Towards SDN-based Fog Computing: MQTT Broker Virtualization for Effective and Reliable Delivery

Yiming Xu, V. Mahendran, and Sridhar Radhakrishnan
School of Computer Science
University of Oklahoma, Norman, Oklahoma 73019
Email: {yiming.xu, mahendran.veeramani, sridhar}@ou.edu

***Abstract*—Performance of data analytics in Internet of Things (IoTs) depends on effective transport services offered by the underlying network. Fog computing enables independent data-plane computational features at the edge-switches, which serves as a platform for performing certain critical analytics required at the IoT source. To this end, in this paper, we implement a working prototype of Fog computing node based on Software-Defined Networking (SDN).**

Message Queuing Telemetry Transport (MQTT) is chosen as the candidate IoT protocol that transports data generated from IoT devices ($a.k.a.$ MQTT publishers) to a remote host (called MQTT broker). We implement the MQTT broker functionalities integrated at the edge-switches, that serves as a platform to perform simple message-based analytics at the switches, and also deliver messages in a reliable manner to the end-host for post-delivery analytics. We mathematically validate the improved delivery performance as offered by the proposed switch-embedded brokers.

I. Introduction

Internet-of-Things (IoTs) find practical use in a broad spectrum of applications. With a potential increase in the number of IoT devices, the need for big-data analytics becomes a practical necessity; however, the performance of such analytics depends on the pace of delivery offered by the underlying network transport. Moreover, certain critical analytics such as detecting hazardous events require quick response. Critical analytics near the data source enables timely actions to performed in controlling the hazard. The right place to utilize computational resources for performing analytics would be at the edge switches. Utilizing the edge switches' resources and services for the end-users is called edge computing or Fog computing [1]. In a metamorphic perspective, 'Fog' is closer than the 'Cloud'. Hence the concept of computing at the edge-switches (closer to source of data generation) is termed as 'Fog computing'.

In this paper, we propose and implement Fog computing architecture at the edge switches using Software-Defined Networks (SDNs). SDN enables programmability to the network switches, and also provides a centralized control-plane controller to enable routing decisions by appropriately utilizing available network resources. While integration of SDN and IoT has been an active field of research in the recent past [2], to the best of our knowledge, exploiting SDN to perform Fog computing has not been explored yet.

As a first step towards using SDN for implementing Fog computing, we have custom-modified the switch-side SDN code to integrate a built-in controller. In this manner, the switch behaves as a discrete functional device that extends services and resources to the end-users. However, without loss of generality the SDN controller can be used external to the switch (as in typical SDN environment), or can be embedded within the switch; based on individual needs.

We chose Message Queuing Telemetry Transport (MQTT) [3] as the candidate IoT protocol for our implementation. The MQTT is a content-based publisher-subscriber model designed on top of traditional transport protocols such as TCP and UDP. MQTT architecture comprises of two components namely, MQTT clients (such as publishers and subscribers) and MQTT broker. The MQTT broker is the main component for mediating messages between publishers and subscribers. In other words, both subscribers and publishers exchange messages through the broker node. IoT applications envisioned to connect small battery-cell powered devices to the Internet, are typically of low-form factor that generate/publish data in sporadic manner. However, potential large-scale deployments of such IoTs create a huge aggregated traffic to the Broker nodes that causes congestion, and thereby reduced throughput (messages per second) in the network.

SDN enables programmability in the network stacks thereby offering flexibility and manageability to the network designers. To adapt MQTT-IoT applications to large-scale operations, we develop and implement an SDN-based proxy broker to play the role of aggregating independent MQTT clients' traffic for effective transport in the network. We also mathematically investigate the improved performance and study the throughput's deviation from mean [4]; with the help of large-deviations theory. Our contributions in this paper are given below:

- We propose a novel SDN-based Fog computing architecture, for basic IoT functionalities and analytics.
- Validate the Fog node's throughput analytically.

II. Related Work

Of late, the research community recognizes SDN as one of the promising solutions to solve resource management needs of the IoT ecosystem [2],[5]. Subsequently, Fog computing is recognized as a key technique to perform big data analytics without suffering from the drawback of cloud, high latency. A federation of fog and cloud can handle big data acquisition, aggregation, and reducing data transportation [1]. However, SDN's centralized architectural design makes it difficult to implement Fog computing at the switches. Our work complements SDN with virtualization functionalities, by constructing an integrated controller to perform discrete and possible distributed computation at the edge-switches.

