Real-time DDoS flood attack monitoring and detection (RT-AMD) model for cloud computing

Omaimah Bamasag [1] , **Alaa Alsaeedi** [Corresp., 2] , **Asmaa Munshi** [3] , **Daniyal Alghazzawi** [4] , **Suhair Alshehri** [5] , **Arwa Jamjoom** [4]

[1] Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

[2] Department of Computer Science, University of Jeddah, Jeddah, Saudi Arabia

[3] Cybersecurity Department, University of Jeddah, Jeddah, Saudi Arabia

[4] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

[5] Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

Corresponding Author: Alaa Alsaeedi
Email address: aalsaeedi0034.stu@uj.edu.sa

In recent years, the advent of cloud computing has transformed the field of computing and information technology. It has been enabling customers to rent virtual resources and take advantage of various on-demand services with the lowest costs. Despite the advantages of cloud computing, it faces several threats; an example is a Distributed Denial of Service (DDoS) attack, which is considered among the most serious ones. This paper presents real-time monitoring and detection of DDoS attacks on the cloud using a machine learning approach. Naïve Bayes, K-Nearest Neighbor, Decision Tree, and Random Forest machine learning classifiers have been selected to build a predictive model named "Real-Time DDoS flood Attack Monitoring and Detection RT-AMD." The DDoS-2020 dataset was constructed with 70,020 records to evaluate RT-AMD's accuracy. The DDoS-2020 contains three protocols for network/transport-level, which are TCP, DNS, and ICMP. This paper evaluates the proposed model by comparing its accuracy with related works. Our model has shown improvement in the results and reached real-time attack detection by using incremental learning. The model achieved 99.38% accuracy for the random forest in real-time on the cloud environment and 99.39% on local testing. The RT-AMD was evaluated on the NSL-KDD dataset as well, in which it achieved 99.30% accuracy in real-time in a cloud environment.

1 *Type: xxx*
2
3
4 # Real-Time DDoS flood Attack Monitoring and Detection (RT-AMD) Model
5 # for Cloud Computing

7 **Omaimah Bamasag,[1] Alaa Alsaeedi,[2*] Asmaa Munshi[3], Daniyal Alghazzawi[1] , Suhair Alshehri[1] ,**
8 **Arwa Jamjoom[1]**

9 [1]Center of Excellence in Smart Environment Research, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
10 [2]Computer Science Department, University of Jeddah, 21921, Saudi Arabia
11 [3]Cybersecurity Department, University of Jeddah, Jeddah, 21921, Saudi Arabia
12 [*]Corresponding Author: Alaa Alsaeedi. Email: aalsaeedi0034.stu@uj.edu.sa

13 Received: XX Month 202X; Accepted: XX Month 202X

14 **Abstract:** In recent years, the advent of cloud computing has transformed the field
15 of computing and information technology. It has been enabling customers to rent
16 virtual resources and take advantage of various on-demand services with the
17 lowest costs. Despite the advantages of cloud computing, it faces several threats;
18 an example is a Distributed Denial of Service (DDoS) attack, which is considered
19 among the most serious ones. This paper presents real-time monitoring and
20 detection of DDoS attacks on the cloud using a machine learning approach. Naïve
21 Bayes, K-Nearest Neighbor, Decision Tree, and Random Forest machine learning
22 classifiers have been selected to build a predictive model named "Real-Time
23 DDoS flood Attack Monitoring and Detection RT-AMD." The DDoS-2020
24 dataset was constructed with 70,020 records to evaluate RT-AMD's accuracy. The
25 DDoS-2020 contains three protocols for network/transport-level, which are TCP,
26 DNS, and ICMP. This paper evaluates the proposed model by comparing its
27 accuracy with related works. Our model has shown improvement in the results and
28 reached real-time attack detection by using incremental learning. The model
29 achieved 99.38% accuracy for the random forest in real-time on the cloud
30 environment and 99.39% on local testing. The RT-AMD was evaluated on the
31 NSL-KDD dataset as well, in which it achieved 99.30% accuracy in real-time in a
32 cloud environment.

33 **Keywords:** machine learning; DDoS; cloud computing; incremental learning

34 ## 1 Introduction

35 The emergence of cloud computing has gained much attention due to its various features such as cost-
36 effectiveness and on-demand service provision. Cloud computing is a shared environment (multi-tenancy)
37 between more than one user, using the same physical resources. Despite its advantages, the shared
38 environment concept may threaten the security and availability of provided services. A Cloud Services
39 Provider (CSP) must have the ability to ensure the security and availability of resources to maintain the
40 commitment to customers, called the Service Level Agreement (SLA). Cloud computing is becoming
41 popular by the day as more people and companies are attracted to employing it in their businesses. Its
42 utilization is of high benefit; however, security remains a serious problem, especially in the public cloud
43 environment.

44 This study will investigate current work in DDoS attacks targeting cloud services and propose an efficient
45 model to detect DDOS flooding attacks at the network/transport-level. This model is called the Real-Time
46 DDoS flood Attack Monitoring and Detection (RT-AMD) Model, which aims to enhance cloud services

47 security by protecting all resources in a cloud environment from DDoS attacks. It is characterized by being
48 real-time as it monitors the cloud environment and alerts any attempted attack in real-time. The
49 administrator will be notified of this incident with a timely alert message. The notification message contains
50 all the information on the attack to facilitate the administrator in dealing with it.

51 The contribution of this research is twofold: the first is to evaluate machine learning algorithms for the
52 collected dataset; the second is to improve the performance to reach real-time attack detection.

2 Background

54 Recently, cloud computing has gained much attention as it has widespread impacts across different fields
55 such as information technology, business, software engineering, and data storage. The cloud environment
56 provides resources to customers in a virtual way with high efficiency and low cost. For example, it enables
57 the users to experiment with software products before purchasing them and use storage capacity at a low
58 cost compared to buying it in traditional ways. The National Institute of Standards and Technology (NIST)
59 defines cloud computing as "a model for enabling convenient, resource pooling, ubiquitous, on-demand
60 access which can be easily delivered with different types of service provider interaction" [1].

61 A cloud environment is characterized by many features such as manageability, scalability, availability,
62 security, on-demand service, expedience, ubiquity, multitenancy, elasticity, and stability. The services
63 delivery in a cloud environment was categorized into three main models: Infrastructure as a Service (IaaS),
64 Platform as a Services (PaaS), Software as a Service (SaaS), and Everything as a service (XaaS) either in
65 public, private, community, or hybrid cloud as defined by NIST [2]. Figure 1 shows the services delivery
66 model details with examples.

67

68   **Figure 1:** Services delivery models

69 Many security challenges are faced by CSP, of which the main one is the trust that must be in place
70 between CSP and the cloud customers. Trust is how the provider can protect the customer data from any
71 breach [3]. One of the popular features of the cloud environment is multi-tenancy and virtualization. Many
72 customers share physical resources, which constitutes a considerable challenge in making such an
73 environment secure [2].

74 Shared data in the cloud can create risks of customers' data being lost or used by an unauthorized third
75 party. There are many other types of cyberattacks on cloud security allowed by system and application
76 vulnerabilities, such as account hijacking, malicious insiders, data loss, denial, and distributed denial of
77 service. These attacks have a substantial negative impact on confidentiality, integrity, and availability of
78 data.

79 The availability of cloud services is one of the most critical CSP goals. Unavailability adversely affects
80 CSP and cloud customers. DoS and DDoS attacks are the main threats leading to a cloud service's
81 unavailability. DoS is a cyberattack where an attacker aims to make the systems and servers unavailable,
82 preventing customers from accessing the servers and resources [4]. DoS attacks launched in a distributed
83 manner to speed up the consumption of the resources for one or many targets are called Distributed Denial
84 of Service attacks (DDoS) [4]. DDoS Attack types are explained in the following subsection.

2.1 DDoS Attack Types

86 DDoS attacks are classified based on targeted protocols such as the network/transport or application level.

87 • Network/Transport level DDoS attacks: These attacks occur mostly when using network and transport
88   layer protocols such as TCP, UDP, and ICMP. These attacks are further categorized into three types:

89   1. Volume attacks: The attacker aims to consume all the resources of the target servers and make them
90   unavailable by sending many packets (bandwidth/flooding attack) such as TCP flood, ICMP flood,
91   etc.

2. Protocol attacks: In this type, the attack consumes all resources and intermediate connection media as a firewall by exploiting protocol vulnerabilities and bugs as TCP SYN flood, TCP SYN-ACK flood, etc.

3. Reflection and amplification attacks: Attempts to consume the victim's resources by sending fake request messages (such as ping requests) through spoofing the victim's IP address to the reflectors. The reflectors send a high volume of response messages to the victim's IP address such as Smurf attacks.

- Application-level DDoS attacks: These attacks aim to consume services' resources or cause starvation of resources to disrupt customers through establishing requests, overloading the application servers. The most popular type of attack at this level is HTTP flooding attacks. Many studies have classified DDOS attack at the application level based on the following categories [5]:

1. Session flooding attack: Servers' resources are disabled from being launched when session request rates are high. These requests usually are higher than those generated by valid users.

2. Request flooding attack: Sends sessions that contain more requests than the valid users.

3. Asymmetric attack: Wastes resources such as CPU and memory of the server by sending sessions with high-workload requests.

4. Slow request/response attack: Uses all server resources by sending incomplete requests slowly to keep the servers in the waiting state to receive data.

2.2 Intrusion Detection System

An intrusion detection system (IDS) is a device or software tool that identifies unusual events by monitoring the network traffic to distinguish the normal from abnormal behaviors [6]. IDS is classified into three main categories based on the analysis method. The choice between methods depends on several factors, such as the anomaly type, applied environment, security level required, and the cost [6].

