing out this method [21]. In this paper, Mieden briefly touches on JA3 and what it is and how the fingerprints are formed, but no other details were discussed regarding JA3. The research conducted in [53] and [54] discuss how to recognize and detect SSL traffic at the beginning stages or how to identify a client using SSL/TLS fingerprinting. The methods of these two studies are briefly mentioned below as their approach demonstrates that it is possible to identify client applications is different than the approach that was taken in this paper.

In Bernaille and Teixeria's study [53], they proposed a method to detect applications that use SSL encrypted connections. Their method relied only on the size of the first couple of packets within the SSL connection which allows early classification to recognize the application [53]. The proposed method was tested on two campus networks and on manually-encrypted traces. This study did not address any security issues in regard to how identification can aid in detecting suspicious activity in the network. Rather, it was more focused on proposing techniques that could be used to detect applications in an SSL communication. The method presented in [54] demonstrated the possibility of estimating the User-Agent of a client application by analysing the SSL/TLS handshake in HTTPS communication [54]. The concentration of this research was to demonstrate that it was possible to identify the client application from examining the TLS/SSL handshake. Bernaille and Teixeria specifically looked at the fingerprints of the SSL/TLS handshakes which included the list of supported ciphersuites. A dictionary was created to observe the SSL/TLS connections in order to identify communicating clients. These two methods did not look at specific SSL ClientHello Packet fields that get converted into MD5 hash fingerprints. Both of these studies do not address any security problems within traffic classification and malicious activity rather the focus is more on the identification of applications that use SSL/TLS for communication.

The study presented in [15] closely related to the scope of this thesis but performed a different experiment. Two of the research questions that Čeleda, Čermák, Husák and Jirsík questioned were similar to what will be studied in this paper – which parameters of an SSL/TLS handshake could be used to identify clients and if it is possible to utilize TLS fingerprinting in network security monitoring and intrusion detection. The experiment performed in [15] was different than what was conducted in this thesis. The scope of this thesis focused attention on the SSL/TLS handshake and comprehended the fields within the SSL ClientHello packets of the client and server to understand what fields are. The main difference between this paper and [15] is that explicit TLS/SSL parameters were already known along with the whether the fingerprints could be used in network monitoring which were stated by the developers of JA3.

Lee Brotherston was first to fingerprint TLS clients in 2015 and he has previously demonstrated that malicious clients could be differentiated from the rest of the other network traffic. This would assist the network administrators to detect the malicious client without requiring any other information. The purpose of Brotherston's work was to be able to quickly identify the client used for communication without needing to know data being sent. He designed FingerprinTLS which was created to quickly identify the known TLS connections and to fingerprint any of the unknown connections [55]. It was designed to improve the accuracy of the fingerprints and to allow others to use this tool in their own environment to create fingerprints. The input of data was taken "either via live network sniffing or reading a PCAP file" [55]. JA3 TLS fingerprinting enabled the detection of an enormous range of malicious traffic without needing to access either endpoint. The developers of JA3 designed this method as an extension to FingerprinTLS. It is a feature that is supported in most IDS engines like Suricata, Bro, and more. This feature can be enabled in an organization's network to trigger alerts against any suspicious activities.

Garcia and Střasák conducted an experiment trying to detect malware that is encrypted [56]. The goal of their experiment was to discover features and techniques to examine HTTPS traffic without decrypting the packets. Instead of using SSL fingerprinting in their experiment, Garcia and Střasák used the SSL communication features such as source and destination IP address, number of packets, number of certificates and more. This is so that they could inspect and detect malicious communication without decryption. They also wanted to have a high malware accuracy detection rate with a minimum number of false positives. The approach that Garcia and Střasák has taken

was similar to the method taken in this thesis. They also used the dataset from StratosphereIPS and feeding those PCAP files into Bro. The main difference with Garcia and Střasák's experiment compared to this thesis were the machine learning algorithms used to perform the experiment. They used XBBoost, Random Forest, Neural Network and SVM whereas this thesis only used the decision tree algorithm.

This research took the current existing information on TLS fingerprinting and JA3 and took a different approach. This study focused on testing the JA3, JA3S, and JA3-JA3S pair method on real-world network traffic and also looking at the classification accuracy based on a machine learning algorithm.

4 Methodology

The purpose of this study was to test the detection accuracy of JA3 and JA3S fingerprinting in SSL communication. This research utilized JA3 and JA3S fingerprints to obtain the detection accuracy to determine if it could be used as an extra piece of information in security monitoring. The original values that are used to generate the JA3 and JA3S fingerprints were tested to determine if they would be more efficient in monitoring the network. In the following trials, Suricata and Bro were used to understand the generated alerts, the kind of client application, and also whether the alerts were accurate. These two techniques can be useful for companies to improve their methods of monitoring their network. This study consists of three different trials which look at how accurate the JA3 fingerprint hashes and the original values of JA3 fingerprinting can be in a network. The first two trials use the TLS fingerprinting hashes while trial 3 uses the original values of the TLS/SSL ClientHello fields that are used to generate the fingerprints in machine learning. Trial one was conducted at TalTech where the latest version of Suricata with JA3 enabled was running for four consecutive days. The data was analyzed to see if any malicious activity existed. Trial two used publicly available normal and malicious datasets to understand what kind of client and server applications were in communication. Bro generated fingerprints for the client and server side. Trial three utilized the original values used to create the JA3 fingerprints from the TLS/SSL ClientHello packet in machine learning to find the classification accuracy of whether the traffic is legitimate or malicious in a network.

These trials will assist to determine whether JA3 hashes or the original values are more valuable in security monitoring.

4.1 JA3 Testing Phase

The first step was to contact abuse.ch to ask questions about JA3 fingerprinting and to get a better understanding of how to proceed with the experiment. A test environment was created to understand the JA3 functionality in Suricata. The test environment was

created with one Windows 7 virtual machine and one Ubuntu (18.04.1) virtual machine. On the Ubuntu machine, the latest version of Suricata (4.1.2) was installed as it supports JA3. In order to utilize JA3, it must be enabled in the suricata.yaml file. The JA3 fingerprint ruleset was downloaded from abuse.ch in order to detect and block the malicious SSL connections [22]. The Windows 7 (client) was used to execute malware samples that were given by abuse.ch.

```
(base) Jessicas-MBP:Downloads jessica$ cat ja3_fingerprints.rules
################################################################
# abuse.ch Suricata JA3 Fingerprint Ruleset                    #
# For Suricata 4.1.0 or newer                                  #
# Last updated: 2019-02-22 07:10:33 UTC                        #
#                                                              #
# Terms Of Use: https://sslbl.abuse.ch/blacklist/              #
# For questions please contact sslbl [at] abuse.ch             #
################################################################
#
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"906004246f3b
a5e755b043c057254a29"; reference:url, sslbl.abuse.ch/ja3-fingerprints/906004246f3ba5e755b043c057254a29/; sid:906200000; rev:1;)
```

Figure 4 below displays one of the rules from the Suricata JA3 fingerprint ruleset. This rule states that the detected JA3 fingerprint belongs to the Tofsee malware family, an abuse.ch link is provided containing more information about the malware, and the SID is also given.


```
(base) Jessicas-MBP:Downloads jessica$ cat ja3_fingerprints.rules
################################################################
# abuse.ch Suricata JA3 Fingerprint Ruleset                    #
# For Suricata 4.1.0 or newer                                  #
# Last updated: 2019-02-22 07:10:33 UTC                        #
#                                                              #
# Terms Of Use: https://sslbl.abuse.ch/blacklist/              #
# For questions please contact sslbl [at] abuse.ch             #
################################################################
#
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"906004246f3b
a5e755b043c057254a29"; reference:url, sslbl.abuse.ch/ja3-fingerprints/906004246f3ba5e755b043c057254a29/; sid:906200000; rev:1;)
```


Figure 4. Suricata JA3 Fingerprint Ruleset

Abuse.ch have provided an approximately 100GB file of malware samples along with their PCAP files. The aim of the experimental setup was to test whether or not Suricata was able to catch the traffic from the Windows 7 machine. After executing a malware sample in the Windows machine, it was possible to view the alerts that were generated by Suricata. The malware executed was “Downloader.Agent.Win32.351249”. The Suricata alerts generated in Figure 5 for the malware executed was the date and timestamp, SID, malware family, classification, priority level, source IP address & port number and destination IP address & port number. Figure 5 below displays the alerts generated for this mini experiment.

```
05/04/2019-11:44:43.005188  [**] [1:906200045:1] SSLBL: Malicious JA3 SSL-Client
Fingerprint detected (Adware) [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.1.198:49212 -> 18.153.11.24:443
05/04/2019-11:44:43.054431  [**] [1:906200045:1] SSLBL: Malicious JA3 SSL-Client
Fingerprint detected (Adware) [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.1.198:49216 -> 216.58.211.130:443
05/04/2019-11:44:43.085931  [**] [1:906200045:1] SSLBL: Malicious JA3 SSL-Client
Fingerprint detected (Adware) [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.1.198:49214 -> 173.241.240.143:443
05/04/2019-11:44:43.293533  [**] [1:906200045:1] SSLBL: Malicious JA3 SSL-Client
Fingerprint detected (Adware) [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.1.198:49215 -> 52.200.63.47:443
```

Figure 5. Alerts generated with JA3 fingerprint ruleset

The alerts above show that that the malware triggered the Adware rule from the JA3 ruleset with a SID of 906200045, classification type of null and a priority level of 3. These were the alerts expected if Suricata functioned properly, which it did. Since the test environment had shown that Suricata was able to generate alerts for the executed malware, the next step was to deploy it to the university's network. This was a simple experiment to understand how Suricata functioned and to verify that the correct alerts were generated based on the examples seen online.

4.2 Trial 1: Analyzing JA3 Fingerprints using Suricata

The alerts obtained for this trial were generated by the university's intrusion detection system from the TalTech SOC. JA3 was enabled in Suricata and used the JA3 Fingerprint Ruleset provided by abuse.ch that contained sixty-seven different alerts based on different types of malware. This configuration on Suricata was deployed on Tallinn University of Technology's (TalTech) network starting from March 3rd, 2019 at midnight and ending on March 6th, 2019 at 8:50 PM. There were two Suricata instances on the same network monitoring sensor. The first instance had Suricata version 4.0.6 with university custom rules and the second instance had Suricata version 4.1.2. The second instance had the latest version of Suricata, which supports JA3. This was needed in order to enable JA3 and have it running in the background. This instance will generate with fingerprints of the SSL client applications. After four days of conducting this experiment, there were two log files that were analyzed (one from the custom rules and the other with JA3 fingerprint ruleset). The experiment was held for four days because so many alerts (107,222) were generated it would not have been possible with time to analyze the logs for a longer time period.