The authors in [6] have proposed a hierarchical Fog computing architecture for big data analysis, and through the experiments the Fog computing achieved significantly improved response times for detecting hazardous events.

III. MQTT-Background

MQTT [3] is an open-source protocol developed by IBM; specifically targeted at IoT devices attributed by unreliable networks and restricted resources (such as high delay and low bandwidth) [7]. MQTT works on a context-based message forwarding paradigm with publisher-subscriber model. Each generated message is associated to specific 'topic', and clients subscribed to those 'topic' will receive the associated messages.

The basic components of MQTT includes the following:

- **MQTT Publishers**: IoT devices generate (publish) data to an end-host (namely, MQTT broker). The published data is of the format $<$ topic,data $>$.
- **MQTT Subscribers**: The end-hosts (such as actuators, PC) that receive published messages based on the 'topic' they have subscribed.
- **MQTT Broker**: The end-host (server machine) that acts as a central node, connecting both MQTT publishers and MQTT subscribers. The broker node collects publish-data generated by the IoT devices ($a.k.a.$ publishers).

The connections between publishers, broker, subscribers are enabled using standard transport protocols such as TCP and UDP. As the IoT devices (MQTT-publishers) are highly resource constrained devices, they establish connection (say, using TCP) with the broker whenever a new data needs to be published. On the other hand, the MQTT subscribers typically high computing devices (such as PCs) will maintain connections with the broker all the time to receive topic-based publish-messages. The ideal place to perform analytics is the MQTT-broker, which is a central repository of all the published data. We henceforth, focus our attention on the MQTT-publishers and MQTT-brokers through out in this paper, as shown in Fig. 1.

IV. SDN-based Fog Computing

In this section, we present our proposed architecture for SDN-based Fog Computing. For the MQTT network considered in Fig. 1, we enable the functionality of the broker node in the edge switch ($i.e.$, first mile switch connecting the MQTT publishers). This edge-switch with the broker functionality is henceforth called the 'Fog node'. The Fog node architecture is shown in Fig. 2. In order to enable the Fog node to behave as an independent computing

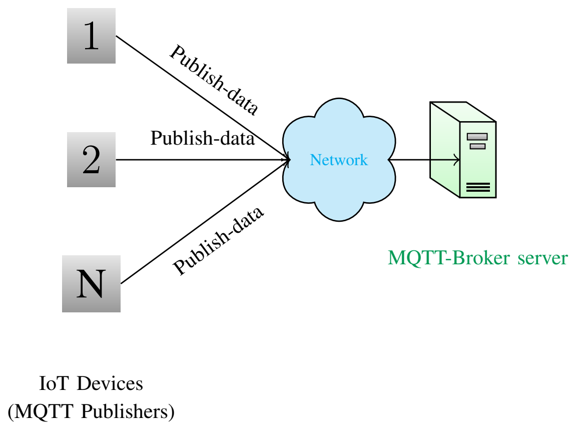

Fig. 1. MQTT Publisher-Broker Architecture

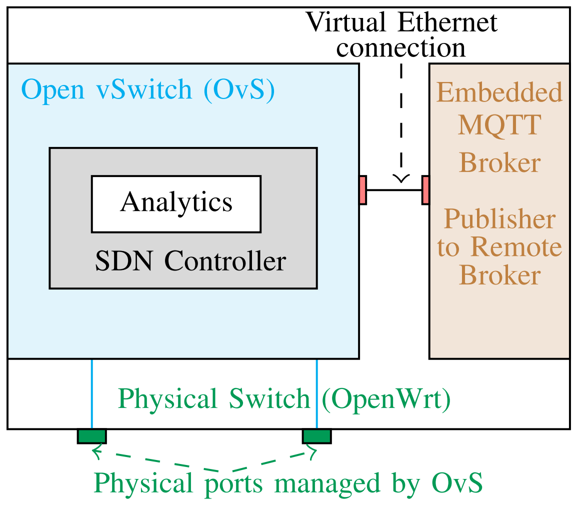
Virtual Ethernet connection
Open vSwitch (OvS)
Analytics
SDN Controller
Embedded MQTT Broker
Publisher to Remote Broker
Physical Switch (OpenWrt)
Physical ports managed by OvS

Fig. 2. SDN-based Fog computing architecture

node, we have custom modified the SDN switch-side code to integrate the SDN-Controller within the switch hardware.

