Intelligent Edge: an Instantaneous Detection of IoT Traffic Load

Maha Alaslani and Basem Shihada
Computer, Electrical, and Mathematical Sciences & Engineering Division
KAUST, Saudi Arabia, {maha.aslani, basem.shihada}@kaust.edu.sa

Abstract—Internet of Things (IoT) can be defined as the interconnection of any device to the Internet that collects and exchanges information. With the rapidly growing heterogeneous IoT applications and its associated devices, massive amount of data is being transmitted in the network. Often, a large spike in network traffic to a particular destination causes a widespread disruption of the Internet services for the end users, which can cause online businesses billions of dollars of losses. In this paper, we analyze an intelligent edge that can identify volumetric traffic and address them in real-time using an instantaneous detection method for IoT applications. This approach can easily detect a large surge and a potential variation in traffic patterns for an IoT application, which can contribute to safer and more efficient operation of the overall system. As per our results, we gave a closer insight on the advantage of having an intelligent edge to serve as a detection mechanism.

Index Terms—IoT, Traffic Load, Queueing Theory, QoS, Arrival Rate, Queueing Delay, Publish/Subscribe Model

I. INTRODUCTION

IoT connect different types of online devices ranging from simple sensors to smart devices such as coffee machines, smart phones, webcam, health monitoring and more. The statistics from Gartner, Inc. [1] says that there will be more than 26 billion IoT devices and will generate more than 44 zettabytes of data by 2020. This massive amount of data defined by the term of big data. Big data has been characterized by three dimensions: volume, variety, and velocity [1]. Volume describes the amount of data, variety describes the different types of data, while velocity describes the speed of data generation. However, such a huge data need to be accurately controlled to satisfy specified Quality of Service (QoS) constraints. Traditional traffic management techniques with the legacy client-server model are unsuitable for ensuring high performance delivery of IoT applications and services where dummy IoT devices will not react appropriately to high network load as human-centric devices. In many cases, prioritizing or limiting the number of redundant data play an important role in mitigating high network load. Thus, a main challenge is to distinguish devices from each other which can be achieved by the unique relationship between Publish/Subscribe model and IoT. Publish/Subscribe model helps dealing with the various traffic issues and providing full control over the data acquisition and filtering process. Fundamentally, the network should dynamically regulating the bandwidth and QoS for the large-scale, cross-domain IoT real-time applications by designing intelligent, transparent, and lightweight detection mechanism for high IoT traffic load. This can be done through suitable network policies and implement specific functions (e.g., traffic detectors) through light-weight middlebox functions that are implemented at the network edges. Using delegation, IoT devices will be able to leverage middleboxes to support different QoS functions (e.g., traffic prioritization) while providing transparency to other end-points.

Detecting, preventing, and mitigating high network load, specifically on IoT traffic, is a challenging issue for accommodating the diverse QoS requirements. Therefore, it is important to understand the unique characteristics of IoT traffic. In [2], Sivanathan et al. classify IoT traffic patterns in terms of traffic rate, burstiness, idle durations in smart cities environments. They propose a method to identify specific type of an IoT device and differentiate IoT from non-IoT traffic to improve network quality and support appropriate security levels. In addition, there are several different approaches for addressing high network loads issues. One solution is to develop load balancing routing techniques such as Adaptive Routing using Multi-technologY and Load balancer (ARMY-Lancer) [3] which introduced by Kotagi et al. to prevent network congestion and overloading of gateways for maximizing throughput of IoT networks. This routing method selects the next hop gateway dynamically based on the expected load. Thus, data packets will be uniformly distributed in an IoT network and will improve the overall throughput. Another solution, proposed by Ray and Kwang in [4], introduces a simple distributed congestion control method that allocates data rates to M2M flows in proportion to their throughput demands. While, Huang et al. propose an improved random early discard (RED) algorithm in [5] which uses the instantaneous queue length to calculate packets drop rate, when high network load appears, and react accordingly for improving IoT service performance.