The IDS methods classifications are Signature-based, Anomaly-based, and Hybrid detection [7]. Signature-based detection, also called Knowledge-based or Rule-based detections, is suitable for detecting known attacks by comparing captured behavior. Anomaly-based detection, also known as Behavior-based, is useful to detect unknown attacks. These techniques compare the observed behavior with normal behavior to detect abnormal events. Hybrid-based detection works by combining the detection techniques mentioned above. The performance of this detection depends on the types of techniques chosen. Table 1 shows the advantage and limitations of these detection methods [7].

Table 1: Summary of IDS techniques

3 Literature Review

This section presents and critically analyzes existing research studies to detect the attacks in the three categories of IDS methods mentioned above.

3.1 Signature-based detection

Tanrıverdi et al. [8], Bakshi et al. [9], and Modi et al. [10] proposed a signature-based detection method. Tanrıverdi et al. [8] presented the detection of web attacks using a blockchain-based attack detection model. The signatures listed in this study are automatically updated by blockchain technology. An additional advantage of this proposed method is that it can be used against zero-day attacks. Bakshi et al. [9] presented a method to distinguish between the normal and abnormal traffic in VMs. It uses Snort to analyze the collected traffic to determine the attack. The virtual server then drops packets coming from the specified IP address. Modi et al. [10] presented a method to detect known attacks and derivatives of known attacks. It also uses a Snort tool to detect the known attacks from network traffic. The detected attack is input to a signature DB to predict derivatives of the attack by using signature as a priority.

3.2 Anomaly-based detection

Hong et al. [11] and Kemp et al. [12] presented an anomaly-based solution to detect slow HTTP attacks, a type of DDoS attack. Hong et al. [11] developed a software-defined networking (SDN) controller; however, Kemp et al. [12] deployed the proposed model using machine learning techniques. It selected eight classification algorithms for predictive models: Random Forest, decision trees, K-Nearest Neighbor, Multilayer Perceptron, RIPPER (JRip), Support Vector Machines, and Naïve Bayes. The authors used the Weka machine learning toolkit to build these models. ANOVA was used to compare the values of slow attack detection among the eight models. They evaluated the models by Area Under the receiver operating characteristic curve (AUC), Receiver Operating Characteristic (ROC) curve graphs, True Positive Rate (TPR), and False Positive Rate (FPR).

Singh et al. [13], Filho et al. [14], Wang et al. [15], and Sreeram et al. [16] presented anomaly-based solutions to detect HTTP attacks. Singh et al. [13] used a Multilayer Perceptron with a Genetic Algorithm (MLP-GA)-based method for detecting DDoS attacks on incoming traffic. Authors in [13] identified four features to detect application-layer attacks; first is the number of HTTP counts, referring to the count number of requests per IP address. It is assumed that any single IP address that sends more than 15–20 HTTP GET/POST requests is an attack. Second is the number of IP addresses, referring to the number of IP addresses in small windows time. It is assumed the attacks have more than 20 IPs in windows time. The third was the constant mapping function; the attacker's ports are different from legitimate users' as the one used by the attacker is varied and remains open. The fourth is fixed frame length; codes with fixed frame length are considered as an attack.

In comparison, Filho et al. [14] proposed an online smart detection system for DoS/DDoS attack detection. The detection approach used the Random Forest Tree algorithm to classify various types of DoS/DDoS attacks such as flood TCP, flood UDP, flood HTTP, and slow HTTP. However, Wang et al. [15] proposed a detection scheme for HTTP-flooding (HTTP-Soldier) based on web browsing clicks. HTTP-soldier used the large-deviation principle of webpage popularity to be able to distinguish between normal and abnormal traffic. The large-deviation probability-based detection may affect some normal users. The authors mentioned that their proposed scheme could not detect a single Uniform Resource Locator (URL) attack. The false positive of a Multi-URL attack with the most popular webpages is 12.2%, but the false positive of Multi-URL attack with the least popular webpages is at 17.1%. This solution can achieve high performance in Multi-URL attacks with the most popular web pages. Sreeram et al. [16] used bio-inspired machine learning metrics to detect HTTP flood attacks to achieve fast and early detection. Authors in [16] adopted the Bat algorithm, which has low process complexity, as a bio-inspired approach.

Choi et al. [17], Aborujilah and Musa [18], and Sahi et al. [19] presented a cloud-based flood attack detection method. Choi et al. [17] proposed a method to integrate the detection of DDoS flood attacks and MapReduce processing in a cloud computing environment. The proposed framework consists of three parts: first is the packet and log collection module (PLCM), which analyses packet transmission and web server logs in the first part. Second is the pattern analysis module (PAM), which produces the pattern for DDoS attack detection. Finally is the detection module (DM), which detects DDoS attacks by comparing them with a normal behavior model. However, Aborujilah and Musa [18] presented the detection based on the covariance matrix approach. The proposed detection was divided into training and testing phases. A training phase aimed to construct a normal network traffic profile. The testing phase was to detect any abnormal traffic by the deviation between the normal and any other network traffic. The normal traffic is captured from end-users browsing the Internet in their cloud, whereas the flooding attack traffic is generated using the PageRebooter tool. It was evaluated by using the confusion matrix and present results for an internal and external cloud environment, while Sahi et al. [19] proposed a detection model for TCP flood DDoS attacks. This model selected different classifiers, the least squares support vector machine (LS-SVM), Naïve Bayes, K-nearest, and Multilayer Perceptron.

Lin et al. [20], Li et al. [21], and Nawir et al. [22] presented an anomaly-based detection method for detecting DDoS attacks. The proposal of Lin et al. [20] and Li et al. [21] used deep learning techniques.

185 Lin et al. [20] used Long Short-Term Memory (LSTM) to build the neural network model, which is a
186 specific recurrent neural network structure (RNN). Li et al. [21] used LSTM and Gated recurrent units
187 (GRU) recurrent neural networks. It used BGP and NSL-KDD datasets. The best accuracy achieved was
188 using the BGP dataset in the range of 90–95%. However, the proposed of Nawir et al. [22] used machine
189 learning algorithms. The authors in [Reff9+] selected five machine learning algorithms to include Naïve
190 Bayes (NB), Averaged One Dependence Estimator (AODE), Radial Basis Function Network (RBFN),
191 Multi-Layer Perceptron (MLP), and J48 trees. A comparison was drawn between these algorithms based
192 on accuracy and processing time. A UNSW-NB15 dataset was selected in this experiment.

193 Haider et al.[23] proposed a deep Convolutional Neural Network (CNN) framework for efficient DDoS
194 attack detection in SDN. This proposed framework has been evaluated using hybrid state-of-the-art
195 algorithms on CICIDS2017 dataset. Hwang et al.[24] proposed an unsupervised deep learning model for
196 early network traffic anomaly detection, namely D-PACK based on CNN. The experimental results show
197 low false-positive rate and high accuracy. Novaes et al.[25] proposed using short-term memory and fuzzy
198 logic for DDoS attack detection and mitigation in SDN. The proposed system consists of three phases:
199 Characterization, anomaly detection, and mitigation. The evaluation of this system has been conducted
200 using CICDDoS2019 dataset with archived accuracy of 96.22%.

201

202 *3.3 Hybrid-based detection*

203 Hatef et al. [26], Zekri et al. [27], and Saleh et al. [25] proposed a hybrid-based detection system. While
204 Hatef et al. [26] and Zekri et al. [27] deployed cloud environment approaches. Hatef et al. [26] is a hybrid
205 intrusion detection approach in cloud computing (HIDCC). The applied detection was a combination of
206 signature-based and anomaly-based detection techniques. The Snort tool is used for known attacks
207 (signature-based detection) by employing the Apriori algorithm to generate a pattern from derived attacks.
208 Both clustering and classification algorithms are applied for the undetectable attack through Snort. The
209 clustering module receives and determines the input packet based on the sample vector. Then the classifier
210 module determines the final class of the packet through algorithm C4.5 as a decision tree classifier
211 according to the found cluster. However, Zekri et al. [27] proposed using machine learning for anomaly
212 detection and Snort technique for signature detection. Three algorithms were selected: Decision Tree, Naïve
213 Bayes, and K-Means algorithms. The decision tree achieved the best accuracy. The proposed Saleh et al.
214 [28] was applied in real time and deployed in three stages. First, the naïve Base feature selection (NBFS)
215 technique was employed to reduce the dimensionality of sample data. Second, optimized Support Vector
216 Machines (OSVM) were used to reject the noisy input sample as it might have caused misclassification.
217 Finally, the attacks were detected by Prioritized K-Nearest Neighbors (PKNN) classifier. This proposed
218 scheme takes time in the first and second stages at feature selection and outlier rejection before attack
219 detection.

220 This study will explore the use of data mining techniques (classification) in real-time detection. We will
221 focus on the network/transport level as it is the core layer of network architecture. There are few existing
222 studies on volume-based network/transport-level DDoS attack detection in the cloud environment.
223 Moreover, there are very few studies that have proposed online detection with a high detection rate. This
224 study will employ different classification algorithms (machine learning) that suit our needs to build
225 detection models. We will then evaluate these models by comparing them against two main factors:
226 efficiency of detection and detection rate.

227 **4 Proposed Framework**

228 This section explains the RT-AMD model framework by presenting its components and the employed data
229 mining classification methods.

230 *4.1 Main components*

231 The proposed RT-AMD model consists of two main components: monitoring and detection.

232   1- The monitoring component is responsible for monitoring the network traffic for requests coming
233   to the webserver on the cloud and extracting the traffic from the network log. If the traffic from the
234   log is found in the Blacklist, an alert with corresponding information is sent to the cloud admin. If
235   it is not in the Blacklist, it will move into the next component, detection.

236   2- The detection component uses trained classifiers to detect if the incoming traffic behavior is normal
237   or abnormal. When abnormal behavior occurs, it will alert the system, send information to the
238   admin, then update the Blacklist with the new traffic information. Otherwise, it can access the cloud
239   and benefit from its services. An overview of the proposed environment is shown in Figure 2.