The Fog node serves the following purposes:

- Behaves as actual broker for MQTT clients,
- Serves as a platform for performing analytics at the Fog node.
- Needs to communicate with the end-host broker server for storage and exhaustive deferred analytics. It should be noted, that an external communication from a Fog node serves multiple purposes such as connecting with another Fog node; in a distributed brokers' environment.

The MQTT 'publish-messages' arrive at the physical port of the Fog (switch) node. The switch *integrated* controller acts as a single 'proxy-publisher' will maintain a TCP connection with the 'remote' end-host broker; and publishes all of the received messages from the real MQTT publisher clients.

To respond to the MQTT publishers the Fog node runs the entire MQTT broker integrated within the switch; which is communicated to the SDN controller through virtual-ports (as shown in Fig. 2). Before sending the publish-messages, the MQTT publisher establishes application-layer connections with the Broker. To this end, the publisher will send MQTT Connect message to the broker (Fog node) and upon receiving the Connect-ACK message the actual publish-message is sent.

The Open vSwitch (OvS) communicates to external hosts through physical ports of the switch, and internally with the MQTT broker through virtual port. The SDN controller forwards the MQTT messages between (external) MQTT clients and switch-integrated MQTT broker. In order to send the received MQTT messages from Fog-broker node to the remote broker, we created a separate thread of 'MQTT publisher' running inside the embedded broker that maintains TCP connection with the remote end-host broker.

The SDN controller can serve as a platform for performing analytics by parsing the MQTT payload contents to retrieve topic and associated data-value. For instance, a threshold-based analytics can be performed as follows: The data on the topic 'temperature', can be detected for beyond safe-limit values. Other analytics include statistical analysis of the received data. We plan to incorporate such features in our future implementations.

V. Analysis of Delivery Throughput of Fog Node

The Fog node maintains a TCP connection with the end-host broker, and transports the IoT (MQTT-clients') published messages that are stored for future exhaustive analytics. We are now interested in throughput of the published messages sent by the Fog node in the considered system. Our system model comprises of 10 end-hosts trying to establish 250 TCP sessions over wired connections to the edge switch (*i.e.*, Fog node). Subsequently, the Fog node uses TCP Reno connection to connect with

$$\lim_{\epsilon \to 0} \lim_{n \to \infty} \frac{1}{n} \log \frac{\#\{i \in \{1, \cdots, x_n\} : \overline{S}_i^{(n)} \in [\alpha - \epsilon, \alpha + \epsilon]\}}{x_n} = f(\alpha) \tag{1}$$

Eq. 1 can be equivalently represented as below:

$$\frac{\#\{i \in \{1, \cdots, x_n\} : \overline{S}_i^{(n)} \in [\alpha - \epsilon, \alpha + \epsilon]\}}{x_n} \underset{1 \ll n \ll N}{\sim} e^{nf(\alpha)} \tag{2}$$

the end-host broker for transferring published messages. Each MQTT publish message is considered to use the entire Maximum Segment Size (MSS) of the underlying TCP transport. Therefore, the TCP throughput computation is sufficient to get the throughput of MQTT messages. Without loss of generality, the MQTT throughput is a function of TCP segments' throughput.

IoTs are naturally deployed in large scale and the data generated is considered to potentially fall in the regime of 'Big data'. As a result of continuous data being transported in the network, the TCP connection (at the Fog node) is long-lived and therefore it is sufficient to analyse the throughput of this TCP in its congestion avoidance phase. A long-lived TCP reno's congestion window (in packets) can be modeled by Markov chain [4].

Let the congestion window S denotes a total number of S publish-messages being transmitted by the Fog node. The possible congestion window values are finite, and are given as E $= \{1, \cdots, S_{\max}\}$; where $S_{\max}$ is the maximum congestion window at the sender (Fog) node. The transition matrix $\mathbf{T}$ of the Markov chain representing the congestion window size (on each RTT instant) is denoted by [4]:

$$\mathbf{T}_{S,S'} = \begin{cases} 1 - \mathbf{Pr}(S), & \text{if } S' = \min(S+1, S_{\max}) \\ \mathbf{Pr}(S), & \text{if } S' = \max(\lfloor \frac{S}{2} \rfloor, 1) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

where $\mathbf{Pr}(S)$ represents the loss-probability at the congestion window is of size S. It is reasonable to assume that the Markov chain is irreducible and aperiodic. A classical result on Large-Deviations Principle (LDP) states that an irreducible and aperiodic Markov chain, with finite state space, holds a large deviations spectrum as given below [4]:

