Enhancing DoS Attack Recovery in MQTT Brokers for IoT Systems through Efficient Scheduling Algorithms

Pranav Sitaraman
Edison Academy Magnet School
Edison, NJ 08837, United States
sitaraman.pranav@gmail.com

***Abstract*—The exponential rise in the use of IoT devices has produced a growing risk of Denial of Service (DoS) attacks on IoT infrastructure. DoS attacks can prevent critical services from reaching legitimate users by targeting communication protocols such as Message Queuing Telemetry Transport (MQTT). Prior research has touched on algorithms to detect DoS attacks on the MQTT protocol, but the discussion of broker recovery from DoS attacks is not prevalent in the literature. This paper proposes the use and adaptation of efficient scheduling algorithms for minimizing tardy processing time to enhance the on-time delivery of backed-up queries to MQTT brokers following DoS attacks.**

***Keywords*—MQTT, Internet of Things, Denial of Service, scheduling, minimum tardy processing time**

I. Introduction

The use of IoT devices in embedded systems has skyrocketed over the past few years. A 2020 Congressional Research Service report [1] predicted the number of active IoT devices to more than double from 2019 to 2025 and reach over 20 billion in number, while the market is expected to grow to $1.5 billion by the same year. This surge can largely be attributed to cheaper, faster, and more manufacturer-friendly embedded systems and their wide variety of uses in "smart" consumer appliances.

Home automation systems, in particular, have seen a surge in popularity, with a wide variety of consumer products developed by tech companies such as Google and Amazon. Home automation is ripe for IoT systems, which can analyze data gathered by sensors and learn about their environment to make decisions for humans. The ability for IoT devices to connect to networks, share information between sensors and actuators, and accumulate data in endpoints easily accessible to users have been defining contributors to their popularity in home automation. Today, home automation often includes an intricate network of smart appliances, entertainment systems, and security devices.

However, the security of IoT devices, particularly within the context of home automation, is a topic of concern. The aggregation of consumer data in the hands of third-party providers has the potential to leak massive amounts of data if subject to cyber-attacks. While collecting data to a single access point is convenient, it can also leave these systems vulnerable. Furthermore, IoT devices introduce unsecured endpoints into networks that can be used to generate botnets and spoof DDoS attacks, affecting even those far beyond one's home.

In light of this threat, this paper proposes the use of efficient scheduling algorithms to improve recovery from DoS attacks against MQTT, a popular communication protocol used by IoT devices to send data over a TCP network.

II. MQTT Architecture and Protocol

A. Background on MQTT

Message Queuing Telemetry Transport (MQTT) is a publish-subscribe (pub/sub) network protocol for message queuing. The protocol uses ordered, lossless, bi-directional connections over TCP/IP and is thus suitable for Internet of Things (IoT) applications such as home automation systems.

The MQTT protocol divides endpoints into MQTT clients (devices such as micro-controllers) that connect to a MQTT broker, which receives messages from sender clients and routes messages to destination clients. Messages contain data to be transmitted and are identified by "topics" to which clients subscribe. The protocol is popular with IoT devices for being lightweight and requiring limited network bandwidth. The structure of the MQTT architecture is described by Fig. 1.

The MQTT protocol allows for multiple qualities of service, which are classified in increasing order of overhead. A quality of service of 0 indicates a message to be sent at most once with or without acknowledgement. A message with QoS 1 is re-sent multiple times until acknowledgement, while a quality of service of 2 is sent exactly once with a four-step handshake to ensure acknowledgement. Other features of MQTT include the ability for brokers to retain messages for offline clients until re-subscription and to provide a last will and testament in the case of unexpected disconnection from the broker.

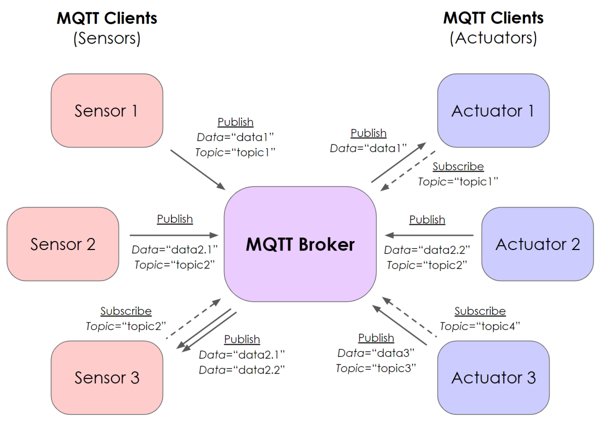


Fig. 1: Diagram of the MQTT architecture.

B. The MQTT Broker

The MQTT broker is a centralized server that receives messages published by sender clients and routes these messages to destination clients subscribed to the appropriate topic. The broker can be run locally or on the cloud and is an intermediate central hub.