4.3 Trial 2: Analyzing JA3, JA3S, and JA3-JA3S pairs using Bro

In trial one only detailed investigation could be done on the SSL client and not on the server. Suricata was capturing data of the entire campus network with some exceptions. Many of the departments within the university are managing their own network and allowing devices to connect to their network at their own risk. The purpose of trial 2 was to use Bro, another IDS engine, to analyze the fingerprints from not only the client side but also the server side. Bro was used in this trial because it supports JA3S whereas Suricata does not currently support it. This trial used public datasets from abuse.ch and StratosphereIPS and not TalTech's data because the network traffic (PCAP files) were not provided, only the JA3 and university rule logs were given. It was not possible to receive the PCAP files for the time that the experiment was performed therefore other datasets were used.

A Bro environment (version 2.6.1) was set up on Ubuntu 16.04 virtual machine. Once Bro was configured, JA3 was installed by using the command: bro-pkg install ja3. By default, "ja3.bro will only append ja3 to the ssl.log, but if you would like to log all aspects of the SSL Client Hello Comment, you have to uncomment the following lines in ja3.bro", shown in Figure 6 below [57]. Uncommenting these lines were done in both ja3.bro and ja3s.bro for this trial.

```
#  ja3_version:  string &optional &log;
#  ja3_ciphers:  string &optional &log;
#  ja3_extensions: string &optional &log;
#  ja3_ec:         string &optional &log;
#  ja3_ec_fmt:     string &optional &log;
```

...

```
#c$ssl$ja3_version = cat(c$tlsfp$client_version);
#c$ssl$ja3_ciphers = c$tlsfp$client_ciphers;
#c$ssl$ja3_extensions = c$tlsfp$extensions;
#c$ssl$ja3_ec = c$tlsfp$e_curves;
#c$ssl$ja3_ec_fmt = c$tlsfp$ec_point_fmt;
```

Figure 6. Lines that can be uncommented to see all SSL Client Hello features. Source copied from [57]

The following command was used to analyze the malicious and normal traffic PCAP files: find . -name "*.pcap" -type f -execdir /nsm/bro/bin/bro -Cr {} /nsm/bro/share/bro/site/local.bro \; and the output displays a good amount of

information including the source IP, destination IP, JA3 and JA3S hash, subject (which consists of the common name, organizational unit, organization, locality, state/province name, and country). For the purpose of this experiment, the information that was focused on was the JA3 and JA3S fingerprints. The PCAP files were taken from abuse.ch and Stratosphere IPS [58].

4.4 Trial 3: Decision Trees in Machine Learning

A decision tree is "one of the most frequently and widely used supervised machine learning algorithms" [59]. It is a part of the supervised learning algorithm. Decision Trees in machine learning can be used to learn patterns and make decisions based off of the given data. An advantage of using this method is because it can be "used to predict continuous and discrete values, require relatively less effort for training the algorithm" and are very fast and efficient compared to other algorithms [59]. Decision trees can perform both regression and classification tasks where the output for regression is numerical whereas the output for classification is categorical [60]. The results given by the algorithm include a confusion matrix, classification accuracy, and classification report. The confusion matrix is a table with values representing true positive, true negative, false positive, and false negative. It is used "to describe the performance of a classification model on a set of data for which the true values are known" [61]. True positive represented the actual positive and were also predicted as positive. True negative represents the actual negative were also predicted as negatives. As mentioned in 2.22, false positive is when something is actually negative but was predicted as positive and false negative is when something is actually positive but has been predicted as negative. A classification report was also generated which "shows a representation of the main classification metrics on a per-class basis" [62]. The columns in a classification report include precision, recall, f1 score, and support. The precision score represents the capability of the classifier to not inaccurately label an instance as positive that is actually negative [62]. In other words, for all of the instances that have been predicted as positive, what was the percentage of accuracy [62]. For example, if there was a precision score of 0.8, it means that when the machine learning algorithm predicts malicious activity, it is correct 80% of the time. The recall score represents the percentage that classifier has identified the actual positive instances correctly [62]. For example, if there was a recall score of 0.6, this means that the machine learning algorithm has accurately identified 60% of the malicious communications. The f1 score

is a weighted balance the precision and recall values, with 1.0 being the best score and 0.0 being the worst [62]. In general the f1 scores are lower than accuracy value since the f1 score incorporates other measurements like precision and recall into its calculations [62].

In this trial, an excel table was created with columns containing the plain text data (SSL version, ciphers, extensions, elliptic curve point, elliptic curve point format for JA3 and SSL version, cipher and extensions for JA3S) from the ClientHello packet for malicious and normal traffic. The last column in the table is the "label" column and this is marked as either normal or malicious depending on the data in the table. The values that were used to label malicious and normal traffic was 0 and 1; 0 represented the malicious traffic and 1 represented the normal traffic. This trial did not include the JA3-JA3S pairs that were seen in both malicious and normal traffic. This table was saved as a CSV file which was processed by the decision tree algorithm. The algorithm divided the data into training and testing sets where some of the data was used for training the learning model before being tested for accuracy.

The Python's Scikit-Learn library was used to implement the decision tree and the libraries imported were panda and numpy. The panda library allows for the input of data files, such as CSV. The numpy library was imported to help with the mathematical calculations. The sources of the implemented algorithm were [59], [63], and [64] The tool that was used to build and run the decision tree algorithm was "Anaconda-Navigator". It is an open source platform that is used to run machine learning algorithms on Mac, Windows and Linux.

5 Results and Discussions

This chapter provides a detailed discussion for Trial 1, Trial 2 and Trial 3 and the results of each trial were discussed in detail.

5.1 Trial 1

Trial 1 was conducted on TalTech's network where the traffic was monitored using Suricata. The data was examined to determine if there were any signs of suspicious activity. Over the span of these four days, the total number of unique IP addresses that are normal and malicious is 2866. 50 IP addresses out of the 2866 (1.7%) were seen in the university's present rule and 694 IP addresses out of 2866 (24.2%) were seen in the JA3 enabled rules. These percentages themselves already give an immense amount of valuable information. This percentage is referring that 24.2% of the IP addresses within the university's network had malicious communication (according to the JA3 rules) and that seems highly unlikely in such an environment. On average, there were about twelve alerts generated per minute, which was calculated by dividing the number of alerts by the total minutes within the span of four days. Figure 9 is a graph that displays the number of alerts generated per hour each day for the duration of the experiment (four days). It can be seen in the screenshot below (Figure 7) of the logs that the client IP addresses have been anonymized since they are internal university IP addresses. Figure 7 displays the default rule logs while Figure 8 displays the JA3 rules log.

Note: The word alerts refer to each set of rows in the log file and the word log is referring to the JA3 log and the university rule log which are files that contain alerts in them.

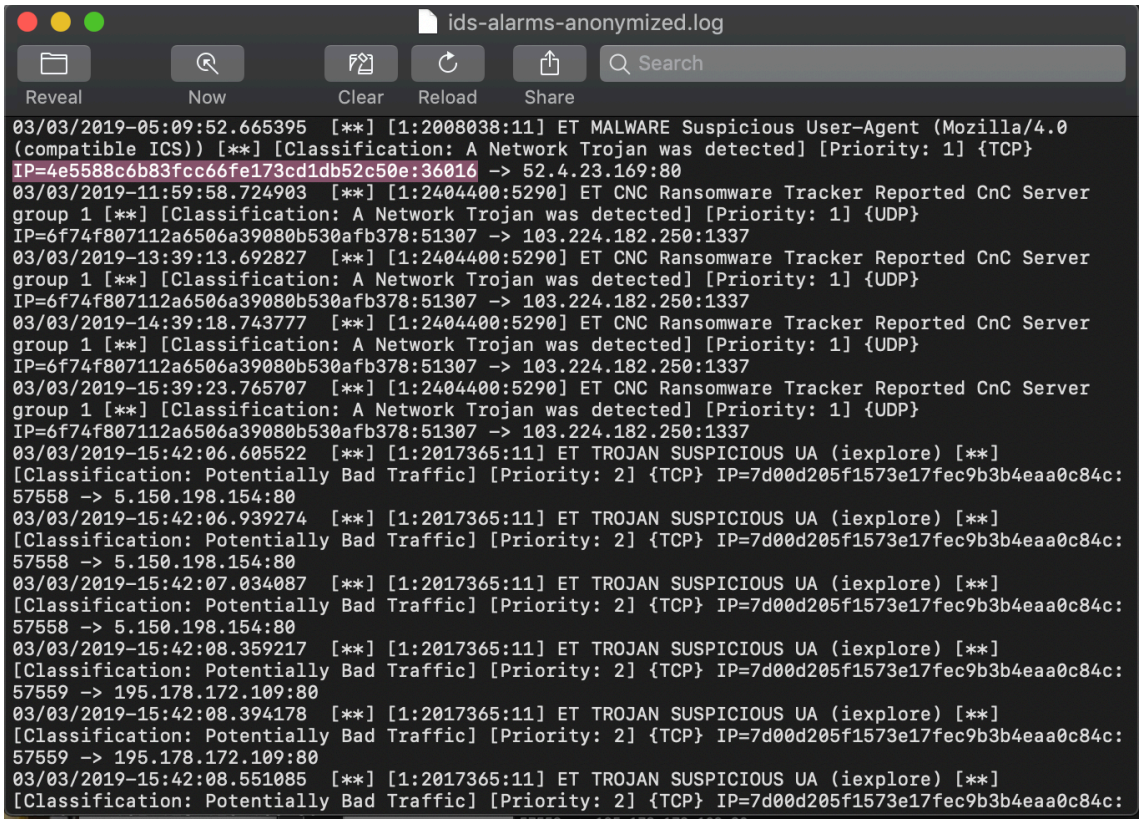

Figure 7. IP addresses have been anonymized by utilizing MD5 hashes; IDS default rule alerts