$$\lim_{\epsilon \to 0} \lim_{N \to \infty} \frac{1}{N} \log \mathbf{Pr}(\overline{S}^{(N)} \in [\alpha - \epsilon, \alpha + \epsilon]) = f(\alpha) \tag{4}$$

where, $\overline{S}^{(N)} = \frac{1}{N} \sum_{i=1}^{N} S_i$ is the sample mean $\overline{S}$ congestion window size scale N, and $f(\alpha)$ is the large deviations spectrum as representing the Legendre-Fenchel transform of the logarithmic moment generating function Λ, and is given below:

$$f(\alpha) = \inf_{q \in \mathbb{R}} (\Lambda(q) - \alpha q)) \tag{5}$$

For our context, the large-deviations spectrum $f(\alpha)$ can be computed from the Markov chain transition matrix $\mathbf{T}$. The authors in [8] have shown that the large deviations spectrum can be obtained as the spectral radius'(ρ) logarithm of the matrix $\mathbf{R}(q)$. $i.e.$,

$$f(\alpha) = \log \rho(\mathbf{R}(q)) = \log \rho(\exp(qj)\mathbf{T}_{ij}) \tag{6}$$

The probability function in Eq. 4 refers to fractions observed over large number of *independent* realizations ($i.e.$, multiple independent TCP flows). In our study, we are dealing with a single TCP flow between the Fog node and the end host.

Interestingly, the authors in [9] have shown an ergodic form of LDP to hold on almost every realization. Considering a single realization of finite-size $(S_i)_{i \in \{1, \cdots, N\}}$, and its mean at scale n. In other words, the single realization of size N is considered as parts of x_n consecutive intervals of size n where $x_n = \lfloor \frac{N}{n} \rfloor$. The mean over i^{th} interval is given by, $\overline{S}_i^{(n)} = \frac{1}{n} \sum_{y=(i-1)n+1}^{i} S_y$. The result in Eq. 1 holds for almost every (single) realization [4], for a large value of x_n. An essential property of large-deviations spectrum is that it is independent of scale n, and satisfies the following:

$$\begin{cases} \text{if } \alpha = \overline{S}^{(\infty)} & \text{then } f(\alpha) = 0, \text{and} \\ \text{else if } \alpha \neq \overline{S}^{(\infty)} & \text{then } f(\alpha) < 0. \end{cases} \tag{7}$$

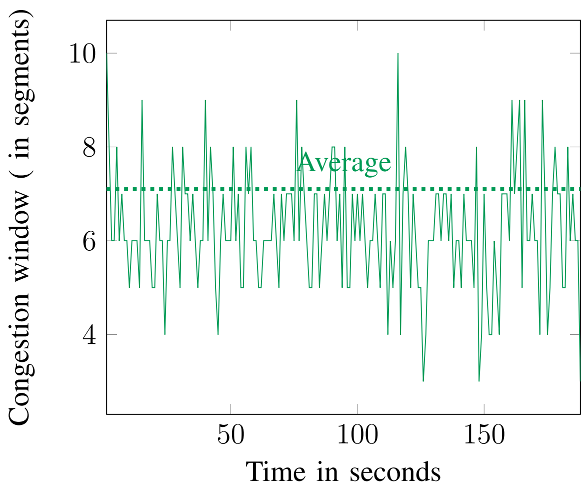

Fig. 3. Congestion window size instantaneous vs average

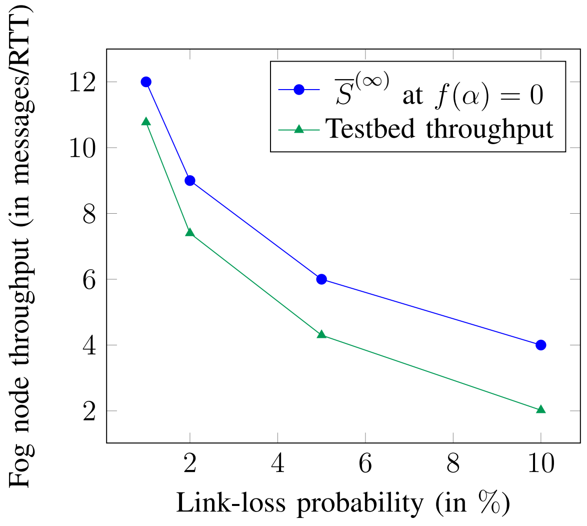

Fig. 4. Average throughput for different loss probability

Therefore it is clear from Eq. 2 that the throughput remains close to mean $\overline{S}^{(\infty)}$ and the probability of being in all other values would exponentially degrade. Figure 4 validates the throughput at Fog node obtained from the testbed, against the analytical throughput obtained satisfying $f(\alpha) = 0$ in Eq. 7.