240   At the beginning, the admin needs to log in/register to benefit from the RT-AMD tool. Once the
241 registration is done, RT-AMD will send a verification code for the entered email to ensure that the email
242 address is correct. The tool will then start monitoring the network traffic to detect any malicious behavior.
243 Once an attack occurs, the RT-AMD will detect it, then send all information assigned to this traffic to the
244 admin email. This makes it easier for the admin to act on any malicious behavior. A flowchart of the
245 proposed detection framework is shown in Figure 3.

246

247   **Figure 2:** Overview of RT-AMD framework.

248

249

250   **Figure 3:** Flowchart of RT-AMD framework

251 *4.2 Data Mining Classification Methods*

252   Data mining analyzes large volumes of data to identify and predict any threats; this helps to solve
253 problems and reduce risks. Data mining can answer questions that have typically been time-consuming to
254 address manually by using several statistical techniques to analyze data in various ways.

255   Recently, high arrival rates of online data streams have imposed high resource requirements on data
256 mining processing systems. DataStream Mining (also known as stream learning) is a technique of extracting
257 knowledge structures from an unbounded and ordered sequence of data that exists over time (stream data)
258 [30]. Some differences between stream data mining and traditional data mining are shown in the following
259 [31]:

260   1- Machine learning of streaming scenarios cannot retrieve all of the data of the dataset in advance.
261   Data chunks are available in a stream, one by one, or bundle by block.

262   2- Data arriving over time in streams may be unlimited in their number, resulting in difficulty storing
263   all arrival data in the memory.

264   3- Data from streams need to be analyzed quickly to provide real-time response and prevent data
265   waiting.

266   4- Some incoming stream data may lack accurate class labels because of the label query's high cost
267   for each data stream.

268   Building knowledge from stream data mining is called incremental learning [31]. Incremental learning
269 has received much attention from both academia and industry. It is a machine learning approach where
270 knowledge is applied as new instances arrive, and what has been learned is updated according to the new
271 instances [32].

272   There are several techniques in data mining, such as regression, clustering association, and classification.
273 Different techniques serve different purposes. However, most data mining techniques usually applied for
274 this area are classification techniques. Classification is a type of supervised learning that predicts the class
275 label to which data belongs. The classifier works by obtaining a training dataset containing several attributes
276 and class labels. The classifier then tests the dataset to evaluate the model.

277 The proposed framework will employ the classifiers in the detection component of the framework with
278 two class labels. The attributes assigned to normal behavior are labelled "normal" and those to abnormal
279 behavior "anomaly." Some classifiers were found to have better detection results than others based on the
280 recommendations in the related works [12, 14, 28]. We selected the following: Naïve Bayes, Decision
281 Trees, K-Nearest Neighbor, and Random Forest. These classifiers have been selected to build the predictive
282 models.

283 **5 Dataset Collection**

284 A DDoS-2020 dataset is a network/transport-level dataset that authors have assembled and that contains
285 of two types of traffic: attack and normal, with 70,020 records of traffic. Attack traffic consists of 27,169
286 instances, and the normal traffic consists of 42,851 instances. We have collected attack traffic from the
287 CAIDA DDoS Attack 2007 dataset and collected normal traffic by capturing packets using Wireshark. The
288 Center for Applied Internet Data Analysis Dataset "DDoS Attack 2007" [33] contains an approximately
289 one-hour collection of anonymized (abnormal) traffic from a DDoS attack on August 4, 2007 [33].
290 Wireshark is one of the most popular open-source network analyzer tools under the GNU General Public
291 License (GPL) [34]. Wireshark captures packets using the "pcap" library and different network media types,
292 including Ethernet, Wi-Fi, Bluetooth, and others [35].

293 The DDoS-2020 dataset contains information corresponding to each packet: source IP address, destination
294 IP address, protocol type (ICMP, TCP, and DNS), packet length, packet timestamp, and label to determine
295 whether the traffic is normal ("0") or attack ("1"). We have 29,554 instances of TCP, 20,727 instances of
296 ICMP, and 19,739 instances of DNS with 0% missing value. Figure 4 shows the distribution of TCP, ICMP,
297 and DNS protocols in the dataset.

298

299   **Figure 4:** The distribution of protocols in the dataset

300 The dataset contains two types of traffic: attack traffic consists of 27,169 instances and normal traffic
301 consists of 42,851 instances. The label for normal traffic is = 0 and the label for attack traffic is = 1. The
302 protocol's distribution ratio on normal traffic and the dataset's attack traffic are shown below in Figure 5.

303

304   **Figure 5:** Protocol's distribution ratio on normal and attack traffic

305 The timestamp range in the dataset is between 89.981–11045.6567 seconds and 0% missing value. Each
306 timestamp has a range of instances for each type of traffic (attack, normal). The timestamp distribution ratio
307 on normal traffic and attack traffic in the dataset is shown below in Figure 6.

308

309   **Figure 6:** Distribution of timestamp over labels

310 The range of length is between 46–1800 with 1,190 distinct and 0% missing value. Each length ranges
311 across a group of instances for each type of traffic (attack, normal). The length distribution ratio on normal
312 traffic and the attack traffic in the dataset is shown below in Figure 7.

313

314   **Figure 7:** Distribution of length over labels

315 During the collection and cleaning of the DDoS-2020 dataset, we were determined to distribute the
316 elements' ratio in a balanced manner. The balance is considered in the distribution of traffic between
317 instances (normal and attack) and the distribution of individual protocol that has range of instances for
318 attack and normal traffic. Figure 8 shows the distribution ratio of attacks and normal traffic for each TCP,
319 ICMP, and DNS protocol.

320

321   **Figure 8:** Traffic distribution of protocols.
322   Figure 9 shows the distribution ratio of attacks and normal traffic for each timestamp.
323
324   **Figure 9:** Traffic distribution of timestamp.
325   Figure 10 shows the distribution ratio of attacks and normal traffic for each length range.
326
327   **Figure 10:** Traffic distribution of length.
328   The timestamp and the length of traffic is also notable; the distribution ratio of the timestamp and length
329   for each protocol should be semi-balanced. Figures 11 and 12 show the distribution ratio of timestamp and
330   length over protocols sequentially.
331
332   **Figure 11:** Distribution of timestamp over protocols
333
334   **Figure 12:** Distribution of length over protocols

335   **6 Evaluation Results**

336   RT-AMD model is implemented using the Python programing language, SQLite, and GCP. Python is a
337   powerful language; it contains many libraries for machine learning. One of the libraries that is well-known
338   for real-time learning purposes is Scikit multi-flow. The evaluation of this model was in the GCP
339   environment.
340   As mentioned above, the RT-AMD tool was evaluated by the selected machine learning algorithms. Naïve
341   Bayes, Decision tree, K-neighbors, and Random forest were selected for this evaluation. We experimented
342   with the RT-AMD tool in three situations: offline localhost, online localhost, and online remote virtual host
343   created by GCP. Configuration of localhost was with Microsoft Windows 10 Pro operating system, 24.0
344   GB RAM, and Intel(R) Core(TM) i7-7500 processor, and the remote virtual host was configured with e2-
345   medium machine type, 2 vCPUs, and 4 GB memory.

346   The evaluation measured accuracy and performance. The random forest achieved the best accuracy in
347   incremental learning either on localhost or remote virtual host at around 99.38%. However, K-neighbors
348   achieved the best accuracy in offline learning. Table 2 and Figure 13 show the details for each experiment.
349   The Naïve Bayes achieved the efficient execution-time for online learning: 12.08s for cloud testing and
350   22.32s for local testing. Table 3 shows the details of execution time for local online testing and online cloud
351   testing.
352   **Table 2:** RT-AMD accuracy for each experiment
353
354
355   **Figure 13:** RT-AMD accuracy for each experiment
356   **Table 3:** Execution time details
357
358   The proposed tool is tested with different datasets; our DDoS-2020 dataset and NSL-KDD dataset [36].
359   The NSL-KDD dataset contains 125,964 samples, among which 67,343 are normal and 58,621 attacks. The
360   NSL-KDD is a useful dataset and popularly used in previous studies [23, 28]. It is a new version of the
361   KDD'99 dataset. Table 4 and Figure 14 show the details of accuracy for each dataset in cloud testing.
362
363   **Table 4:** Accuracy and execution time for each dataset in cloud testing

364

365

366

367

368

369

370

371   **Figure 14:** Accuracy and execution time for each dataset in cloud testing

372   **7 Discussion**

373   As we mentioned above, the experimental RT-AMD tool was used in three different situations; offline
374   localhost, online localhost, and online remote virtual host created by GCP. We achieved 99.38% accuracy
375   with real-time detection in a cloud environment for the random forest. The accuracy for online detection is
376   much higher than offline detection; this is because of Scikit-multi-flow library incremental learning
377   characteristics. The execution time of random forest on the cloud was worst for many reasons, such as the
378   virtual machine's abilities and the way random forest algorithms work.

379   Our model achieved the best accuracy with a real-time response on the cloud environment in comparison
380   with related work. This is due to the model work and the machine learning algorithm's efficiency using the
381   Scikit-multi-flow library. We have seen above the same model results offline using a Scikit-learn and online
382   using the Scikit-multi-flow library. The latter library features incremental learning that gradually improves
383   the algorithms' performance with runtime, thus improving results. Table 5 and Figure 15 show the details
384   of this comparison.

385   **Table 5:** Comparison of the result with related work

386

387

388   **Figure 15:** Comparison of the result with related work

389   **8 Conclusion and future work**

390   This study discusses the issues surrounding DDoS attacks on cloud environments, presenting the main
391   types of DDoS attacks and the challenges and risks faced. Further, it reviews some of the previous
392   techniques detecting DDoS attacks. Machine learning is one of the most common techniques used to detect
393   DDoS attacks. Furthermore, incremental learning is one of the best strategies to learn and classify in real-
394   time. This study's main contributions are to evaluate machine learning algorithms for the dataset collected
395   and investigate the results with related works. Furthermore, we improve outcomes and reach real-time
396   attack detection by using incremental learning.