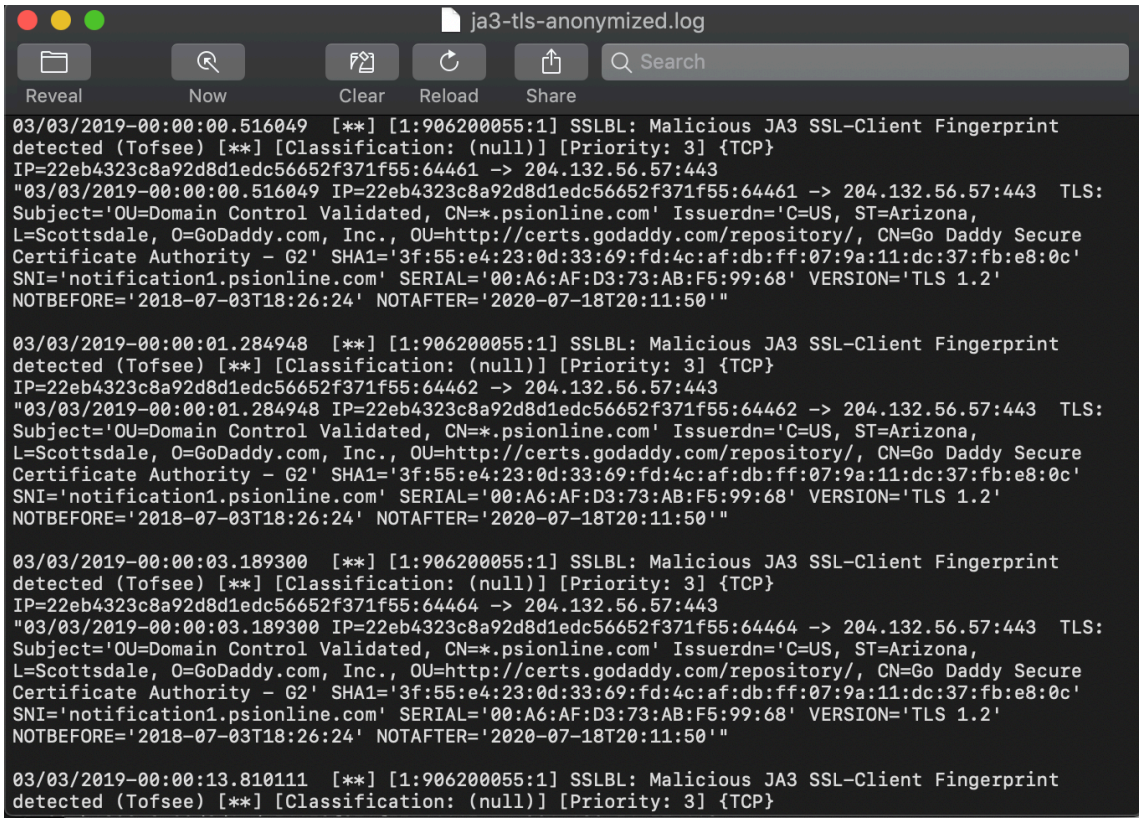

Figure 8. JA3 ruleset generated alerts

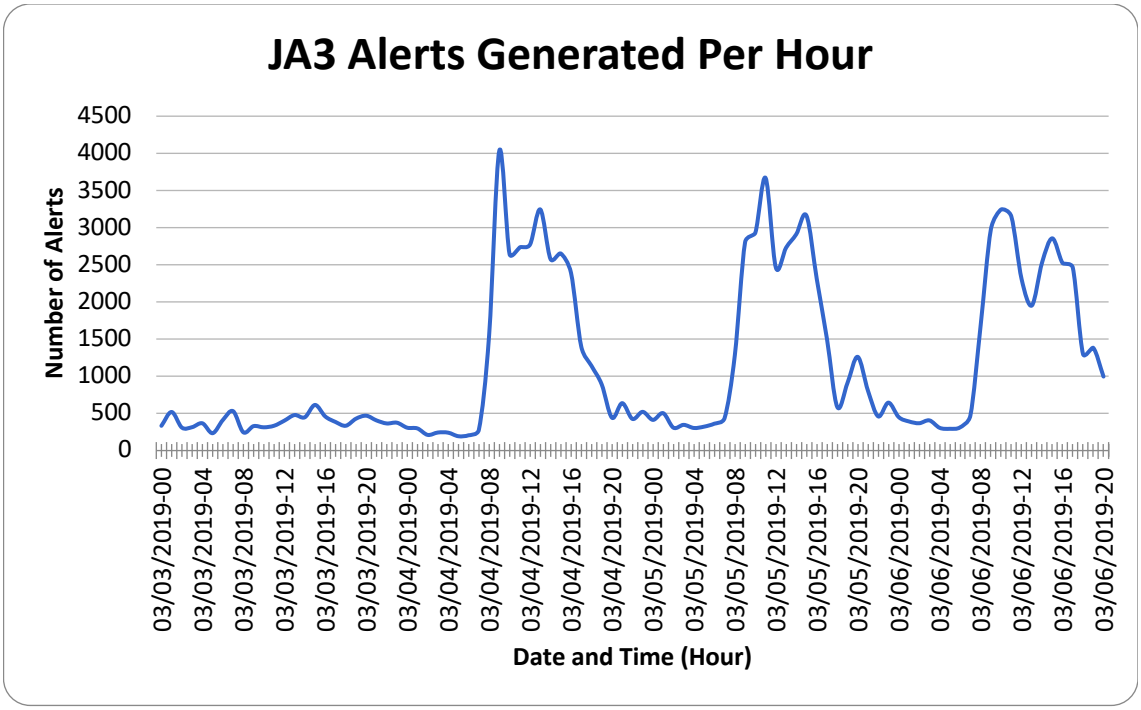

Figure 9. The number of alerts generated each day per hour

As seen in Figure 7, the default rule alerts generated are generic and not as detailed as the JA3 log. The log displays the timestamp, source IP hash and port, rule message, priority, destination IP and port. The JA3 logs display more features including the malware family, SSL features (SNI, Organizational Unit (OU), Common Name (CN), Certificate Authority, Locality (L), SHA1 hash, State or Province Name (ST), Issuer domain name (Issuerdn), Serial, and the TLS version. The rest of the information remains the same besides the additional information listed above. This allows us to learn more about the destination.

While analyzing the data, it was noted that most of the alerts generated were false positives. The reason is for this is since JA3 fingerprints are not unique, some of them may be fingerprints for legitimate browsers, email applications, for example. Most of the communications were going to legitimate URLs such as Fleep, Microsoft (and their subdomains), Facebook, etc. The number of the alerts generated for these clients were extremely high even compared with the university rule alerts that it was not reasonable to deduce that there was malicious activity. There were 107,222 alerts generated by Suricata with JA3 enabled for 694 IP addresses. There were four malware families that existed within the JA3 alerts as seen in Table 2 below. As for the university rule log, there were fifty alerts generated for fifty IP addresses. There were twenty-three source IP addresses that occurred in both the university rule log and in the JA3 enabled log.

From these twenty-three IP addresses the alerts generated for each IP address were analyzed by checking the timestamps from the university rule log. The JA3 log was examined for the timestamp occurring either fifteen minutes before or after (if it is not exact) and checked to see if during that timeframe the same IP address had any alerts generated. If it did, the alerts from the university rule log and the JA3 log were checked to determine if they were related to one another. The information contained in the alerts were analyzed (destination IP address and SNI from JA3 log and destination IP address from university rule log) to figure out if a relationship exists between the two alerts. After that, both destination IP addresses were searched in Hybrid Analysis and VirusTotal Intelligence to see what other communications existed for each of these IP addresses and if any were malicious. This process was done for all twenty-three IP addresses. The URLs in the SNI fields cannot be assumed that they are legitimate, they should be validated in Hybrid Analysis and VirusTotal Intelligence and also that the SNI resolves to the corresponding IP address.

One IP address was found that had an SNI of 's1.driverboosterscan.com' and was searched in VirusTotal Intelligence. VirusTotal Intelligence displayed that malware is using this URL to communicate with other malicious IP addresses. Each alert generated is assigned a Signature ID (SID), which can be seen in the university rule log. A signature ID, also known as SID "gives every signature its own id" and this ID is a number with the format sid:123 [65]. The SID can be searched in the JA3 Fingerprint ruleset and each SID is unique and contains a reference link (highlighted in Figure 10) which gives more information about that specific type of malware. This link shows other malware samples along with the destination IP and port number the client is trying to connect to. In the case of s1.driverboosterscan.com, the two destination IP addresses that the SSL client is trying to connect to are 35.156.43.90 (from the JA3 log) to port 443 and 93.184.221.133 (from the university rule log) to port 80.

The SID was searched (2017365) in the JA3 fingerprint ruleset to obtain the reference link that gave us more detailed information. The takeaway from this page was that this JA3 had been blacklisted. When searching this SNI (s1.driverboosterscan.com) in google, a Hybrid Analysis page was found that consisted both of the destination IP addresses that were mentioned in the default and JA3 logs [66].

During this cross validation of the twenty-three IP addresses, only one was found to exist in both custom and JA3 alerts. The other twenty-two IP address could not be easily