The works mentioned above will help minimizing the negative impacts of high network load, but will not prevent volumetric IoT traffic from accessing the production network. As a consequence, we want to alleviate volumetric traffic intelligently at the edges as it is well known that the tremendous amount of IoT traffic is a major concern for the stability and safety of the network services. IoT devices generate a packet stream that often cannot be easily distinguished from each other. Thus, selection of a reaction mechanism is a trade-off between the effective prevention of flooding traffic
and the damage to useful information. In this paper, we propose an instantaneous detection method for IoT (IDIoT) and examine IoT communication protocol where IoT devices communicate in a Publish/Subscribe architecture. We employ Message Queuing Telemetry Transport (MQTT) communication protocol as our Publish/Subscribe model. Furthermore, we will use M/G/1 queuing model to characterize the MQTT central point and propose a detection technique that helps preventing flooding the network near the edges.

The rest of the paper is organized as follows. The system model is presented in Section II. Then, we will present the mathematical analysis for our proposed model in Section III. In section IV simulation results will demonstrate the validation of the analytic model. Finally, we will conclude the work in Section V.

II. SYSTEM MODEL

Several IoT protocols have been designed to fit low-cost, low-power, and resource constrained devices. Message Queuing Telemetry Transport (MQTT) [6] provides standard communication foundations for the IoT implemented as Publish/Subscribe architecture. This protocol is used for machine-to-machine communication where a network bandwidth is scarce and small code footprint is needed. Publish/Subscribe is a well-established communications paradigm that allows the publisher to push their messages to any number of subscribers through a central communication point (i.e., MQTT broker). Each message that is published to the broker will be associated with a specific topic, and every MQTT client must have a unique client ID. The broker uses the topic as a routing information where each client that wants to receive messages subscribes to a certain topic, and the broker will be responsible for distributing all messages that belong to the matching topic. MQTT is a topic-centric communication protocol where the clients communicate over the topics without dependencies between the data publishers and the subscribers. The Publish/Subscribe model is shown in Figure 1. Generally, IoT applications can be categorized into two groups, they are, event-based and participatory applications. Event-based applications where IoT devices are triggered to collect information by specific event such as disaster monitoring, surveillance security, traffic management, smart parking system, and supermarkets/retail stores. The framework for other applications are based on participatory and periodically sensing. For example, health care monitoring, smart grids, weather signal, water and air quality. Thus, in this study, we adopt these two kinds of IoT applications and their associated traffic patterns. It is worth noting that each traffic type requires different processing time.

Our system considers an MQTT application scenario, shown in Figure 1, in which M IoT-based sensors collect certain information, and then, transmit them to the adjacent switches all the way to the broker. We assume that the network switches are smart enough to be programmed remotely and act according to the traffic changes. This assumption can hold with the new emerging technologies such as Software-Defined-Networking (SDN). The broker will diffuse the interest data to the subscribe applications (Ai, i = 1, 2, ..., M) and the traffic management information to the switches on the path to block or allow MQTT clients to alleviate high network load. Two types of IoT applications are deployed with deterministic (participatory-based applications) and random (event-based applications) packets arrivals. Thus, sensing information i.e., IoT data packet arrival is modeled by both fixed and exponential inter-arrival times with rates γm, m = 1, 2, n, n + 1, ..., M.

Deterministic application at the publisher’s side has an inter-arrival rate of

\[ \lambda_D = \sum_{i=1}^{n} \gamma_i, \quad (1) \]

while random packet arrivals from event-based publishers are assumed to be Poisson distributed with rate

\[ \lambda_R = \sum_{j=n+1}^{M} \gamma_j \quad (2) \]

We employ two priority queues with exponentially distributed packet arrival rate and arbitrary distribution for service rate. Hence, the intermediate node and the relay switches are modeled as single server facilities with M/G/1 queue that responsible for forwarding the IoT data to the subscribed clients. Due to the distributed nature of the IoT devices, the broker is the bottleneck point in our network. We aim to limit the number of packets that arrive to the central broker to prevent overwhelming the network. The broker operates in non-preemptive M/G/1 priority queues with exponentially distributed inter-arrival time λ and mean service time of Ï€. Random packets (i.e., event-based IoT traffic) are considered to have the higher priority.