MQTT brokers may sometimes provide Transport Layer Security (TLS) or Secure Sockets Layer (SSL) encryption for secure communication, but these protocols greatly reduce the efficiency of the broker. The protocol is most popularly used in embedded systems with limited bandwidth and resources, and thus encryption heavily erodes the viability of the protocol.

C. DoS Attacks Against MQTT

Denial of Service attacks, abbreviated DoS, are types of attacks that disrupt clients from accessing a host or service through a network. Flooding a service with unwanted traffic results in it being unable to host legitimate users.

DoS attacks come in two broad types. Smurf Attacks involve flooding the target server with Internet Control Message Protocol (ICMP) packets, while a SYN flood creates an incomplete connection that occupies host ports and prevents clients from being able to connect. [2]

Previous literature has categorized DoS attacks against MQTT into three broad categories: [3]

1) **BF_DoS.** A malicious threat continuously prompts broker connection using random usernames and passwords.
2) **Delay_DoS.** An attacker begins a TCP handshake but sleeps before completing the connection.
3) **Sub_DoS.** An illegitimate user subscribes to many large topics simultaneously upon connection.

At the same time, various sources have explored methods to detect DoS attacks against the MQTT protocol, whether using machine learning models [4] or a "fuzzy logic-based approach" [5].

D. Recovery from DoS Attacks

Despite the breadth of literature on detecting DoS attacks, particularly on vulnerable protocols such as MQTT, there has been an evident lack of attention toward recovery from these attacks. Recovery from a DoS attack is especially important for protocols such as MQTT that operate with a "pub-sub" architecture, whereby these publish/subscribe requests are sent to a queue that is then executed in a FIFO (first-in-first-out) manner

This paper suggests addressing this issue by utilizing scheduling algorithms to schedule "backed-up" queries in an ideal manner. Such an algorithm seeks to maximize the amount of information that can be sent within a message's allowable time frame (before it is replaced with new information). Rephrasing this objective as minimizing the amount of "expired" data sent lends itself to a discussion of the MTPT problem.

III. The MTPT Problem

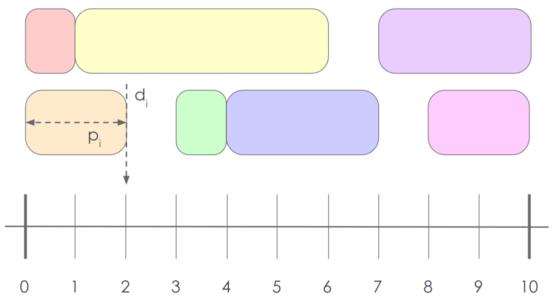
The Minimum Tardy Processing Time (MTPT) problem is a scheduling problem that seeks to minimize the total processing time of tardy jobs on a single machine. This is a classical scheduling problem, first considered by Lawler and Moore in 1969.

The input to the MTPT problem is n jobs, each associated with a due date d_i and processing time p_i. The output of an algorithm for the problem is a non-preemptive schedule of these jobs on a single machine that can execute one job at a time. Given that a job is tardy if it terminates after its due date, the schedule aims to minimize the total processing time of all tardy jobs. A visualization of this can be seen in Fig. 2.

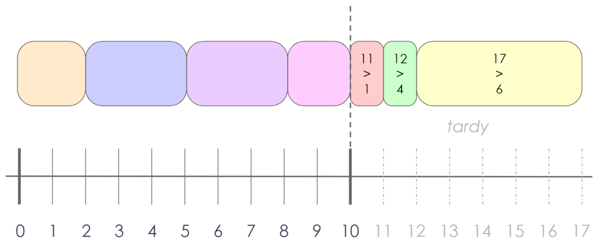
Lawler and Moore [6] showed that the MTPT problem is weakly NP-hard as a generalization of the Subset Sum Problem (SSP). Using a dynamic programming approach, they gave an $\mathrm{O}(P \cdot n)$ time algorithm for this problem. They also showed that the set of jobs in the optimal solution can always be arranged in an EDD (Earliest Due Date) schedule, in which jobs with earlier due dates are completed before jobs with later due dates.

Recently, it was shown that this problem can be solved efficiently by computing (max,min)-skewed-convolutions. Bringmann et al. [7] extended the (max,min)-convolution to a skewed form and presented an $\tilde{\mathrm{O}}(P^{\alpha})$ time algorithm for the MTPT problem, where $\tilde{\mathrm{O}}(P^{\alpha})$ is the running time of a (max,min)-skewed-convolution of vectors of size P. They then gave an $\tilde{\mathrm{O}}\left(P^{7/4}\right)$ time algorithm for a (max,min)-skewed convolution and thus for the MTPT problem as well. Klein et al. [8] improved this result with an $\tilde{\mathrm{O}}\left(P^{5/3}\right)$ time algorithm for a generalized version of the (max,min)-skewed convolution. The resulting algorithm is referred to as ConvScheduler.