found having alerts generated around the same timestamp. This was a time-consuming process to just only find one IP address that might be suspicious. This is not a suggested methodology to search for possible malicious activity.

```
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"906004246f3ba5e755b043c057254a29"; reference
sslbl.abuse.ch/ja3-fingerprints/906004246f3ba5e755b043c057254a29/; sid:906200000; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"fd80fa9c6120cdeea8520510f3c644ac"; reference
sslbl.abuse.ch/ja3-fingerprints/fd80fa9c6120cdeea8520510f3c644ac/; sid:906200001; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"f6fd83a21f9f3c5f9ff7b5c63bbc179d"; reference
sslbl.abuse.ch/ja3-fingerprints/f6fd83a21f9f3c5f9ff7b5c63bbc179d/; sid:906200002; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (TrickBot)"; ja3_hash; content:"c50f6a8b9173676b47ba6085bd0c6cee"; referer
sslbl.abuse.ch/ja3-fingerprints/c50f6a8b9173676b47ba6085bd0c6cee/; sid:906200003; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Gozi)"; ja3_hash; content:"c201b92f8b483fa388be174d6689f534"; reference:u
sslbl.abuse.ch/ja3-fingerprints/c201b92f8b483fa388be174d6689f534/; sid:906200004; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"b90bdbe961a648f0427db21aaa6ccb59"; reference
sslbl.abuse.ch/ja3-fingerprints/b90bdbe961a648f0427db21aaa6ccb59/; sid:906200005; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"9f62c4f26b90d3d757bea609e82f2eaf"; reference
sslbl.abuse.ch/ja3-fingerprints/9f62c4f26b90d3d757bea609e82f2eaf/; sid:906200006; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Gozi)"; ja3_hash; content:"57f3642b4e37e28f5cbe3020c9331b4c"; reference:u
sslbl.abuse.ch/ja3-fingerprints/57f3642b4e37e28f5cbe3020c9331b4c/; sid:906200007; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Ransomware)"; ja3_hash; content:"2d8794cb7b52b777bee2695e79c15760"; refer
sslbl.abuse.ch/ja3-fingerprints/2d8794cb7b52b777bee2695e79c15760/; sid:906200008; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"7c410ce832e848a3321432c9a82e972b"; reference
sslbl.abuse.ch/ja3-fingerprints/7c410ce832e848a3321432c9a82e972b/; sid:906200009; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"96eba628dcb2b47607192ba74a3b55ba"; reference
sslbl.abuse.ch/ja3-fingerprints/96eba628dcb2b47607192ba74a3b55ba/; sid:906200010; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"9c2589e1c0e9f533a022c6205f9719e1"; reference
sslbl.abuse.ch/ja3-fingerprints/9c2589e1c0e9f533a022c6205f9719e1/; sid:906200011; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"849b04bdbd1d2b983f6e8a457e0632a8"; reference
sslbl.abuse.ch/ja3-fingerprints/849b04bdbd1d2b983f6e8a457e0632a8/; sid:906200012; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"590a232d04d56409fab72e752a8a2634"; reference
sslbl.abuse.ch/ja3-fingerprints/590a232d04d56409fab72e752a8a2634/; sid:906200013; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"5e573c9c9f8ba720ef9b18e9fce2e2f7"; reference
sslbl.abuse.ch/ja3-fingerprints/5e573c9c9f8ba720ef9b18e9fce2e2f7/; sid:906200014; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"93d056782d649deb51cda44ecb714bb0"; reference
sslbl.abuse.ch/ja3-fingerprints/93d056782d649deb51cda44ecb714bb0/; sid:906200015; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Dridex)"; ja3_hash; content:"d6f04b5a910115f4b50ecec09d40a1df"; reference
sslbl.abuse.ch/ja3-fingerprints/d6f04b5a910115f4b50ecec09d40a1df/; sid:906200016; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"1543a7c46633acf71e8401baccbd0568"; reference
sslbl.abuse.ch/ja3-fingerprints/1543a7c46633acf71e8401baccbd0568/; sid:906200017; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"16efcf0e00504ddfeddel3bfea997952"; reference
sslbl.abuse.ch/ja3-fingerprints/16efcf0e00504ddfeddel3bfea997952/; sid:906200018; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"7691297bcb20a41233fd0a0baa0a3628"; reference
sslbl.abuse.ch/ja3-fingerprints/7691297bcb20a41233fd0a0baa0a3628/; sid:906200019; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"92579701f145605e9edc0b01a901c6d5"; reference
sslbl.abuse.ch/ja3-fingerprints/92579701f145605e9edc0b01a901c6d5/; sid:906200020; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Adware)"; ja3_hash; content:"b2b61db7b9490a60d270ccb20b462826"; reference
sslbl.abuse.ch/ja3-fingerprints/b2b61db7b9490a60d270ccb20b462826/; sid:906200021; rev:1;)
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash; content:"4d7a28d6f2263ed61de88ca66eb011e3"; reference
sslbl.abuse.ch/ja3-fingerprints/4d7a28d6f2263ed61de88ca66eb011e3/; sid:906200022; rev:1;)
```

Figure 10. Suricata JA3 Fingerprint Ruleset that provides additional information for each fingerprint

A screenshot of the alerts generated for the IP address that had suspicious communication that was discussed above is displayed in

```
03/04/2019-13:26:56.648489  [**] [1:2019714:10] ET CURRENT_EVENTS Terse alphanumeric executable downloader
high likelihood of being hostile [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP}
IP=7dcee1289ada29bdd30ae239d0cb4cf8:1391 -> 93.184.221.133:80
```

Figure 11 and

```
03/04/2019-13:27:16.194687  [**] [1:906200053:1] SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)
[**] [Classification: (null)] [Priority: 3] {TCP} IP=7dcee1289ada29bdd30ae239d0cb4cf8:1415 -> 35.156.43.90:443
"03/04/2019-13:27:16.194691 IP=7dcee1289ada29bdd30ae239d0cb4cf8:1415 -> 35.156.43.90:443  TLS:
Subject='CN=*.driverboosterscan.com' Issuerdn='C=US, O=Amazon, OU=Server CA 1B, CN=Amazon' SHA1='ef:ee:91:18:3c:
04:b6:6a:40:7b:44:0b:94:68:aa:91:58:44:27:81' SNI='s1.driverboosterscan.com' SERIAL='09:C3:ED:5C:
01:65:F0:F1:47:85:BE:DB:73:ED:D9:5B' VERSION='TLS 1.2' NOTBEFORE='2018-10-31T00:00:00'
NOTAFTER='2019-11-30T12:00:00'"
```

Figure 12. This was enough evidence to conclude that the communication is suspicious and out of these twenty-three IP addresses, only one of the IPs appeared to have malicious activities. The other twenty-two IP addresses did not have timestamps that were relatively close to one another therefore no further analysis was conducted. The alerts from the JA3 enabled ruleset and custom rules were not generated close in the manner of time to one another that would make one to believe that further investigation was needed. The alerts from the custom rule alerts and the JA3 alerts may not be related to each other.

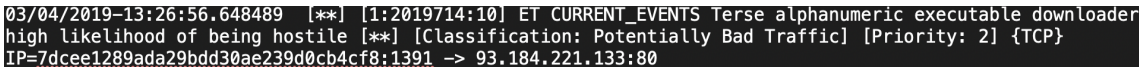

Figure 11. Custom rules alert of the IP address that had suspicious SSL communication

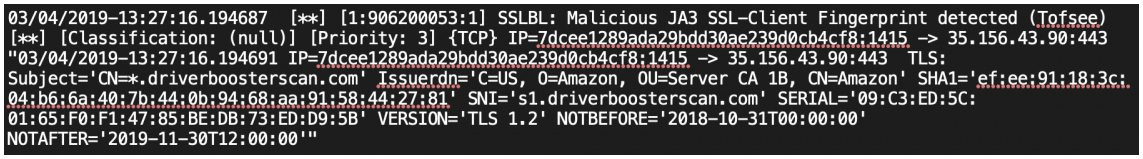

Figure 12. JA3 alert of the IP address that had suspicious SSL communication

Table 2. Malware families within the JA3 alerts

Malware Family	**Number of Alerts**
Adware	76201
Adwind	110
JBitfrost	40
Tofsee	30871

The priority keyword has a numeric value that ranges from 1 till 255 however the values 1 to 4 are more commonly used and the highest priority is 1. Signatures that have a higher priority will always be inspected first. [65].

In this research, all of the JA3 alerts generated had a priority of 3 with a classification of null, while the priority levels in the university rule log were scattered between 1 to 3. A classification of null means that the particular alert does not have any descriptive information about what type of alert has been generated. Six of the alerts had a priority level of 3, 172 alerts had a priority level of 2 and 299 alerts had a priority level of 1. The classifications of the university log rule are shown in Table 3. The priority levels and classifications of the malicious JA3 fingerprints have been assigned by abuse.ch. The classification in the default rules log give a generic overview of why the alert has been generated.

Table 3. Classification types and how many alerts were generated for each in the university rule log

Number of Alerts	**Classification Type**
2	Attempted Administrator Privilege Gain
6	Miscellaneous Activity
172	Potentially Bad Traffic
297	A Network Trojan was Detected

The JA3 alerts are vital to pay attention to because they are supply valuable indicators that could determine if a specific client is malicious or legitimate. In Bahnsen, Camacho, and Torroledo's study, they focused on "what information is contained or implicit in a certificate to make it trustworthy, keeping in mind that certificates with less information are more suspicious" [67]. The same can be said in this study, but about the amount of information displayed in the Suricata alerts. The research [67] demonstrate that the less information in the Suricata JA3 enabled alerts, the more suspicious it is. The fields in the SSL ClientHello packet is necessary to have in order to be able to identify the client.

One IP address was found that had an interesting alert that was only seen in the JA3 log. The only information that was given in the alert was the SNI field. At a first glimpse, the alert looked suspicious due to the lack of any other information besides the SNI. The information in the SNI field was also not a familiar URL. This alert was the only one out of all the alerts that did not have any other SSL features. When the SNI was searched in VirusTotal Intelligence, it showed that malware is using this URL for malicious communication. The important point to understand here is that the lack of SSL features is a signal that malicious communication may exist as also mentioned in this study [67]. The alert is shown below:

```
03/05/2019-23:12:54.000994 IP=2cfa8601c7c76d3f4a838cb8a74d3946:8146 ->
47.89.76.72:443  TLS: SNI='aligtr015.mmstat.com' VERSION='UNDETERMINED'
```

Figure 13. JA3 alert that did not have any fields besides SNI

Some statistics were generated to understand the JA3 alerts further. Table 4 below displays the number of IPs that had alerts for the corresponding SIDs of the JA3 ruleset. These signature IDs can be searched in the JA3 fingerprint ruleset [22] to determine the malware family and a reference URL is given to show more information about that specific malware as to whether or not it is blacklisted. This table presents which SID existed the most within the university's network.

Table 4. Number of IP addresses that had alerts with the corresponding SID based of the JA3 ruleset

Number of IPs that had alerts for the corresponding SID	**SID**
1	906200020
1	906200033

Number of IPs that had alerts for the corresponding SID	SID
1	906200045
1	906200053
2	906200027
2	906200034
3	906200017
4	906200001
5	906200050
6	906200025
8	906200009
8	906200037
9	906200019
9	906200039
9	906200055
12	906200041
13	906200024
15	906200036
16	906200030
19	906200022
19	906200040
19	906200051
21	906200015
21	906200056
31	906200023
41	906200044
48	906200047
49	906200063
52	906200048
72	906200042
115	906200014
141	906200043
186	906200038
208	906200032

Sandnet is a portal by Abuse.ch where information such as an IP address, domain name, MD5 hash, JA3/JA3S fingerprint, SID, or SSL certificate fingerprint can be searched to acquire more information [68]. Abuse.ch provided access to their portal for this thesis. This portal can be used to search for an IP address, JA3 and JA3S fingerprint. It was used to attain more information about the search term given. Information such as the referencing malware samples can be seen if the JA3/JA3S fingerprint is searched, IDS alerts displaying the signature (malware type), the score given by VirusTotal and source of the malware. Figure 14 displays the result when entering the SID, as an example, to obtain more information about a specific malware.