![Fig. 1: An Overview of the System.](image)

Packets arrival rates (traffic velocity) and total number of publishers (traffic volume) are two important factors to reduce network load. Our model experimentally maintains and analyzes a number of previous packet arrival rate at the IoT broker to identify the confidence in which an increased arrival rate (λ) is in the application defined range or not. In this case, the rate of the packets must not be greater than the maximum arrival rate \( \lambda_{max} \). On the other hand, the maximum allowable publishers \( M_{max} \) should be determined to meet the
QoS requirements. Thus, the probability of blocking an IoT device to prevent high network load is given by,

\[ P_{\text{blocking}} = \Pr \{ \lambda > \lambda_{\text{max}} \mid M > M_{\text{max}} \} \quad (3) \]

The ultimate objective of our work is to analyze the prevention method for high traffic load in IoT network (IDIoT) while minimizing the negative effect on the system users. Our main contributions are as follow:

1) Compute the maximum packets arrival rate that fully represents the intrinsic characteristics of a publisher node, and
2) Compute the maximum sampling interval for calculating the current packets rate such that the total queuing delay must meet a predefined QoS threshold.
3) Compute the maximum number of publishers that can be served by the system without exceeding some delay constraints.

To address the above concerns in IoT network, we analyze the above mentioned queuing model for an IoT-MQTT network and evaluate a detection mechanism based on the collected traffic arrival rate and the number of users before a high network load takes place.

### III. Analysis

In this section, we present our analytical expressions for the major QoS performance measurements that can be affected by a high network load in terms of the system response time, the queuing delay, and the total number of packets arrived to the central broker [7]. Table I shows a summary of notations used throughout this paper.

**TABLE I: Notations**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td>Number of MQTT publishers (IoT devices)</td>
</tr>
<tr>
<td>( M_{\text{max}} )</td>
<td>Maximum number of MQTT publishers (IoT devices) that can be served by the system</td>
</tr>
<tr>
<td>( T_{\text{QoS}} )</td>
<td>Predefined queuing delay threshold for QoS constraints</td>
</tr>
<tr>
<td>( \overline{Q} )</td>
<td>Total Queuing delay</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>Packet arrival rate</td>
</tr>
<tr>
<td>( \lambda_{\text{max}} )</td>
<td>Maximum historical packet arrival rate</td>
</tr>
<tr>
<td>( \overline{X} )</td>
<td>Mean service time at the broker</td>
</tr>
<tr>
<td>( \overline{X}^2 )</td>
<td>Second moment service time at the broker</td>
</tr>
<tr>
<td>( \overline{R} )</td>
<td>Mean residual time</td>
</tr>
<tr>
<td>( \rho )</td>
<td>The utilization</td>
</tr>
<tr>
<td>( \overline{W} )</td>
<td>Average queue waiting time for packets</td>
</tr>
<tr>
<td>( \overline{T} )</td>
<td>Average response time</td>
</tr>
<tr>
<td>( \overline{N} )</td>
<td>Average number of packets</td>
</tr>
<tr>
<td>( I )</td>
<td>Sampling interval to calculate the current maximum arrival rate</td>
</tr>
</tbody>
</table>

In our system, we are interested in analyzing two types of data packets, random (R) and deterministic (D) packets with different priority classes. For each priority class \( p \in \{ R, D \} \), the packets arrive according to a Poisson process with \( \lambda = \lambda_D + \lambda_R \), service time is of mean \( \overline{X} \), and the second moment is \( \overline{X}^2 \).

To achieve our goal, we first note that the average waiting time of high priority packets (i.e., random packets), denoted as \( \overline{W}_R \), is given by,

\[ \overline{W}_R = \frac{\overline{R}}{(1 - \rho_R)} \quad (4) \]

where \( \rho_R = \lambda_R \cdot \overline{X}_R \) is the fraction of time a broker is serving high priority traffic, and \( \overline{R} \) is the residual time. This residual time can then be written as,

\[ \overline{R} = \frac{1}{2} (\rho_R + \rho_D) \cdot \frac{\overline{X}^2}{\overline{X}} \quad (5) \]

The average waiting time expressions, denoted as \( \overline{W}_D \), of the lower class priority packets (i.e., deterministic packets) can be expressed as,

\[ \overline{W}_D = \frac{\overline{R}}{(1 - \rho_R)(1 - \rho_R - \rho_D)} \quad (6) \]

where the load \( \rho_D \) is defined as \( \rho_D = \lambda_D \cdot \overline{X}_D \). The total time packets from class \( p \in \{ R, D \} \) spend in the system is,

\[ T = \overline{W} + \overline{X} \quad (7) \]

The total number of packets for each class \( p \in \{ R, D \} \) is given by,

\[ \overline{N} = \lambda \cdot \overline{W} \quad (8) \]