Recently, Schieber and Sitaraman [9] have shown an $\tilde{\mathrm{O}}\left(P^{2-1/\alpha}\right)$ algorithm for the MTPT problem. Rather than attempting to improve on the algorithm for computing a (max,min)-skewed-convolution, they use the novel approach



(a) Input: a set of jobs, represented as bars with width equal to processing time and right end-line equal to due date.



(b) Output: the minimum sum of processing times of tardy jobs, $17 - 10 = 7$.

Fig. 2: A visualization of solving the MTPT problem.

of "bundling" jobs from adjacent due dates. Using the iterative nature of the (max,min)-skewed-convolution, they then compute the possible processing times of jobs in these bundles before combining the results into an overall solution. When coupled with the results of Klein et al., this produces an $\tilde{O}(P^{7/5})$ algorithm referred to as SolvScheduler that is currently the fastest known algorithm for the MTPT problem and will be the algorithm used in the remainder of this paper.

Implementations of the algorithms developed and referenced above can be found at [10], and a runtime analysis is given by Fig. 3.

IV. Applying MTPT to MQTT Brokers

A. Demonstration of Equivalence

Consider that following a DoS attack that has been detected and resolved, the queue of the MQTT broker will be filled with publish, subscribe, and unsubscribe queries sent by clients during the period for which the broker was overwhelmed by illegitimate requests. For the sake of ensuring message delivery in the critical time following a DoS attack, the subscribe queries can be moved to the front of the queue while the unsubscribe queries can be moved to the end of the queue. This ensures that all subscriptions continue to apply to all of the backed-up data. Both groups of requests can be evaluated in the same order in which they were made, leaving simply the publish queries remaining to deal with.

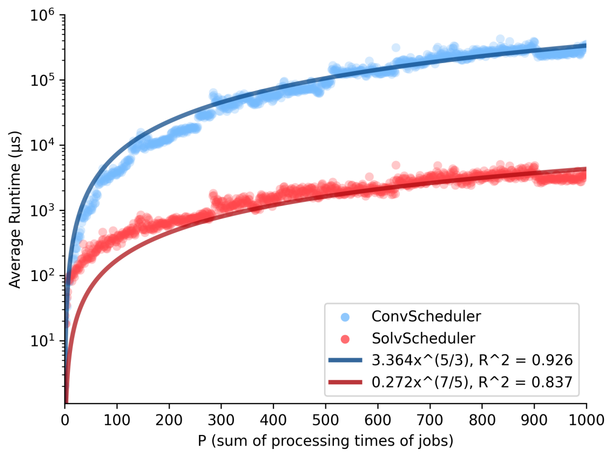


Fig. 3: Runtime analysis of implementations of MTPT algorithms.

In implementations such as home automation systems, publish queries tend to be sent at regular intervals by sensors in order to enable actuators to maintain a homeostatic balance and respond to deviations from expectation. The rate at which data is produced and shared with the broker can be quantified by the variable l, the maximum allowed latency of a message. A system that sends data at shorter intervals will therefore have a lower value of l, as the published data is quicker to expire and be replaced by more recent data. Similarly, a system that is more critical will have a smaller l, as action must be taken upon the data immediately.

Furthermore, in larger systems, the topic and data sent over the MQTT protocol may, in combination with a higher volume of information, be sufficiently large enough to warrant dropping certain messages when the queue is backed up. This will be done in favor of sending new data from the sensor which will enter the queue a short interval later. The processing time of each message is proportional to the amount of data to be transmitted.

Consider the objective function to be maximizing the amount of data sent on time; this is evidently equivalent to the MTPT problem. The time of DoS resolution can be represented by $t = 0$, and for message i with length $|i|$, there exists the equivalence $d_i = l_i$ and $p_i = \alpha \cdot |i| + \beta$ for some weights α and β.

B. Modifications

It must be noted that the algorithm for MTPT presented above must be slightly modified to be useful for this problem. In seeking to minimize tardy processing time, MTPT simply finds the maximum length of jobs (i.e. amount of data) that can be scheduled on-time rather than exactly which jobs should be scheduled for this to occur. This is because the algorithm relies on quickly computing sumsets and subset sums by performing polynomial multiplication using the Fast Fourier Transform (FFT) as described in [7].

However, this method cannot keep track of the origin of sums in the final result (i.e. it cannot recognize which $a \in A$, $b \in B$ resulted in $c \in C$ with $c = a + b$).