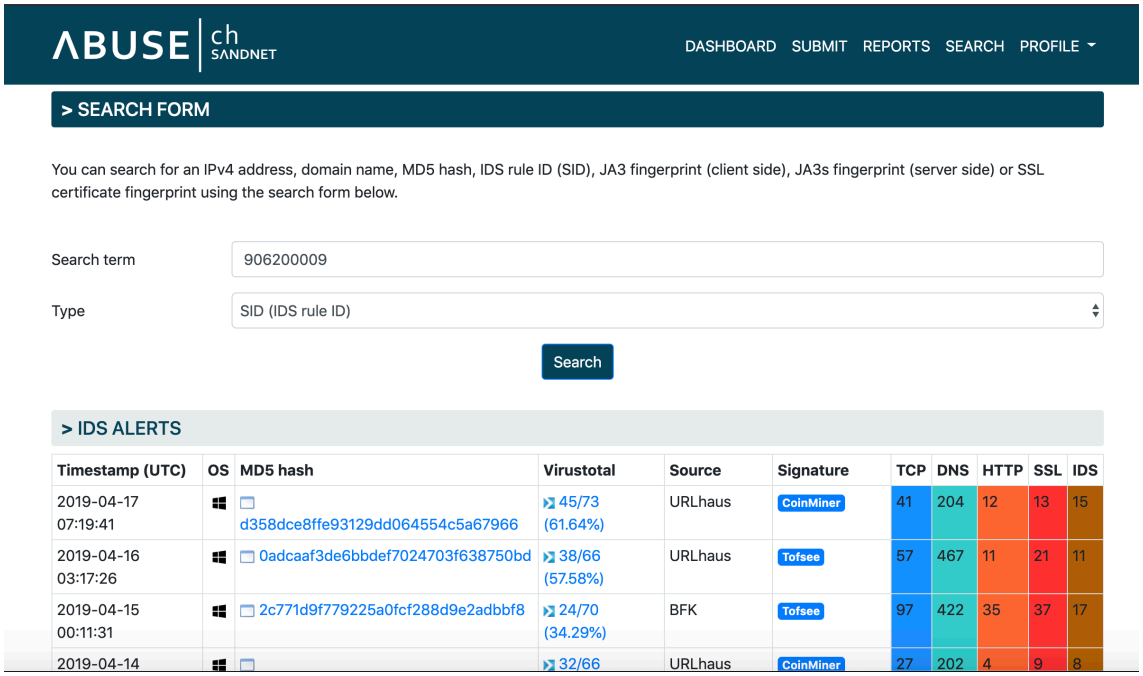

Figure 14. Search result of using the SID as the search term from Abuse.ch

Error! Not a valid bookmark self-reference. displays the number of alerts generated per SID. This information shows that over 24,000 alerts were generated for the last SID. It can be seen that the last two SIDs 906200014 and 906200038 corresponded with the Adware malware and SID 906200024 corresponded to the Tofsee malware generated the most alerts in TalTech's traffic. This is an indicator of possible false positive because most of the traffic coming from the SIDs that generated the most alerts were going to domains such as fleep.io, *.microsoft.com, *.skype.com which are not known for hosting malicious content or being domain frontable. This revealed that there was not only one specific malware generating the majority of the alerts and therefore these three hashes composed 65% of the alerts generated throughout this experiment.

Table 5. Number of alerts for each signature ID

Number of alerts per SID	SID	Malware
1	906200020	Adware
2	906200027	Tofsee
2	906200034	Adware
3	906200053	Tofsee
15	906200025	Adware
16	906200050	Tofsee
36	906200019	Adware
38	906200033	Tofsee
40	906200039	JBifrost
81	906200009	Tofsee
105	906200041	Adware
110	906200063	Adwind
132	906200045	Adware
137	906200001	Tofsee
194	906200051	Tofsee
265	906200036	Adware
344	906200015	Adware
389	906200017	Tofsee
475	906200037	Adware
484	906200044	Adware
804	906200048	Adware
1157	906200022	Tofsee
1345	906200055	Tofsee
1398	906200056	Tofsee
1947	906200030	Tofsee
2201	906200043	Adware
2229	906200032	Tofsee
2640	906200040	Adware
6033	906200023	Adware
7399	906200047	Adware
7705	906200042	Adware
21935	906200024	Tofsee

Number of alerts per SID	SID	Malware
22687	906200014	Adware
24873	906200038	Adware

The results from this experiment have shown that using JA3 as the only determinant for intrusion detection cannot be useful. Just from analyzing log files from over the course of four days has shown a tremendous number of false positives. Even though there was one IP address that seemed to be suspicious it is not enough evidence to be able to conclude that JA3 is effective and is a feature that should always be enabled when monitoring network traffic. Therefore, another experiment was conducted using Bro instead of Suricata since it supports JA3S. This experiment will test whether or not it is more effective to combine JA3 with JA3S. This second experiment will look at JA3 and JA3S hashes and pair them together based on their communication.

5.2 Trial 2

There were 8.3 GB of normal traffic and 117 GB of malicious PCAP files and these were obtained from StratosphereIPS and abuse.ch. All of the normal traffic was taken from StratosphereIPS and these captures were generated by performing normal actions such as logging into social media accounts, installing applications, accessing news pages, accessing YouTube and playing music. Most of the normal traffic were generated by a normal user using a Linux or Windows operating system in a real environment. Some of the traffic are making connections to legitimate websites to capture the HTTPS traffic. Abuse.ch provided all of the malware samples along with their PCAP files that they had. This information can be found under each dataset, like this one [69], for example. There were 102,915 SSL communications from malicious traffic and 69,098 SSL communications from the normal traffic. Communication refers to the SSL sessions per record. In the log files, each row represents an SSL record. Table 6 below displays the number of JA3, JA3S, JA3-JA3S pair fingerprints in normal and malicious traffic.

Table 6. Number of fingerprints from normal and malicious traffic

Type of fingerprint	Normal	Malicious
JA3	13	233
JA3S	249	460

JA3-JA3S Pairs	527	1363

Table 7. Common fingerprint hashes that were seen in both logs

Type of Fingerprint	**Number of Types of Fingerprints that are seen in both normal and malicious traffic**
JA3	7
JA3S	134
JA3-JA3S Pairs	56

Table 8. Total number of SSL traffic communication

Type of Traffic	**Total number of SSL traffic communication**
Normal	69098
Malicious	102915

Table 7 displays the common number of JA3/JA3S/JA3-JA3S hash pairs that were seen in both normal and malicious traffic. These number will be used to obtain the percentages shown in Table 9 and the explanation of how to obtain these percentages will be addressed in the later sections.

Table 8 shows the total number of SSL traffic communication that was in normal traffic and malicious traffic. These percentages were calculated and shown in Table 9 and the formula used to obtain these percentages are shown in Table 10, Table 11, Table 13, Table 14. Figure 15 is a visual representation of Table 9.

Table 9. Percentages representing incorrectly detected hashes and hashes that will go undetected

Fingerprint Types	**% of normal hashes inaccurately detected as malicious**	**% of malicious hashes that will not be detected**	**% of normal traffic communication that contains hashes that have been inaccurately detected as malicious**	**% of malicious traffic communication that will go undetected**
JA3	53.85%	3.00%	99.53%	1.31%
JA3S	53.82%	29.13%	92.48%	32.00%
JA3-JA3S Pair	10.63%	4.11%	48.39%	1.05%

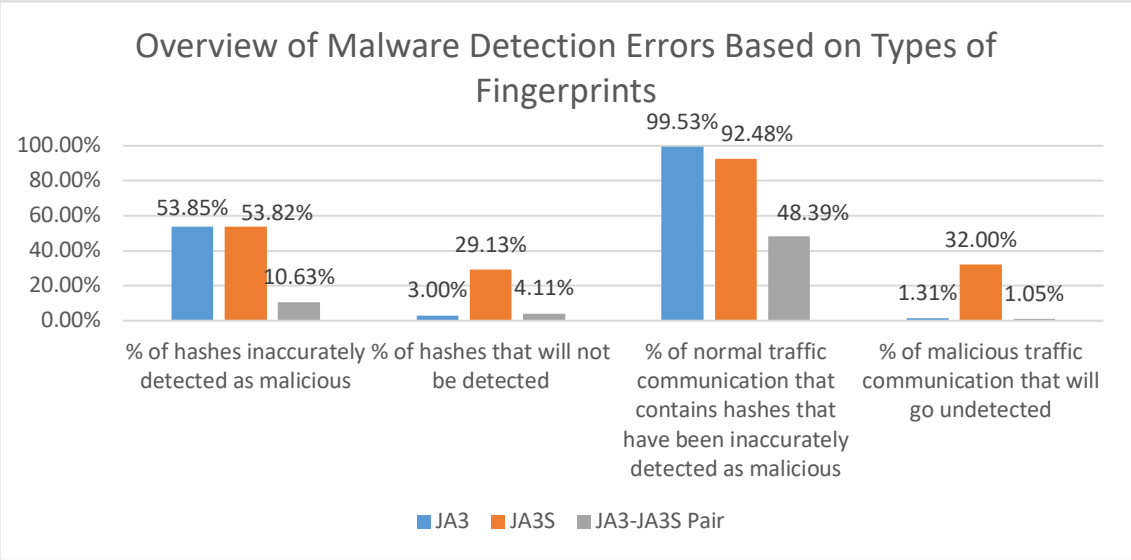

Figure 15. Overview of Malware Detection Errors Based on Types of Fingerprints

An interesting observation that can be deduced based off of Table 9 is that most of the traffic that has been inaccurately detected as malicious were *.skype.com and *.microsoft.com.

5.2.1 Percentage of hashes that have been inaccurately detected as malicious

This column represents the percentages of hashes that have been inaccurately detected as malicious. For each connection that is coming from the client an alert will be generated. Also, multiple IP addresses or clients may have the same JA3 fingerprint.

The following information below displays how the percentages in Table 9 were calculated:

- *N1* represents the number of JA3 fingerprints that were seen in both normal and malicious logs (see Table 7 for the N1 value)
- *N2* represents the number of JA3S fingerprints that were seen in both normal and malicious logs (see Table 7 for the N2 value)
- *N3* represents the number of JA3-JA3S fingerprint pairs that were seen in both normal and malicious logs (see Table 7 for the N3 value)
- *T1* represents the number of JA3 fingerprints that are normal (see Table 6)

- *T2* represents the number of JA3S fingerprints that are normal (see Table 6)
- *T3* represents the number of JA3-JA3S pairs that are normal (see Table 6)

Table 10. The formulas used to calculate the percentage of hashes that have been inaccurately detected as malicious.

Fingerprint Type	**Formula Used**
JA3	$\frac{N1 \times 100}{T1}$
JA3S	$\frac{N2 \times 100}{T2}$
JA3-JA3S pair	$\frac{N3 \times 100}{T3}$

5.2.2 Percentage of malicious hashes that will not be detected

There are some JA3, JA3S, JA3-JA3S pair fingerprints that were seen in both malicious and normal traffic. If these fingerprints are removed from the ruleset, then there is a chance that some of the malicious hashes will not be detected in the network. This second column in Table 9 represents the percentage of malicious hashes that will not be detected. This means that there is a possibility that the IDS engine will not detect malicious activity. Table 11 displays the formulas that were used to calculate these percentages.