By collecting and experimentally analyzing sufficiently large number of historical arrival rate \( \lambda_{\text{max}} \) for each type of traffic at the MQTT broker, we will achieve a clean separation between IoT devices. The proposed model enhances the responsiveness of the IoT broker upon any packets flooding. When any superfluous requests from single or multiple MQTT publishers are identified, the broker will react accordingly to address such volumetric traffic and stop the IoT devices with abnormal behavior. From (8) and by applying Little’s law, the arrival rates for each class \( p \in \{ R, D \} \) can be calculated as,

\[ \lambda = \frac{\overline{N}}{\overline{W}} \quad (9) \]

In large packet counts caused by a legitimate burst, a maximum value of consecutive \( \lambda \) in short time interval is expected to be close to \( \lambda_{\text{max}} \) since the high utilization of network resources remains only for a short period of time. A larger value of \( \lambda \) can be considered that event is more likely due to non-legitimate traffic load.

To maintain a predefined QoS, the total delay experienced by the detection method must not exceed a predefined threshold \( T_{\text{QoS}} \). Therefore, the cumulative delay can be directly linked with the sampling interval, denoted by \( I \), to collect and calculate the current maximum \( \lambda \). Thus, if we let \( \overline{Q} \) be a
random variable that stands for total queuing delay, then the maximum allowable sampling interval that respects the QoS delay constraint is given by,

\[ I = \arg \max_I \{ \bar{Q} \leq T_{QoS} \}, \]  

(10)

where,

\[ \bar{Q} = W_R + W_D, \]  

(11)

with \( W_R \) and \( W_D \) are given by (4) and (6) for event-based and participatory-based clients, respectively.

Moreover, the target system can be flooded with requests from large amount of distributed IoT publishers. Thus, there is a hard limit on the number of publishers that each system can serve. This limit can be defined by the total delay constraints \( T_{QoS} \) imposed by the application requirements. Then, the maximum allowable number of publishers the system can serve is,

\[ M_{max} \approx \frac{T_{QoS}}{Q}, \]  

(12)

where \( M_{max} \) is the maximum number of IoT publishers for both types (participatory-based and event-based) applications. By delegating the above traffic management information (\( \lambda_{max} \) and \( M_{max} \)) to the network edges, we will be able to avoid high network load and control IoT volumetric traffic.

IV. PERFORMANCE EVALUATION

In order to verify our proposed model, a simulation is conducted using MATLAB/Simulink, where the network topology presented in Figure 1. The MQTT core node is simulated as single M/G/1 queue with generally distributed service time of 2000 packets/second. The publisher nodes are attached to the MQTT broker, and as mentioned above, two categories of published applications are considered with constant and Poisson data traffic. We simulate weather signal application of three cities where each city has 1000 IoT devices that generate data at a constant rate of one packet every 15 milliseconds. We also simulated four car parking lots with 1000 IoT sensors produce Poisson traffic at a mean arrival rate of one packet every 64 milliseconds. As described above, two factors played a major role in avoiding high network load, publisher’s maximum arrival rate \( \lambda_{max} \) and total number of publisher nodes \( M_{max} \).

The simulation results related to the maximum arrival rate \( \lambda_{max} \) are obtained. For this reason, our simulation runs for relatively long period of time to be able to maintain a number of historical arrival rates. From Figure 2, we observe that smart parking traffic rates range between 0 and 9 packets/millisecond, while the weather signal application has a maximum of 8 packets/millisecond. MQTT broker maintains a list of the maximum allowed arrival rate for each client and communicates the traffic policies (maximum arrival rate) to the programmable entry-point switches. The switches continuously collect the statistics and update its decisions based on network conditions and the broker needs. At this point, if the arrival rate of a specific MQTT client (IoT device) within a sampling interval of 100 milliseconds is higher than the maximum arrival rate with 98% confidence, then that client will be blocked to eliminate network load.

The reaction of the system with and without our IDIoT method has been simulated. We ran the simulation for 100 seconds. At the 50th second, one city and two parking lots start sending at higher rates of one packet every 1.5 milliseconds and one packet every 6.4 milliseconds, respectfully. This kind of changes in the data rates can be done intentionally or not. Our system identified the overrated devices and only react to them while maintaining the system behavior stable to other normal users. As per Figure 3, MQTT server’s average utilization is 0.13 in normal condition, but it escalates to 0.39 under the network load. What can be inferred from this is that, under high network load situation, the server is extremely burdened with high volume requests hence the increase in utilization. At some point, the server eventually reaches a very high utilization and is not being able to accept any request from the system clients. As opposed, when our technique

Fig. 2: Packet Arrival Rate Distribution.