397   The RT-AMD model is proposed to detect DDoS attacks on the cloud environment using machine
398   learning techniques. Four machine learning algorithms were selected to evaluate this model: Naïve Bayes,
399   Decision Tree, K-neighbors, and Random Forest. The RT-AMD model was developed by python, SQLite
400   databases, and GCP to detect and alert of the DDoS attacks and test on the cloud environment platform. It
401   was evaluated by using two datasets, the DDoS-2020 and NSL-KDD dataset. The DDoS-2020 dataset has
402   been collected with two ranges of traffic (attack and normal), with three distinct network/transport

403 protocols: TCP, ICMP, and DNS. The attack traffic were obtained from CAIDA DDoS Attack 2007 and
404 the normal traffic were obtained by using Wireshark.

405 As a result, the RT-AMD model achieved high accuracy in DDoS-2020 dataset testing and NSL-KDD
406 dataset testing. The random forest algorithm obtained the best accuracy, reaching 99.38% with the DDoS-
407 2020 dataset and 99.30% with the NSL-KDD dataset. This model achieved real-time detection without the
408 negative effect on accuracy by using an incremental learning strategy, and without needing pre-training
409 machine learning.

410 There are various ways to extend the study presented in this research. These include extending dataset
411 samples to include different types of DDoS, and evaluating and testing this model on other cloud
412 computing-related environments such as Mobile Cloud Computing (MCC), a combination between cloud
413 computing and mobile computing.

414 **Funding Statement:** The authors extend their appreciation to the Deputyship for Research & Innovation,
415 Ministry of Education in Saudi Arabia for funding this research work through the project number (IFRPC-
416 114-612-2020) at King Abdulaziz University, DSR, jeddah, Saudi arabia.

References

418 [1] Zissis, D., & Lekkas, D., "Addressing cloud computing security issues," *Future Generation Computer*
419 *Systems*, 28(3), 583–592, 2012.

420 [2] Singh, S., Jeong, Y. S., & Park, J. H., "A survey on cloud computing security: Issues, threats, and
421 solutions," *Journal of Network and Computer Applications*, 75, 200–222, 2016.

422 [3] Ghaffari, F., Gharaee, H., & Arabsorkhi, A., "Cloud Security Issues Based on People, Process and Technology
423 Model: A Survey," 2019.

424 [4] Douligeris, C., & Mitrokotsa, A., "DDoS attacks and defense mechanisms: classification and state-of-the-
425 art," *Computer Networks*, 44(5), 643–666, 2004.

426 [5] Jaafar, G. A., Abdullah, S. M., & Ismail, S., "Review of recent detection methods for HTTP DDoS
427 attack," *Journal of Computer Networks and Communications*, 2019.

428 [6] Kaur, P., Kumar, M., & Bhandari, A., "A review of detection approaches for distributed denial of service attacks,"
429 *Systems Science & Control Engineering*, 5(1), 301–320, 2017.

430 [7] Alzahrani, S., & Hong, L., "A survey of cloud computing detection techniques against DDoS attacks," *Journal*
431 *of Information Security*, 9(01), 45, 2017.

432 [8] Tanrıverdi, M., & Tekerek, A., "Implementation of blockchain based distributed web attack detection
433 application," In *2019 1st International Informatics and Software Engineering Conference* (UBMYK) (pp. 1–6),
434 2019.

435 [9] Bakshi, A., & Dujodwala, Y. B. "Securing cloud from ddos attacks using intrusion detection system in virtual
436 machine," In *2010 Second International Conference on Communication Software and Networks* (pp. 260–264),
437 2010.

438 [10] Modi, C., Patel, D., Borisaniya, B., Patel, A., & Rajarajan, M., "A survey on security issues and solutions at
439 different layers of cloud computing," *The Journal of Supercomputing*, 63(2), 561–592, 2013.

440 [11] Hong, K., Kim, Y., Choi, H., & Park, J, "SDN-assisted slow HTTP DDoS attack defense method," *IEEE*
441 *Communications Letters*, 22(4), 688–691, 2017.

442 [12] Kemp, C., Calvert, C., & Khoshgoftaar, T., "Utilizing netflow data to detect slow read attacks," In *2018 IEEE*
443 *International Conference on Information Reuse and Integration (IRI)* (pp. 108–116), 2018.

444 [13] Singh, S., Jeong, Y. S., & Park, J. H, "A survey on cloud computing security: Issues, threats, and solutions,"
445 *Journal of Network and Computer Applications*, 75, 200–222, 2016.

446 [14] Lima Filho, F. S. D., Silveira, F. A., de Medeiros Brito Junior, A., Vargas-Solar, G., & Silveira, L. F., "Smart
447 detection: An online approach for DoS/DDoS attack detection using machine learning," *Security and*
448 *Communication Networks*, 2019.

449 [15] Wang, J., Yang, X., Zhang, M., Long, K., & Xu, J., "HTTP-SoLDiER: An HTTP-flooding attack detection
450 scheme with the large deviation principle," *Science China Information Sciences*, 57(10), 1–15, 2014.

451   [16] Sreeram, I., & Vuppala, V. P. K., "HTTP flood attack detection in application layer using machine learning
452   metrics and bio inspired bat algorithm," *Applied Computing and Informatics*, 15(1), 59–66, 2019.

453   [17] Choi, J., Choi, C., Ko, B., & Kim, P., "A method of DDoS attack detection using HTTP packet pattern and rule
454   engine in cloud computing environment," *Soft Computing*, 18(9), 1697–1703, 2014.

455   [18] Aborujilah, A., & Musa, S., "Cloud-based DDoS HTTP attack detection using a covariance matrix
456   approach," *Journal of Computer Networks and Communications*, 2017.

457   [19] Sahi, A., Lai, D., Li, Y., & Diykh, M., "An efficient DDoS TCP flood attack detection and prevention system in
458   a cloud environment," *IEEE Access*, 5, 6036–6048, 2017.

459   [20] Lin, P., Ye, K., & Xu, C. Z., "Dynamic network anomaly detection system by using deep learning techniques,"
460   In *International Conference on Cloud Computing* (pp. 161–176). Springer, 2019.

461   [21] Li, Z., Rios, A. L. G., Xu, G., & Trajković, L., "Machine learning techniques for classifying network anomalies
462   and intrusions," In *2019 IEEE international symposium on circuits and systems (ISCAS)* (pp. 1–5), 2019.

463   [22] Nawir, M., Amir, A., Yaakob, N., & Lynn, O. B., "Effective and efficient network anomaly detection system
464   using machine learning algorithm," *Bulletin of Electrical Engineering and Informatics*, 8(1), 46–51, 2019.

465   [23] Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R., & Iqbal, J. (2020). A deep
466   CNN ensemble framework for efficient DDoS attack detection in software defined networks. *Ieee Access*, 8,
467   53972-53983.

468   [24] Hwang, R. H., Peng, M. C., Huang, C. W., Lin, P. C., & Nguyen, V. L. (2020). An unsupervised deep learning
469   model for early network traffic anomaly detection. *IEEE Access*, 8, 30387-30399.

470   [25] Novaes, M. P., Carvalho, L. F., Lloret, J., & Proença, M. L. (2020). Long short-term memory and fuzzy logic
471   for anomaly detection and mitigation in software-defined network environment. *IEEE Access*, 8, 83765-83781.

472   [26] Hatef, M. A., Shaker, V., Jabbarpour, M. R., Jung, J., & Zarrabi, H., "HIDCC: A hybrid intrusion detection
473   approach in cloud computing," *Concurrency and Computation: Practice and Experience*, 30(3), e4171, 2018.

474   [27] Zekri, M., El Kafhali, S., Aboutabit, N., & Saadi, Y., "DDoS attack detection using machine learning techniques
475   in cloud computing environments," In *2017 3rd International Conference of Cloud Computing Technologies and
476   Applications (CloudTech)* (pp. 1–7). 2017.

477   [28] Saleh, A. I., Talaat, F. M., & Labib, L. M., "A hybrid intrusion detection system (HIDS) based on prioritized k-
478   nearest neighbors and optimized SVM classifiers," *Artificial Intelligence Review*, 51(3), 403–443, 2019.

479   [29] Wang, B., Zheng, Y., Lou, W., & Hou, Y. T., "DDoS attack protection in the era of cloud computing and
480   software-defined networking," *Computer Networks*, 81, 308–319, 2015.

481   [30] Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., ... & Abdessalem, T., "Adaptive
482   random forests for evolving data stream classification," *Machine Learning*, 106(9), 1469–1495, 2017.

483   [31] Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., & Herrera, F., "A survey on data preprocessing
484   for data stream mining: Current status and future directions," *Neurocomputing*, 239, 39–57, 2017.

485   [32] Xie, X., & Lam, K. M. "An efficient illumination normalization method for face recognition," *Pattern
486   Recognition Letters*, 27(6), 609–617, 2006.

487   [33] CAIDA: Center for Applied Internet Data Analysis. (n.d.). Data collection, curation and sharing. Retrieved from
488   https://www.caida.org/data/.

489   [34] Munz, G., & Carle, G, "Distributed network analysis using TOPAS and wireshark," In *NOMS Workshops 2008-
490   IEEE Network Operations and Management Symposium Workshops* (pp. 161–164). IEEE, 2008.

491   [35] Wireshark.   Introduction.   (n.d.).   Retrieved   from
492   https://www.wireshark.org/docs/wsug_html_chunked/ChapterIntroduction.html.

493   [36] University of New Brunswick. NSL-KDD. (n.d.). Retrieved from https://www.unb.ca/cic/datasets/nsl.html.