The following information below displays how the percentages in Table 9 were calculated:

- The values of N1, N2, and N3 can be found in Table 7; these values represent the number of each of fingerprints that were seen in both normal and malicious logs
- The values of T4, T5, and T6 can be found in the second column of Table 6; these values represent the number of each fingerprint type that are malicious

Table 11. The formulas used to obtain the percentage of hashes that will not be detected

Fingerprint Type	**Formula Used**
JA3	$\frac{N1 \times 100}{T4}$

Fingerprint Type	Formula Used
JA3S	$\frac{N2 \times 100}{T5}$
JA3-JA3S pair	$\frac{N3 \times 100}{T6}$

5.2.3 Percentage of normal traffic communication that contains hashes that have been inaccurately detected as malicious

This column in Table 9 represents the amount of normal traffic within the network that has been incorrectly detected as malicious. The percentage is based on the number JA3, JA3S, and JA3-JA3S pair communications that are generated. For example, if an organization decided only to use JA3 fingerprints and rules to test against their network, then there is a 99.53% chance that most of the normal traffic has inaccurately been detected as malicious. The formula in Table 13 is used to calculate the percentages that are seen in the third column of Table 9.

- *N4* represents the number of normal traffic communications that existed where JA3 hashes were seen in both malicious and normal log (see Table 12 for the value of N4)

- *N5* represents the number of normal traffic communications that existed where JA3S hashes were seen in both malicious and normal log (see Table 12 for the value of N5)

- *N6* represents the number of normal traffic communications that existed where JA3-JA3S pair hashes were seen in both malicious and normal log (see Table 12 for the value of N6)

Table 12. The number of normal traffic that have been inaccurately detected as malicious, if the common hashes are not removed from the ruleset

Type of Fingerprint	Normal
JA3	68778
JA3S	63900
JA3-JA3S Pairs	33434

Table 13. The formulas used to obtain the percentage of normal traffic communication with hashes that have been inaccurately detected as malicious

Fingerprint Type	Formula Used
JA3	$\frac{N4 \times 100}{69098}$
JA3S	$\frac{N5 \times 100}{69098}$
JA3-JA3S Pair	$\frac{N6 \times 100}{69098}$

5.2.4 Percentage of malicious traffic communication that will go undetected if the hashes are removed from the malicious ruleset

This column in Table 9 represents the percentage of malicious traffic communication that will go undetected. This shows the percentage of traffic that will go undetected if the has hashes are removed from the malicious ruleset. The formula given in Table 14 is used to calculate the percentages shown in Table 9.

- *N7* represents the number of malicious traffic communications that existed where JA3 hashes were seen in both malicious and normal log (see Table 15 for N7 value)

- *N8* represents the number of malicious traffic communications that existed where JA3S hashes were seen in both malicious and normal log (see Table 15 for N8 value)

- *N9* represents the number of malicious traffic communications that existed where JA3-JA3S pair hashes were seen in both malicious and normal log (see Table 15 for N9 value)

Table 14. The formula used to obtain the percentage of malicious traffic communication that will go undetected

Fingerprint Type	Formula Used
JA3	$\frac{N7 \times 100}{102915}$
JA3S	$\frac{N8 \times 100}{102915}$
JA3-JA3S pair	$\frac{N9 \times 100}{102915}$

Table 15. The number of malicious traffic will go undetected if the common hashes are removed from the malicious ruleset

Type of Fingerprint	Malicious
JA3	1353
JA3S	32932
JA3-JA3S Pairs	1082

5.2.5 JA3/JA3S/JA3-JA3S Hash Comparisons

This study was designed to detect malicious traffic based on the JA3, JA3S, JA3S pair hashes therefore two decisions had to be made to either include the common hashes between malicious and normal or not to include them. Each choice has its advantages and disadvantages.

1. JA3

If the IDS rules were made based on the JA3 hashes that were seen in both normal and malicious traffic and if it is not removed from the JA3 ruleset, then there will be a 53.85% chance of the normal JA3 fingerprint hashes marked as malicious (the number of JA3 fingerprint that have been inaccurately detected as malicious). This can be seen in Table 9.

Removing the JA3 hashes that are seen in both normal and malicious traffic will have an effect on the detection rate. There is a possibility that it will not be able to detect 3.00% of the malicious JA3 hashes. This can be seen in Table 9.

2. JA3S

If the IDS rules were made based on the JA3S hashes that were seen in both normal and malicious traffic and if they are not removed from the ruleset, then there is a 53.82% chance of the normal JA3S fingerprints marked as malicious. This can be seen in Table 9.

If the IDS rules were made based on the JA3S hashes that are seen in both normal and malicious JA3S and if they are removed, there is a chance that the IDS engine will not be able to detect 29.13% of the malicious JA3S hashes. This can be seen in Table 9.

3. JA3-JA3S Pairs

If the IDS rules were made based on the JA3-JA3S pair hashes that were seen in both normal and malicious JA3-JA3S pair and if they are not removed, then there is a 10.63% chance of the normal JA3-JA3S pairs marked as malicious. This can be seen in Table 9.

If the IDS rules were made based on the JA3-JA3S pair hashes that were seen in both normal and malicious JA3 and if removed, there is a possibility that 4.11% of the traffic will go undetected. This can be seen in Table 9.

There was a total of 56 JA3-JA3S pairs that appeared in both normal and malicious traffic and from these there were only six unique JA3 fingerprints. These fingerprints were individually searched in the Sandnet portal and from the trisulnsm GitHub page [70]. This was done to learn more information about SSL clients. It turned out that three of these clients were browsers and that they could be removed from the rulesets to lessen the number of false positives. The other three fingerprints were inconclusive because the status of these in both sources was unknown. The network traffic that had JA3 fingerprints as Firefox were removed leaving only twelve JA3-JA3S pairs in both malicious and normal traffic. The percentage of false positive given by JA3-JA3S pairs is still much less than considering JA3 and JA3S alone.

This test showed that JA3S combined with JA3 is more effective than JA3 alone and that the number of false positives was much lower. This experiment demonstrated that JA3-JA3S pair reduces the number of false positives in the network traffic. However, at the same time there is still a possibility that false positives will still exist.

5.3 Trial 3

Trial three utilized the decision tree algorithm to predict the accuracy of a dataset by training from the data that has been given to the algorithm and the code used the classification technique. Although this study was not focused on machine learning, one of the supervised algorithms, decision tree, utilized the original values and not the hashes to see if machine learning can be used to create a detection system. This algorithm “can be used to predict class or value of target variables by learning decision rules inferred from training data” [71]. The decision tree was the chosen algorithm for trial three because they are “relatively efficient to learn and are easy to interpret, i.e., a

set of rules can be associated with each output" [72]. However, there is "a high probability of overfitting in decision trees" and it "gives a low prediction accuracy for a dataset as compared to other machine learning algorithms" [71]. There were other algorithms that could have been used for this trial, but decision tree was the most logical. It makes decisions based off the root node, which in this case is, "Is the network traffic normal or malicious?".

The purpose of this trial was to see how accurate a machine learning algorithm could classify the fingerprints as malicious or normal. The important aspect here is that this trial used the data of the ClientHello packet. As mentioned in 2.9, JA3 fingerprints are generated by concatenating the five ClientHello packet fields (SSL version, ciphers, extensions, elliptic curves, and elliptic curve point formats) together and converting them an MD5 hash. It is assumed that during the conversion there is a chance that some data has been lost. Therefore, this experiment was done to see if preserving the original values of the fields would change the classification accuracy.

The Pandas package was used to read the CSV data which is then split into training data and testing data by using the train_test_split function. In this study, seventy percent of the data was used for training while the rest of the data (thirty percent) had been denoted to testing. The algorithm learns how to detect which fingerprints were normal or malicious because they will apply what they have learned from the training data onto the test data.

Two different CSV files were given to the algorithm to make a comparison between the accuracy of classification percentages. Both of the CSV files have been randomized, the difference is that one CSV file contained the duplicates of the SSL records (as can be seen in 5.3.1) while the other had the duplicated SSL records removed.

5.3.1 Trial 3A

The first CSV file that was processed by the algorithm contained duplicated SSL records. The rows in the CSV file have been randomized. The randomization of records has been done so that the machine learning algorithm can get an equal amount of normal and malicious fields to train and test.

The confusion matrix generated by the decision tree algorithm is shown in Table 16. 9381 JA3-JA3S fingerprints were predicted correctly with their label, 1132 JA3-JA3S fingerprints were predicted as malicious when they were actually not, 1361 JA3-JA3S

fingerprints were predicted as normal when they were actually malicious, and 33645 JA3-JA3S fingerprints were actually negative and were correctly predicted.

Table 16. Confusion Matrix that contained duplicate SSL fields

True Positive 9381	**False Positive** 1132
False Negative 1361	**True Negative** 33645

The algorithm displays the accuracy for classification and in this trial, there was a classification accuracy of 94.5%, this meant that approximately 5.5% was inaccurately classified. It is important to mention that the duplicated SSL records were valuable for this experiment. The duplicated SSL records helped the machine learning algorithm assign the same classification value to the duplicated SSL records each time it was seen in the dataset. The reason for this is because the algorithm has already seen the same exact SSL record in the training set, therefore it did not have to guess the label for that record in the test set.

The classification report is shown below.

Table 17. Classification Report

	Precision	**Recall**	**f1-score**	**Support**
0 (Normal Traffic)	0.87	0.89	0.88	10513
1 (Malicious Traffic)	0.97	0.96	0.96	35006
Micro avg	0.95	0.95	0.95	45519
Macro avg	0.92	0.93	0.92	45519
Weighted avg	0.95	0.95	0.95	45519

As seen in Table 17, when the decision tree algorithm predicts SSL records as normal it is correct 87% of the time and when the algorithm predicts SSL records as malicious it is correct 97% of the time. The recall for normal traffic was 0.89 which meant that the algorithm was able to correctly identify 89% of the traffic as normal. The recall for malicious traffic was 0.96 which meant that the algorithm was able to correctly identify 96% traffic as malicious. As a result, this would lower the number of false positives as the algorithm was able to identify malicious much more than normal traffic.

The f1-score is 0.96 which is a good score since the best f1-score one can have is 1. This indicates that a low number of false positive and false negative exists, which means that the algorithm correctly distinguished the malicious traffic in the network.

The support value for the normal traffic, 10513, is the addition of the true positive and false positive values taken from the confusion matrix.

The support value for the malicious traffic, 35006, is the addition of the false negative and true negative values taken from the confusion matrix.