Fig. 3: The MQTT Broker Utilization.
is considered, the MQTT broker can identify clients with abnormal behavior precisely by their client ID and maintain the system at a stable state in order to respond to other clients. That gives a clear indication to the benefits of incorporating the power of the MQTT-IoT protocol and the IDIoT method.

The network load merely delays or prevent data packets from being processed for legitimate connections; such additional delays degrade the QoS of traffic ongoing connections. IDIoT should be robust and efficient to improve QoS performance for traffic when subjected to such threat. To validate this, we further examine the correctness of the analytic expressions for $T$ (i.e., response time in single M/G/1 priority queues). As shown from Figure 4, the proposed technique reduced the system response time and prevented QoS degradation performance by blocking the overrated nodes. Parking lots application in Figure 4a has an average response time of 0.91 milliseconds under high network load without employing our technique and an average of 0.64 milliseconds with the IDIoT method. Weather signal application in Figure 4b has almost the same reaction with an average of 0.65 milliseconds and 0.93 milliseconds with and without IDIoT, respectively.

The number of blocked packets is another important measure and a definite decision should be made at the right time to prevent flooding the system with huge amount of useless data packets. By considering the total number of packets $N$ for each client, we can obviously show the advantage of our method. Under a normal situation in Figure 5a, we can see that each parking lot participates by around 1556 packets and when the data rate increases the misbehaved parking lots send out around 8513 packets. In our proposed method and before the load takes place, the first and second parking lot send out...
only 648 packets to the broker, then the broker detected an increase in the arrival rates and prevented them from accessing the system. Figure 5b shows the same situation happened with the weather signal application. IDIoT prevents around 35000 unuseful packets from the third city which can be responsible for breakdown the overall system quality.

The sampling interval is playing an important role to maintain the predefined QoS constraints. When the traffic volume increased, our system is still resilience against extreme case (new arrival rate $\lambda_{new} = 5\lambda$) and identifying the overrated devices precisely by their client ID with some extra queuing delay cost. This queuing delay can be minimized by optimizing the sampling interval used to calculate the current arrival rate, as shown in Figure 6. With sampling interval of 100 milliseconds, the queuing delay was around 1 millisecond which can be further reduced to 0.2 millisecond by reducing the sampling interval to 20 millisecond. Optimizing the sampling interval should add another dimension to circumstantially reduce the performance overhead. We leave the sampling interval optimization as a future work.

The total number of IoT publishers must be controlled to maintain system stability and prevent IoT network load at the early stages. Any system can be able to serve a specified number of users. When, the number of IoT devices increased above the adequate limit, a degradation in service quality will appears. Thus, we conduct a simple experiment to measure the queuing delay as the number of IoT devices (for both applications) grew by a factor of (2x, 4x, 6x, 8x, 10x and 12x). As shown in Figure 7, the queuing delay was about 0.42, 0.5, and 0.68 millisecond for 2x, 4x, and 6x, respectfully. The delay continues to increase by an order of seconds till it reached 7 seconds for (12x) which is totally unacceptable for most applications. Consequently, periodically monitoring the system at the network edges will help achieving network QoS requirements. As shown from the above, our simulation results closely matching the analytical results for all performance metrics.

![Fig. 6: Queuing Delay v.s Sampling Interval.](image1)

![Fig. 7: Queuing Delay v.s Number of IoT Publishers.](image2)

V. CONCLUSIONS

The paper introduced IDIoT, an instantaneous network load detection and prevention technique in the IoT networks supported by Publish/Subscribe model (MQTT protocol). The proposed scheme aims to prevent the massive traffic generated by IoT devices as close as possible to the network edges. The proposed scheme collects and analyzes the historical traffic arrival rates and allow the central broker node to react according to the collected arrival rates and the current number of publisher nodes at the occurrence of high demand event. M/G/1 queue analysis and simulations were used to verify the proposed model. We compare our mechanism with the conventional case to evaluate the obtained results. Simulation results showed that understanding the underlying traffic patterns can help to identify high network load at an early stage and without propagating the effects to the system users.

REFERENCES