Table 1: Summary of IDS techniques

Summary of IDS techniques

Methods	Signature-based	Anomaly-based	Hybrid-based
Advantage	✓ Easy to implement in real-time ✓ Low cost	✓ Effective against unknown attacks ✓ Effective against new attacks without database update	✓ High accuracy rating
Limitation	✓ Ineffective against zero-day attacks ✓ Must update the database for new attacks	✓ Difficult to implement at run-time ✓ Detection accuracy affected by the number of collected features in dataset	✓ Cost is high

Table 3: Execution time details

Execution time details

Algorithms	Local online testing execution time	Online cloud testing execution time
Naïve bayes	22.32s	12.08s
Decision tree	27.28s	14.81s
K-neighbors	297.94s	260.84s
Random forest	567.64s	741.84s

Table 4: Accuracy and execution time for each dataset in cloud testing

Accuracy and execution time for each dataset in cloud testing

Algorithms	DDoS-2020_Dataset accuracy	NSL-KDD_Dataset accuracy
Naïve Bayes	81.86%	89.42%
Decision Tree	97.89%	97.25%
K-neighbors	98.49%	95.23%
Random Forest	99.38%	99.30%

Figure 1

Figure 1: Services delivery models

Cloud Services delivery models
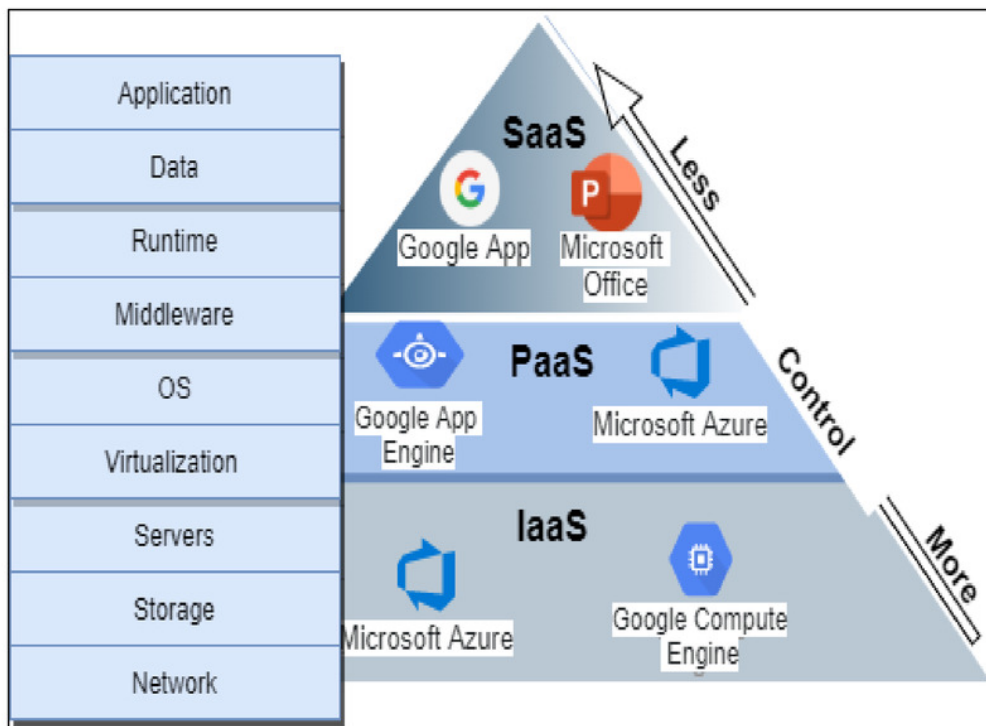
Table 2: RT-AMD accuracy for each experiment

RT-AMD accuracy for each experiment

Algorithms	Local offline testing accuracy	Local online testing accuracy	Online cloud testing accuracy
Naïve bayes	77.20 %	81.88%	81.86%
Decision tree	92.99%	97.90%	97.89%
K-neighbors	93.87%	98.48%	98.49%
Random forest	92.37%	99.39%	99.38%

Figure 2

Figure 2: Overview of RT-AMD framework

Overview of RT-AMD framework
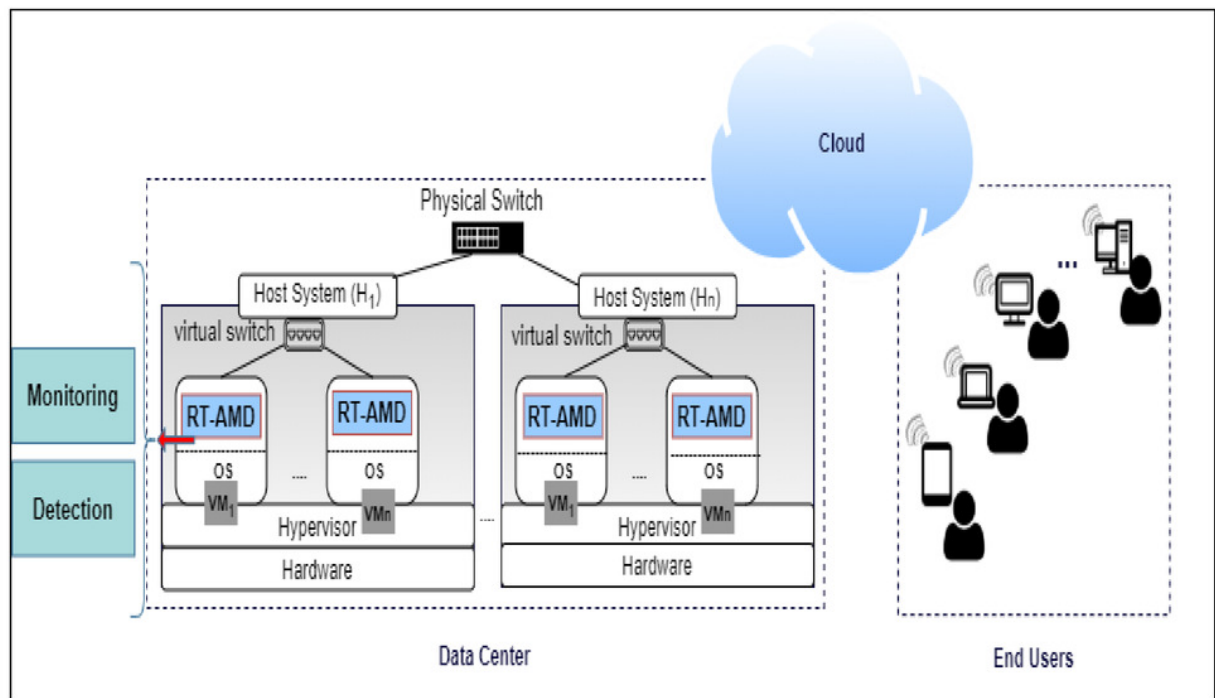
Table 5: Comparison of the result with related work

Comparison of the result with related work

Related work	Detection Method	Dataset	Accuracy (%)	Cloud-based	Online
[13]	Anomaly-based detection using Random Forest Tree algorithm	Customized dataset	96.5%	×	✓
[14]	Anomaly-based detection using Genetic Algorithm	CAIDA2007	98.04%	×	×
[15]	Anomaly-based detection based on web browsing clicks	Used the weblog of CDU website (www.cdu.edu.cn) as a simulation dataset and replayed the dataset with NS-34.	94.9%	×	×
[16]	Anomaly-based detection using bio-Inspired	CAIDA 2007	94.8%	×	×
[19]	Anomaly-based detection using four different classifiers LS-SVM, Naïve Bayes, K-nearest, and Multilayer Perceptron	N/A	97%	✓	×
[20]	Anomaly-based detection used deep learning techniques LSTM	CSE-CIC-IDS2018	96.2%	×	×
[21]	Anomaly-based detection used deep learning techniques LSTM and GRU	BGP	95.21%	×	×
[22]	Anomaly-based detection using machine learning algorithms including Naïve Bayes, Averaged One Dependence Estimator, Radial Basis Function Network, Multi-Layer Perceptron, and J48 trees	UNSW-NB15	97.26%	×	×
[24]	Hybrid-based detection using machine learning for anomaly detection and Snort technique for signature detection	N/A	98.8%	✓	×
[25]	Hybrid-based detection using K-Nearest Neighbors classifier for detection	NSL-KDD	95.77%	✓	✓

RT-AMD (our proposed)	Anomaly-based detection using machine learning algorithms including Naïve Bayes, Decision Tree, K-neighbors, and Random Forest	DDoS-2020	99.38%	✓	✓