In order to accomplish this, the sumset and subset sum operations must use classical polynomial multiplication algorithms. This runs in $O(n^2)$ time compared to the $O(n \log n)$ required when using FFT. With this change to the algorithm and the addition of a method for indexing jobs/messages, the algorithm for scheduling queries when recovering from DoS attacks is complete. The implemented algorithm can be found at [11].

C. Analysis

To analyze the performance of the MTPT algorithm for recovery from DoS attacks, the total data of queries executed before their respective maximum latencies is compared to that of a naive algorithm that simply executes the MQTT queue in order. This is displayed in Fig. 4.

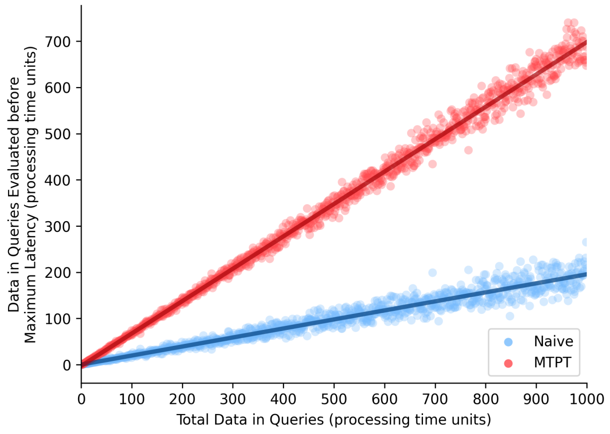


Fig. 4: Quantity of data sent on-time by MTPT versus naive schedulers.

MTPT consistently outperforms the naive algorithm by a constant factor, and the amount of data evaluated on time follows a strong linear correlation. It can be experimentally observed that the MTPT algorithm appears to evaluate $\approx 70\%$ of the data in queries before their maximum latency. This is nearly 3.5 times better than the naive algorithm that is the standard implementation of MQTT brokers.

V. Conclusions

As this paper demonstrates, the recovery of MQTT brokers from DoS attacks can be improved with the use of efficient scheduling algorithms. The equivalence of queries in the MQTT broker queue to a sequence of jobs in a scheduling algorithm serves as a powerful tool for this problem. A modified implementation of the algorithm for MTPT given in [9], which seeks to minimize the processing times of tardy jobs on a single machine, performs on average over 3.5 times better than a naive algorithm that evaluates queries to the broker in a FIFO order.

As this is one of the first discussions of an algorithm for improving the recovery of the MQTT protocol from DoS attacks, it will be interesting to see whether increased attention to this aspect of the problem can produce further improvements. A more complex representation of the problem, factoring in the criticality of various messages and client connections, could further improve the practicality of the algorithm for real-world implementations.

References

[1] *The Internet of Things (IoT): An Overview*, Feb 2020. [Online]. Available: https://sgp.fas.org/crs/misc/IF11239.pdf

[2] "Understanding denial-of-service attacks," Jul 2023. [Online]. Available: https://www.cisa.gov/news-events/news/understanding-denial-service-attacks

[3] A. Alatram, L. Sikos, M. Johnstone, P. Szewczyk, and J. Kang, "Dos/ddos-mqtt-iot: A dataset for evaluating intrusions in iot networks using the mqtt protocol," *Computer Networks*, vol. 231, p. 109809, 2023.

[4] M. Al-Fayoumi and Q. Abu Al Haija, "Capturing low-rate ddos attack based on mqtt protocol in software defined-iot environment," *SSRN Electronic Journal*, 2023.

[5] H. A. P. and K. K., "Secure-mqtt: An efficient fuzzy logic-based approach to detect dos attack in mqtt protocol for internet of things," *EURASIP Journal on Wireless Communications and Networking*, 2019.

[6] E. L. Lawler and J. M. Moore, "A functional equation and its application to resource allocation and sequencing problems," *Management Science*, vol. 16, pp. 77–84, 1969.

[7] K. Bringmann, N. Fischer, D. Hermelin, D. Shabtay, and P. Wellnitz, "Faster minimization of tardy processing time on a single machine," *Algorithmica*, vol. 84, pp. 1341–1356, 2022.

[8] K.-M. Klein, A. Polak, and L. Rohwedder, "On minimizing tardy processing time, max-min skewed convolution, and triangular structured ILPs," in *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. USA: Society for Industrial and Applied Mathematics, 2023, pp. 2947–2960.

[9] B. Schieber and P. Sitaraman, "Quick minimization of tardy processing time on a single machine," *Lecture Notes in Computer Science*, p. 637–643, 2023.

[10] PranavSitaraman, "PranavSitaraman/MTPT." [Online]. Available: https://github.com/PranavSitaraman/MTPT

[11] ——, "PranavSitaraman/MQTTScheduler." [Online]. Available: https://github.com/PranavSitaraman/MQTTScheduler