5.3.2 Trial 3B

As mentioned earlier, the second CSV file had the duplicated SSL records removed and randomized. The confusion matrix generated for this CSV file is shown in below.

Table 18. Confusion Matrix for the CSV file that had duplicate SSL fields removed

True Positive 103	**False Positive** 66
False Negative 44	**True Negative** 339

103 JA3-JA3S fingerprints were predicted correctly with their label, 66 JA3-JA3S fingerprints were predicted as malicious when they were actually not, 44 JA3-JA3S fingerprints were predicted as normal when they were actually malicious, and 339 JA3-JA3S fingerprints were actually negative and were correctly predicted.

The classification accuracy was 80.1% which is less than the classification accuracy of 5.3.1. It can be implied that the training data set only had unique SSL records. When labeling the SSL records in the test dataset the algorithm could not rely on the training dataset because all of the SSL records in the test dataset have not been seen before. Therefore, the classification accuracy has decreased compared to when there were duplicate SSL records in the training and test sets.

The classification report generated for this CSV file is shown below.

Table 19. Classification Report generated for CSV file without duplicate SSL records

	Precision	**Recall**	**f1-score**	**Support**
0 (Normal Traffic)	0.56	0.51	0.53	146

	Precision	Recall	f1-score	Support
1 (Malicious Traffic)	0.83	0.86	0.84	406
Micro avg	0.77	0.77	0.77	552
Macro avg	0.70	0.68	0.69	552
Weighted avg	0.76	0.77	0.76	552

As seen in Table 19, when the decision tree algorithm predicts SSL records as normal it is correct 56% of the time and when the algorithm predicts SSL records as malicious it is correct 83% of the time. The recall for normal traffic was 0.51 which meant that the algorithm was able to correctly identify 51% of the traffic as normal. The recall for malicious traffic was 0.86 which meant that the algorithm was able to correctly identify 86% traffic as malicious. Again, as a result, this would lower the number of false positives as the algorithm was able to identify malicious much more than normal traffic.

The f1-score for the malicious traffic is 0.84 which is relatively good because that means that there will not be as much false positives or false negatives. Although some false positive and false negative may still appear in the network it is still quite a high score since the value is close to 1.

The support value for the normal traffic, 146, is the addition of the true positive and false positive values taken from the confusion matrix.

The support value for the malicious traffic, 406, is the addition of the false negative and true negative values taken from the confusion matrix.

5.4 Machine Learning vs. TLS Fingerprint Hashes

Machine learning was used in this study to compare the accuracy of detecting malicious activity. After completing all three trials and performing full analysis, it seems that machine learning was the better approach for this research. It is preserving the original values of the SSL ClientHello fields that are used to produce fingerprints. Since the algorithm trains a percentage of data before performing on the test data, this will help the algorithm recognize patterns to be able to decide if these fields are considered malicious or normal. Machine learning has the capability to learn and guess new SSL fields that it has not seen in the training data set, but in the test set. This will allow it to make the best guess it can about the SSL fields whereas in TLS fingerprinting, the

fingerprint would be inconclusive. Also, as the algorithm gets to work with more data, it can improve its accuracy of classification by training itself with variety of data that is given.

In TLS fingerprinting, the values of the SSL fields get transformed to an MD5 hash value. However, it is possible that during this process, some of the data may be lost. JA3 fingerprinting is heavily dependent on predefined whitelists and blacklists therefore causing many inaccurate classifications as to whether a fingerprint is normal or malicious. This means that if the fingerprint does not exist in the blacklist or whitelist database, then the IDS engine will not be able to detect if a specific client application is normal or malicious.

The results from trial 3 displays that machine learning can do a better job of detecting malicious traffic than TLS fingerprinting and would reduce the number of false positives. In both experiments, the percentages of accurately identifying malicious traffic as malicious was much higher than identifying legitimate traffic.

5.5 Limitations

One of the limitations of this study is the size of the ruleset that is being used to test against the university's network traffic. One other limitation with trial one was that it was not feasible, due to time constraints, to go to each computer system in the university to analyze how the packet was generated and the type of client application that sent this packet. As mentioned earlier as a limitation of JA3, client applications using the same library or OS socket may have the same JA3 fingerprint therefore these fingerprints alone will not be useful in security monitoring. This created a high number of false positives. There is a possibility that although the custom rules generated an alert, the Suricata instance with JA3 enabled may not have generated an alert because a rule does not exist for that specific malicious traffic. An additional limitation was time and due to the limited amount of time it was not possible to capture real-world network traffic using Bro with JA3 and JA3S to analyze the effectiveness of JA3 combined with JA3S.

It was also not possible to test out other types of machine learning algorithm to classify which traffic was malicious or normal due to time. Two network traffics were used during Trial three's experiment – one with only malicious traffic and one for only the

normal dataset. In reality, it should have been one network traffic file that contained a mixture of both malicious and normal traffic, but there were not any public mixture available datasets available. If a large network traffic were to exist, it also would have been challenging to differentiate between malicious and normal traffic.

6 Conclusion

Monitoring of encrypted communication is important. The goal of this research was to decide whether JA3 fingerprinting or preserving the ClientHello fields used to generate the fingerprint would be useful in security monitoring. This will help recognize malicious activities that are going on in the network. The initial experiment proved that JA3 fingerprints generated a lot of false positives in the alerts. Based on previous studies that have been done on this topic, this was the first experiment that tested JA3 fingerprinting in a real-world network. The scope of this study was extended by testing out how effective JA3S would be combined with JA3 and looking at how preserving the fields that create the JA3 fingerprints would give a more accurate decision of detecting malicious or suspicious activity.

Results indicated that the detection accuracy of the JA3-JA3S pairs were more efficient than JA3 alone and utilizing machine learning with the preserved fields from the ClientHello packet gave an improved classification between malicious and legitimate traffic. The detection accuracy calculated by the machine learning model demonstrated that utilizing the original values from the fields that are used to produce the fingerprints can be of better help than JA3 fingerprinting in security monitoring. JA3 and JA3-JA3S pair fingerprints demonstrated that it was not as reliable on its own in comparison to preserving the original values.

In future research, more research is needed to apply and test JA3-JA3S fingerprinting pairs. As JA3-JA3S pairs seemed to be more effective it is important to test this in a real-world network. This will help get a better understanding of the pairs and how beneficial they may be to detect malicious traffic. This research only focused on the ClientHello packet fields that were mentioned by John Althouse and the rest of the developers of JA3/JA3S, however future studies could look into other fields of the ClientHello packet to determine if it would be of value. Preserving the original values with machine learning rather than hashing may be a more helpful feature in security monitoring. As this research only focused on one type of supervised algorithm, it could be possible that other supervised algorithms give much more desirable results.

Since this thesis focused on the detection accuracy of JA3 and JA3S and their original values, the effect of how JA3 and JA3S affect the performance of monitoring systems other than detection accuracy such as relation to other detection rules was not considered. Identifying the source of how these hashes were generated still remain an issue. As mentioned earlier, some SSL client applications may utilize the same libraries such as Python and OS socket. It is important to identify the hashes that belong to specific libraries to further understand how these hashes are generated. Identifying these hashes might help in preventing false positives in the network. This way, JA3 and JA3S fingerprinting could be meaningful in security monitoring.

References

[1] J. Maddison, “Encrypted Traffic Reaches A New Threshold | IT Infrastructure Advice, Discussion, Community - Network Computing,” 2018. [Online]. Available: https://www.networkcomputing.com/network-security/encrypted-traffic-reaches-new-threshold. [Accessed: 15-Apr-2019].

[2] J. Kohout, T. Komárek, P. Čech, J. Bodnár, and J. Lokoč, “Learning communication patterns for malware discovery in HTTPs data,” *Expert Syst. Appl.*, vol. 101, pp. 129–142, 2018.

[3] T. Kovanen, G. David, and T. Hämäläinen, “Survey: Intrusion Detection Systems in Encrypted Traffic,” 2016.

[4] V. T. Goh, J. Zimmermann, and M. Looi, “Towards intrusion detection for encrypted networks,” *Proc. - Int. Conf. Availability, Reliab. Secur. ARES 2009*, pp. 540–545, 2009.

[5] “Detect encrypted malware traffic and secure network - Cisco.” [Online]. Available: https://www.cisco.com/c/en_uk/solutions/enterprise-networks/network-refresh-guidance/secure-your-network-by-detecting-encrypted-malware-traffic-with-machine-learning.html. [Accessed: 06-May-2019].

[6] J. Althouse, John; Atkinson, Jeff; Atkins, “salesforce/ja3.”

[7] Intel & Analysis Working Group, “What is Cyber Threat Intelligence?” [Online]. Available: https://www.cisecurity.org/blog/what-is-cyber-threat-intelligence/. [Accessed: 06-May-2019].

[8] “What is Security Monitoring? – HPE Definition Glossary | HPE EUROPE.” [Online]. Available: https://www.hpe.com/emea_europe/en/what-is/security-monitoring.html. [Accessed: 07-May-2019].

[9] Holm Security, “Security monitoring.” [Online]. Available: https://www.holmsecurity.com/security-monitoring. [Accessed: 07-May-2019].

[10] D. Cooper and W. Polk, “Network Working Group,” 2008.

[11] M. Korczyński and A. Duda, *Markov chain fingerprinting to classify encrypted traffic*. 2014.

[12] K. Cabaj, L. Caviglione, W. Mazurczyk, S. Wendzel, A. Woodward, and S. Zander, “The new threats of information hiding: The road ahead,” *IT Prof.*, vol. 20, no. 3, pp. 31–39, 2018.

[13] B. Anderson, S. Paul, and D. McGrew, “Deciphering malware’s use of TLS (without decryption),” *J. Comput. Virol. Hacking Tech.*, vol. 14, no. 3, pp. 195–211, 2018.

[14] M. Carling, C. Security, and T. Organisation, “TLS Exposed,” no. October, 2017.

[15] M. Husák, M. Čermák, T. Jirsík, and P. Čeleda, “HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting,” *Eurasip J. Inf. Secur.*, vol. 2016, no. 1, pp. 1–14, 2016.

[16] “Network Traffic Analysis,” 2018.

[17] A. Razaghpanah, J. Amann, K. G. Pater-Son, N. Vallina-Rodriguez, and J. Caballero, “Coming of Age: A Longitudinal Study of TLS Deployment,” p. 14, 2018.

[18] “SSL/TLS Client Fingerprinting for Malware Detection,” 2017. [Online].