Figure 3

Figure 3: Flowchart of RT-AMD framework

Flowchart of RT-AMD framework
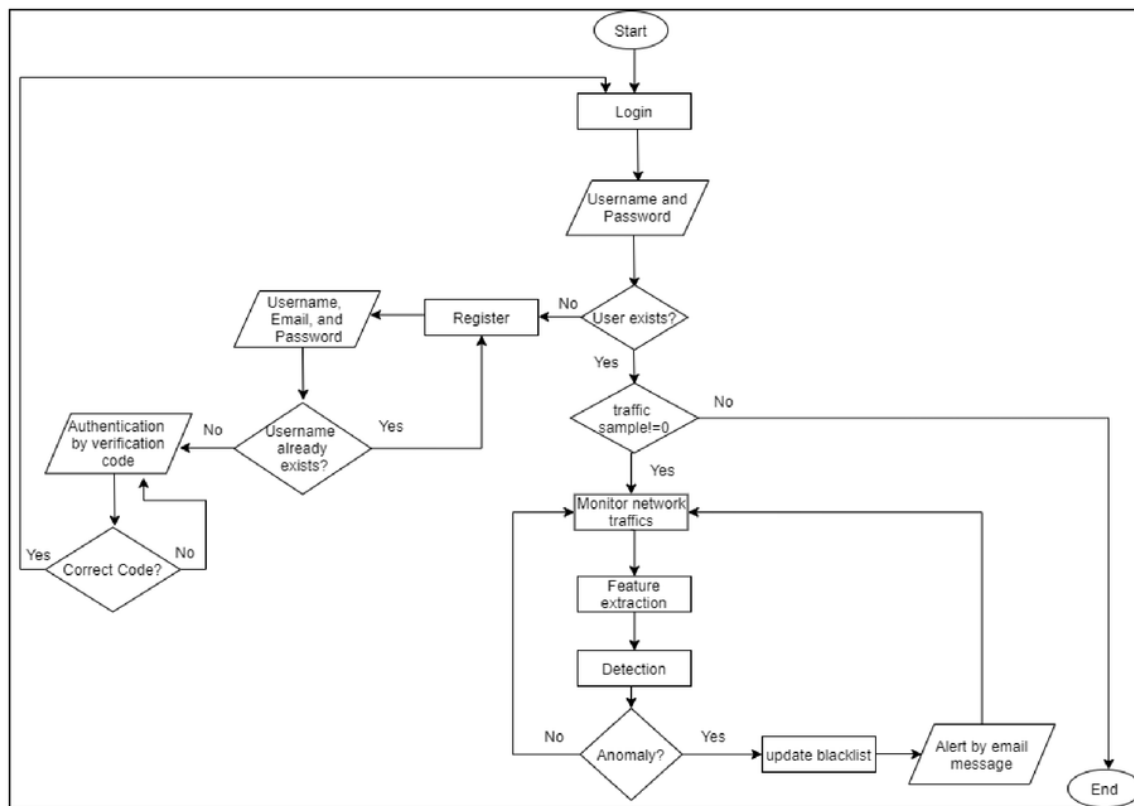
Figure 4

Figure 4: The distribution of protocols in the dataset

The distribution of protocols in the dataset
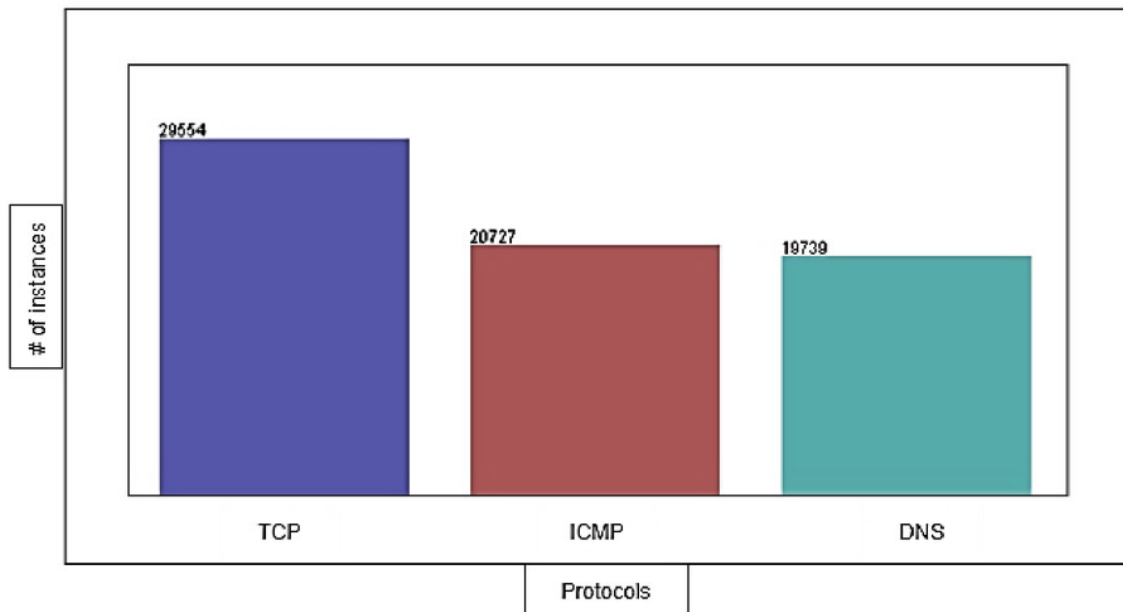
Figure 5

Figure 6: Distribution of timestamp over labels

Distribution of timestamp over labels
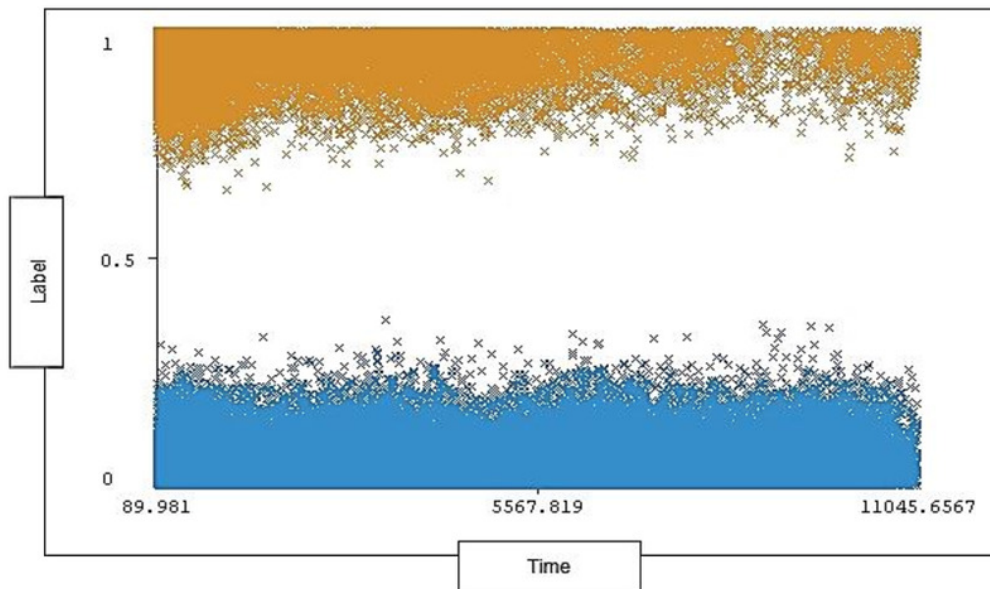
Figure 6

Figure 5: Protocol's distribution ratio on normal and attack traffic

Protocol's distribution ratio on normal and attack traffic
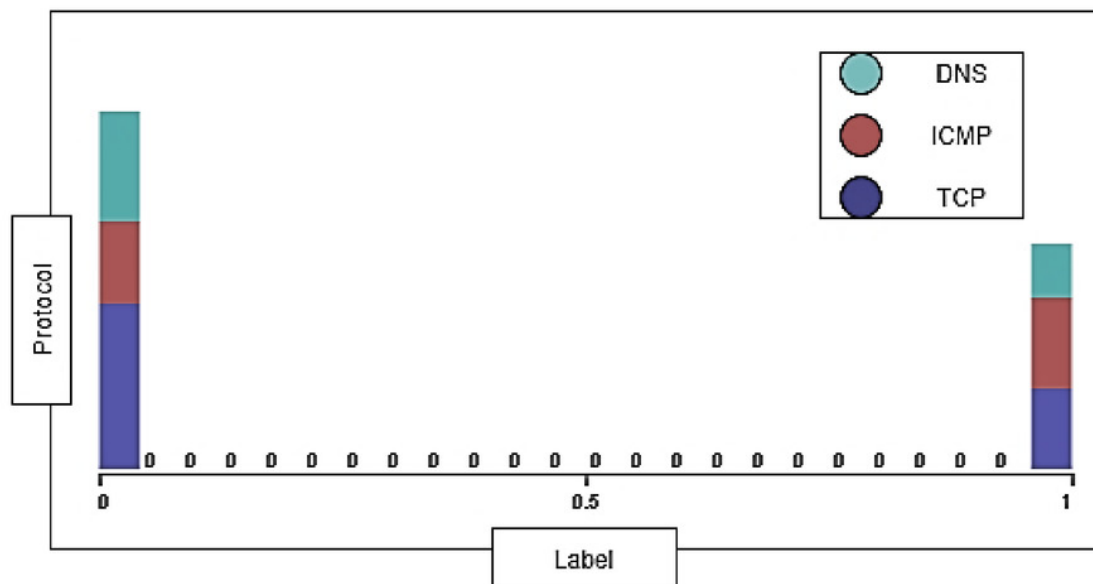
Figure 7

Figure 7: Distribution of length over labels

Distribution of length over labels
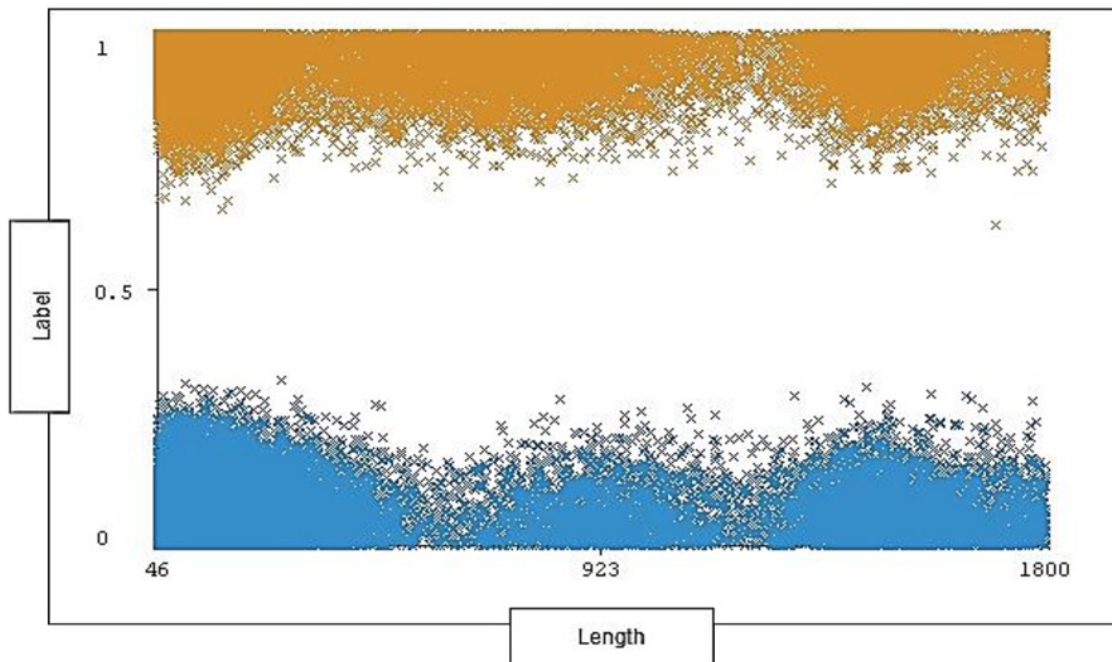
Figure 8

Figure 8: Traffic distribution of protocols.

Traffic distribution of protocols.
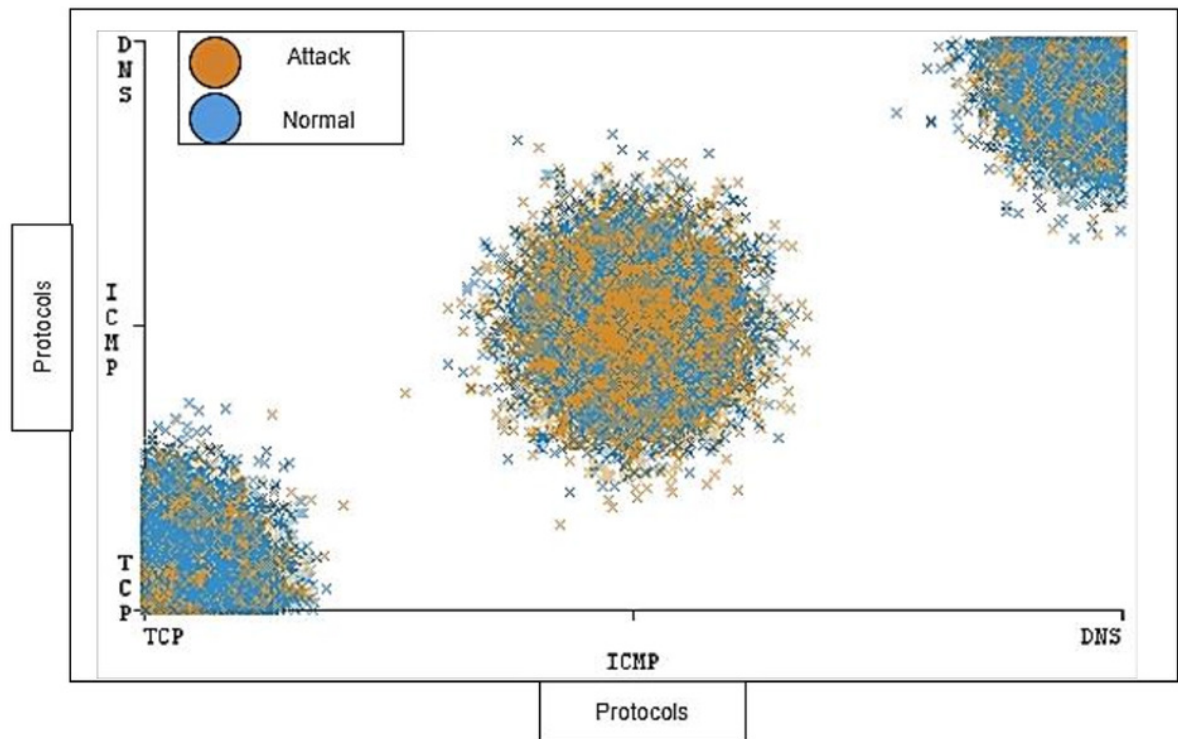
Figure 9

Figure 9: Traffic distribution of timestamp.

Traffic distribution of timestamp.
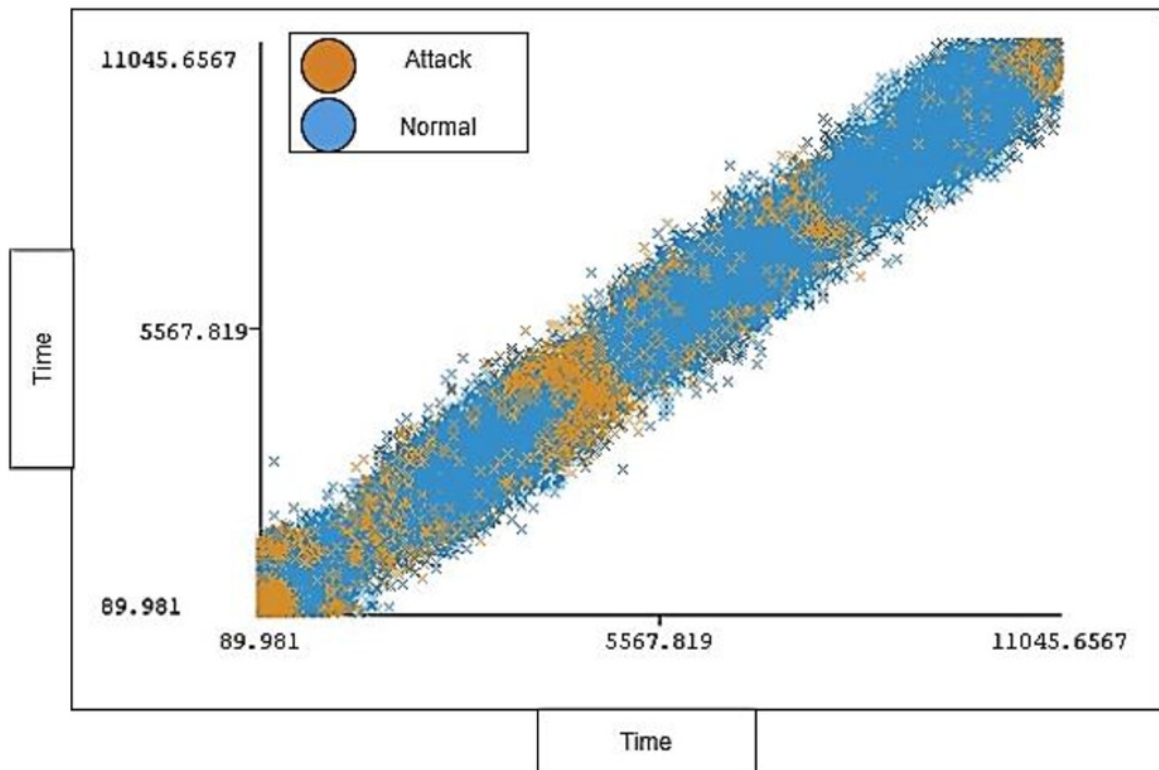
Figure 10

Figure 10: Traffic distribution of length.

Traffic distribution of length.
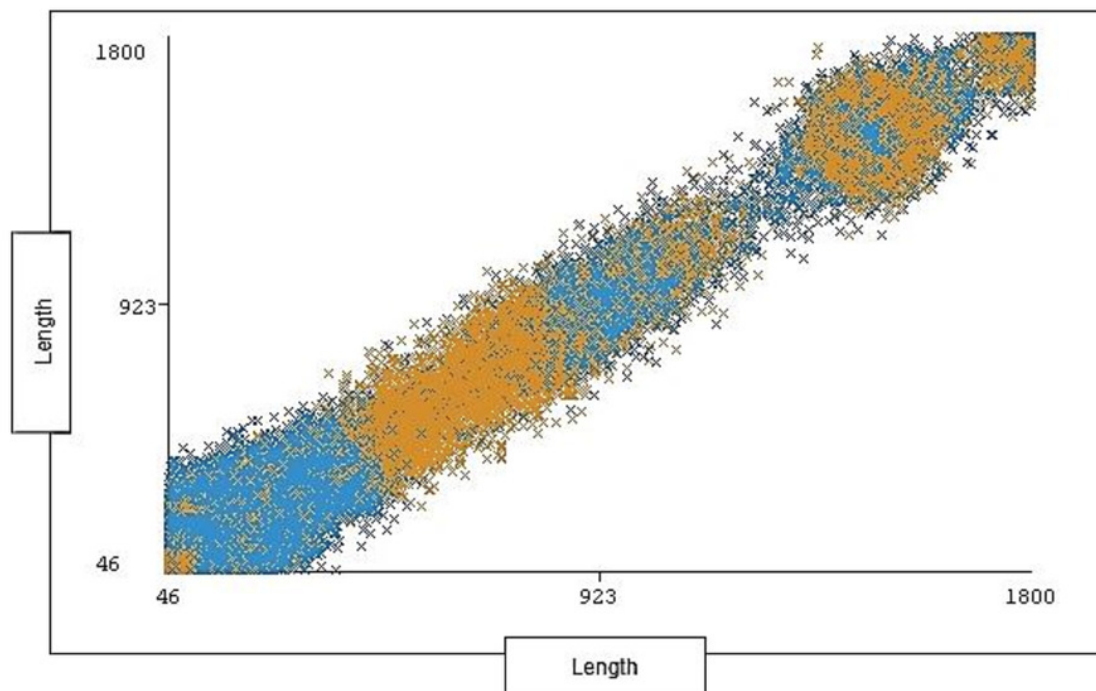
Figure 11