Available: https://www.securitynewspaper.com/2017/07/27/ssltls-client-fingerprinting-malware-detection/. [Accessed: 30-Jan-2019].
[19] V. Rajagopal, “TLS Fingerprinting techniques using TrisulNSM,” 2017. [Online]. Available: https://medium.com/@vivekrj/tls-fingerprinting-techniques-using-trisulnsm-d0b845dd4378. [Accessed: 30-Jan-2019].
[20] T. Dierks and E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.2,” 2008.
[21] P. Mieden, “Implementation and evaluation of secure and scalable anomaly-based network intrusion detection Philipp Mieden,” no. December, 2018.
[22] “Suricata JA3 Fingerprint Ruleset.” [Online]. Available: https://sslbl.abuse.ch/blacklist/#ja3-fingerprints-csv.
[23] J. Althouse, “Open Sourcing JA3,” 2017. [Online]. Available: https://engineering.salesforce.com/open-sourcing-ja3-92c9e53c3c41. [Accessed: 30-Jan-2019].
[24] R. Verhoef, “Hunting SSL/TLS clients using JA3,” 2018. [Online]. Available: https://isc.sans.edu/forums/diary/Hunting+SSLTLS+clients+using+JA3/23972/.
[25] B. Smith, “Inspecting Encrypted Network Traffic with JA3,” 2018. [Online]. Available: https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/inspecting-encrypted-network-traffic-with-ja3/. [Accessed: 30-Jan-2019].
[26] Cisco, “SSL Introduction with Sample Transaction and Packet Exchange - Cisco,” 2017. [Online]. Available: http://www.cisco.com/c/en/us/support/docs/security-vpn/secure-socket-layer-ssl/116181-technote-product-00.html. [Accessed: 05-Apr-2019].
[27] M. Heinemeyer, “Beyond the hash: How unsupervised machine learning unlocks the true power of JA3,” 2018. [Online]. Available: https://www.darktrace.com/en/blog/beyond-the-hash-how-unsupervised-machine-learning-unlocks-the-true-power-of-ja-3/. [Accessed: 30-Jan-2019].
[28] R. Luks, “Nail These 6 Encrypted Traffic Cases with Flowmon | Flowmon.” [Online]. Available: https://www.flowmon.com/en/blog/nail-these-6-encrypted-traffic-cases-with-flowmon. [Accessed: 30-Mar-2019].
[29] “TLS Fingerprinting with JA3 and JA3S – Salesforce Engineering.” [Online]. Available: https://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s-247362855967. [Accessed: 20-Mar-2019].
[30] J. Althouse, “I’m curious what you would find if you added JA3S to your JA3 Blacklist. It could increase the fidelity of the data, reducing false positives for the consumers of the data,” 2019. [Online]. Available: https://twitter.com/abuse_ch/status/1098827404469039104.
[31] A. Nisar, A. Kashaf, Z. A. Uzmi, and I. A. Qazi, “A Case for Marrying Censorship Measurements with Circumvention.”
[32] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, “Blocking-resistant communication through domain fronting,” *Proc. Priv. Enhancing Technol.*, vol. 2015, no. 2, pp. 46–64, 2015.
[33] D. Fifield and D. Fifield, “Threat modeling and circumvention of Internet censorship,” 2017.
[34] “Defend against Domain Fronting with JA3,” 2018. [Online]. Available: https://www.netscylla.com/blog/2018/08/09/defend-against-domain-fronting-with-ja3.html. [Accessed: 30-Mar-2019].
[35] “Detect and Stop Data Exfiltration | Use Cases | Security and Fraud | Splunk.” [Online]. Available: https://www.splunk.com/en_us/solutions/solution-areas/security-and-fraud/use-cases/detect-and-stop-data-exfiltration.html. [Accessed: 25-Dec-2018].
[36] “Detecting and Preventing Data Exfiltration.”

[37] J. Costa, “Blocking Data Exfiltration (Part 19 of 20: CERT Best Practices to Mitigate Insider Threats Series).” [Online]. Available: https://insights.sei.cmu.edu/insider-threat/2017/08/blocking-data-exfiltration-part-19-of-20-cert-best-practices-to-mitigate-insider-threats-series.html. [Accessed: 01-Jan-2019].
[38] “What is Network Monitoring and How Does It Work?” [Online]. Available: https://www.ouritdept.co.uk/what-is-network-monitoring/. [Accessed: 02-Mar-2019].
[39] S. M. GadAllah, “The Importance of Logging and Traffic Monitoring for Information Security,” *SANS Inst.*, p. 18, 2003.
[40] S. A. R. Shah and B. Issac, “Performance comparison of intrusion detection systems and application of machine learning to Snort system,” *Futur. Gener. Comput. Syst.*, vol. 80, pp. 157–170, 2018.
[41] A. Cecil, “A Summary of Network Traffic Monitoring and Analysis Techniques.”
[42] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, *Network Traffic Anomaly Detection and Prevention*. 2017.
[43] “Open Source IDS: Snort or Suricata?,” 2018. [Online]. Available: https://resources.infosecinstitute.com/open-source-ids-snort-suricata/. [Accessed: 02-Mar-2019].
[44] “Snort, Suricata and Bro: 3 Open Source Technologies for Securing Modern Networks | Bricata.” [Online]. Available: https://bricata.com/blog/snort-suricata-bro-ids/. [Accessed: 02-Mar-2019].
[45] J. White, T. Fitzsimmons, J. Licata, and J. Matthews, “Quantitative Analysis Of Intrusion Detection Systems: Snort and Suricata.”
[46] “Bro vs Snort or Suricata,” 2018.
[47] “Classifying Network Traffic,” 2011.
[48] “What is Suricata - Suricata - Open Information Security Foundation,” *Open Information Security Foundation*, 2015. [Online]. Available: https://redmine.openinfosecfoundation.org/projects/suricata/wiki/What_is_Suricata. [Accessed: 16-Apr-2019].
[49] “Network security monitoring using Suricata.” [Online]. Available: https://subscription.packtpub.com/book/networking_and_servers/9781789138399/10/ch10lvl1sec88/network-security-monitoring-using-suricata. [Accessed: 02-Mar-2019].
[50] D. Day and B. Burns, “A Performance Analysis of Snort and Suricata Network Intrusion Detection and Prevention Engines,” *ICDS 2011, Fifth Int. Conf. Digit. Soc.*, no. c, pp. 187–192, 2011.
[51] C. Ho and N. Chiao, “Statistical Analysis of False Positives and False Negatives from Real Traffic with Intrusion Detection / Prevention Systems,” *IEEE Commun. Mag.*, vol. 50, no. March, pp. 146–154, 2012.
[52] E. Albin, “Calhoun: The NPS Institutional Archive DSpace Repository A comparative analysis of the Snort and Suricata intrusion-detection systems.”
[53] L. Bernaille *et al.*, “Early Recognition of Encrypted Applications To cite this version : Early Recognition of Encrypted Applications,” 2014.
[54] M. Husák, M. Čermák, T. Jirsík, and P. Čeleda, “Network-based HTTPS client identification using SSL/TLS fingerprinting,” *Proc. - 10th Int. Conf. Availability, Reliab. Secur. ARES 2015*, pp. 389–396, 2015.
[55] L. Brotherston, “TLS Fingerprinting: Smarter Defending & Stealthier Attacking,” 2015. [Online]. Available: https://blog.squarelemon.com/tls-fingerprinting/. [Accessed: 07-Apr-2019].
[56] F. Střasák and S. Garcia, “Machine Learning for network HTTPS analysis

Detecting malware even when it is encrypted."
[57] J. Althouse, John; Atkinson, Jeff; Atkins, "ja3/bro/," 2018. [Online]. Available: https://github.com/salesforce/ja3/tree/master/bro.
[58] "Normal Captures — Stratosphere IPS." [Online]. Available: https://www.stratosphereips.org/datasets-normal. [Accessed: 24-Mar-2019].
[59] Robinson.S, "Decision Trees in Python with Scikit-Learn," 2018. [Online]. Available: https://stackabuse.com/decision-trees-in-python-with-scikit-learn/. [Accessed: 10-Apr-2019].
[60] M. Garbade, "Regression Versus Classification Machine Learning: What's the Difference?," 2018. [Online]. Available: https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7. [Accessed: 11-Apr-2019].
[61] Kevin Markham, "Simple guide to confusion matrix terminology," *Data School*, 2014. [Online]. Available: http://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/. [Accessed: 12-Apr-2019].
[62] "Classification Report — yellowbrick 0.9.1 documentation." [Online]. Available: https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html. [Accessed: 12-Apr-2019].
[63] Russell, "Creating and Visualizing Decision Trees with Python," 2017. [Online]. Available: https://medium.com/@rnbrown/creating-and-visualizing-decision-trees-with-python-f8e8fa394176. [Accessed: 09-May-2019].
[64] C. Strelioff, "Decision trees in python with scikit-learn and pandas — chris' sandbox," 2015. [Online]. Available: http://chrisstrelioff.ws/sandbox/2015/06/08/decision_trees_in_python_with_scikit_learn_and_pandas.html. [Accessed: 09-May-2019].
[65] "4.2. Meta Keywords — Suricata 4.1.0-dev documentation." [Online]. Available: https://suricata.readthedocs.io/en/suricata-4.1.0/rules/meta.html. [Accessed: 03-Apr-2019].
[66] "reverse.it - DriverBooster.exe," 2018. [Online]. Available: https://www.reverse.it/sample/8fbb04e836f94742fd5800677364ffc873c8308d541eb766bdf3420cc54d05d5?environmentId=100.
[67] I. Torroledo, L. D. Camacho, and A. C. Bahnsen, "Hunting Malicious TLS Certificates with Deep Neural Networks," pp. 64–73, 2018.
[68] "Sandnet :: Login." [Online]. Available: https://sandnet.abuse.ch/login/. [Accessed: 24-Mar-2019].
[69] "Index of /publicDatasets/CTU-Normal-33," 2018. [Online]. Available: https://mcfp.felk.cvut.cz/publicDatasets/CTU-Normal-33/. [Accessed: 05-May-2019].
[70] "GitHub - trisulnsm/ja3prints: JA3 TLS Fingerprint database." [Online]. Available: https://github.com/trisulnsm/ja3prints. [Accessed: 23-Mar-2019].
[71] S. Rawale, "Understanding Decision Tree, Algorithm, Drawbacks and Advantages.," 2018. [Online]. Available: https://medium.com/@sagar.rawale3/understanding-decision-tree-algorithm-drawbacks-and-advantages-4486efa6b8c3. [Accessed: 12-Apr-2019].
[72] B. Anderson and D. Mcgrew, "Machine Learning for Encrypted Malware Traaic Classification: Accounting for Noisy Labels and Non-Stationarity," *KDD*, vol. 17, 2017.