Figure 11: Distribution of timestamp over protocols

Distribution of timestamp over protocols
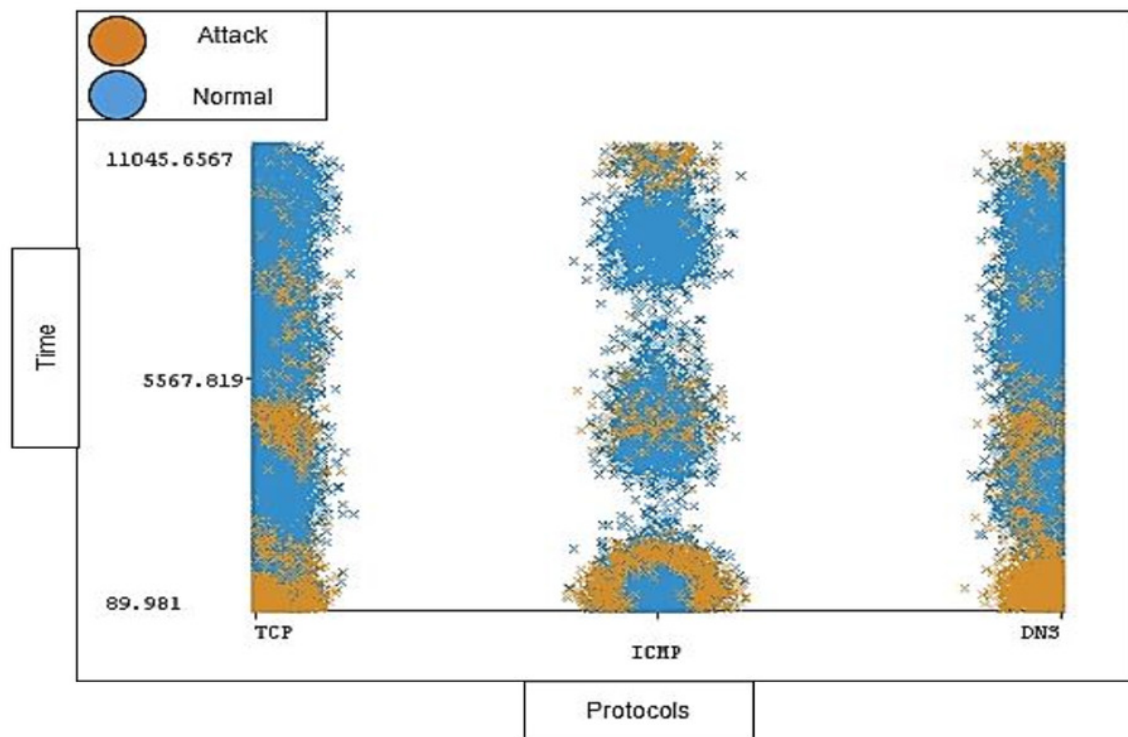
Figure 12

Figure 12: Distribution of length over protocols

Distribution of length over protocols
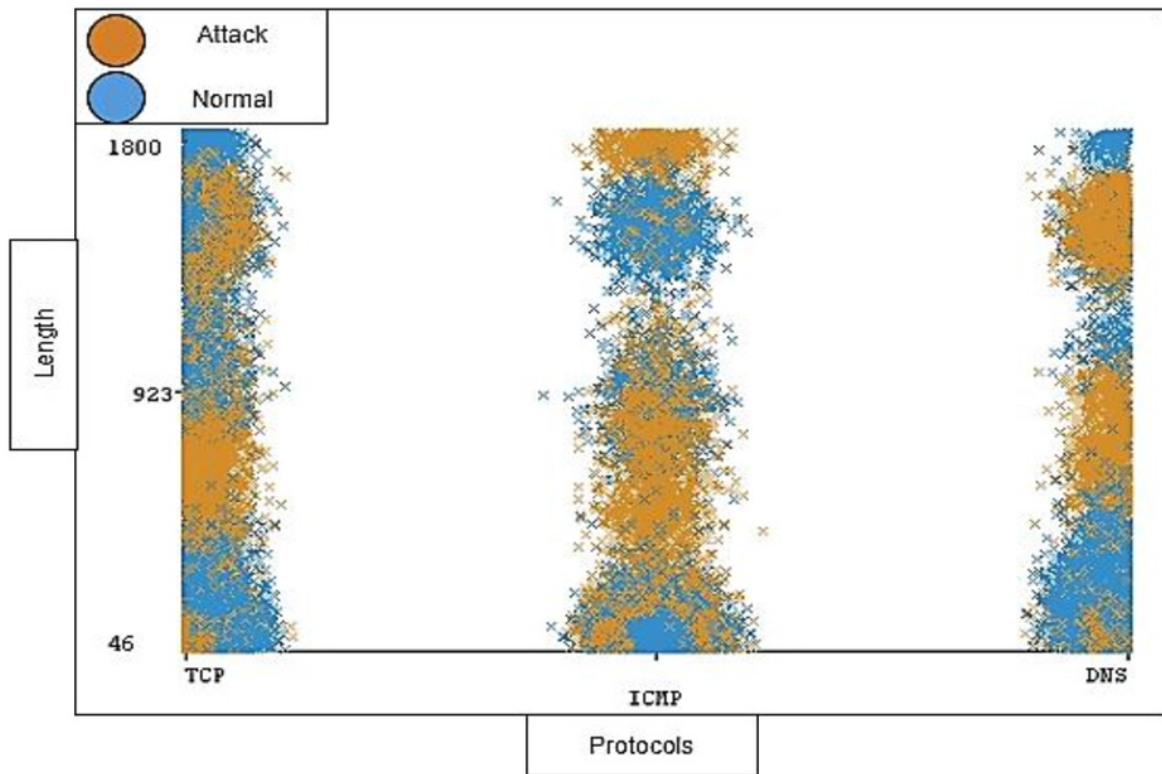
Figure 13

Figure 13: RT-AMD accuracy for each experiment

RT-AMD accuracy for each experiment
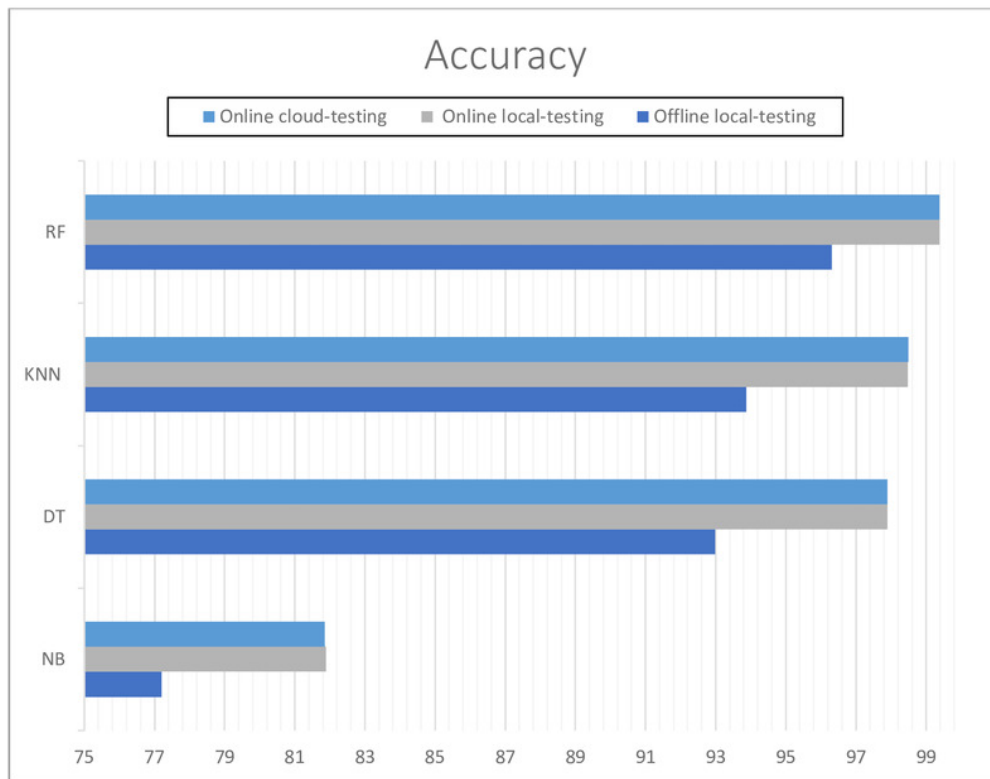
Figure 14

Figure 14: Accuracy and execution time for each dataset in cloud testing

Accuracy and execution time for each dataset in cloud testing
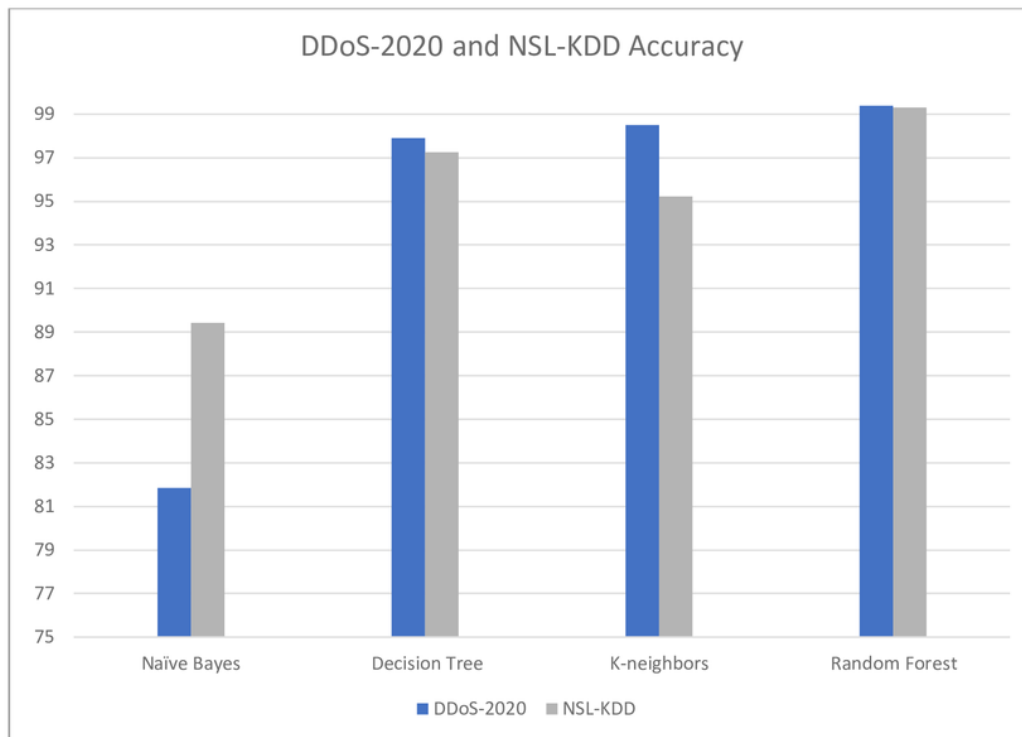
Figure 15

Figure 15: Comparison of the result with related work

Comparison of the result with related work