VI. Testbed Setup

Our testbed environment consists of 2 PCs, and 4 switches (namely, the Mikrotik RB2011IAUS). The PCs run on Ubuntu OS 12.04, the switches run on OpenWrt 15.05. The Fog node; containing integrated SDN controller is run within a virtual switch environment, namely Open vSwitch (OvS) v10.0.

For scalability reasons, and from high computational resource perspective, we ran IoT devices (MQTT publishers) as a set of virtual hosts, and the Fog node as virtual devices both deployed in a single physical PC. The testbed environment is shown in Fig. 5. In our experiments, we ran 10 mininet virtual hosts as MQTT publishers, 1 MQTT broker (Mosquitto - an open-source broker implementation [10]), and 1 virtual switch using OvS. However, without loss of generality, our implementation can run on physical switches and real devices.

Our network consists of 4 physical switches in a line topology with a physical MQTT broker run by an end-host PC. Each of the physical switches 1 to 4 run respective instance of OvS that manages two physical ports, namely the input and output ports. We considered the third switch, namely '*Physical Switch 3*' as the bottle neck by controlling delay, bandwidth, and packet loss probability; through emulation using Network Emulator (NETEM). The internal Fog node Broker receives and processes all of the publishers request- and data messages; and send appropriate feedback. The integrated controller apart from forwarding messages, can also be used to perform in-house analytics (by providing features for parsing MQTT message's payload). For analytical tractability, we used Bernoulli loss model (with loss probability p) in our experiments. Therefore the loss probability function $\mathbf{Pr}(\mathbf{S})$ is given by

$$\mathbf{Pr}(S) = 1 - (1 - p)^S. \tag{8}$$

Fig. 6, shows the throughput results of respective UDP and TCP MQTT clients. For both the cases, the throughput follows the trend roughly close to the analytical throughput. In contrast, the throughput for the same experiments without Fog node is zero; because the input MQTT traffic is significantly higher as it could not compete to establish connections with physical end-host broker. Therefore, it is clear that, for large scale IoTs the Fog node is essential for transportation. For the traditional setup (without Fog node), with native MQTT clients connected to the remote end-host broker, for similar input traffic configurations, the connections were not stable. For initial few moments of time, the throughput reached up to 250 messages per second (which is

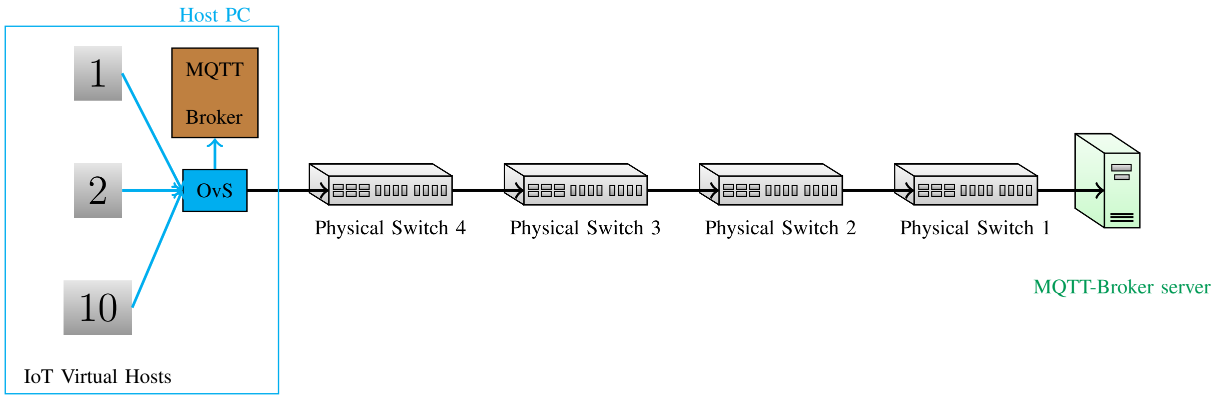

Fig. 5. MQTT Network with Fog Node Testbed Setup. *Each element inside 'Host PC' is run as virtual machine. 'MQTT-Broker' and 'OvS' represent the same architecture as shown in Fig. 2, but running on Ubuntu OS.*

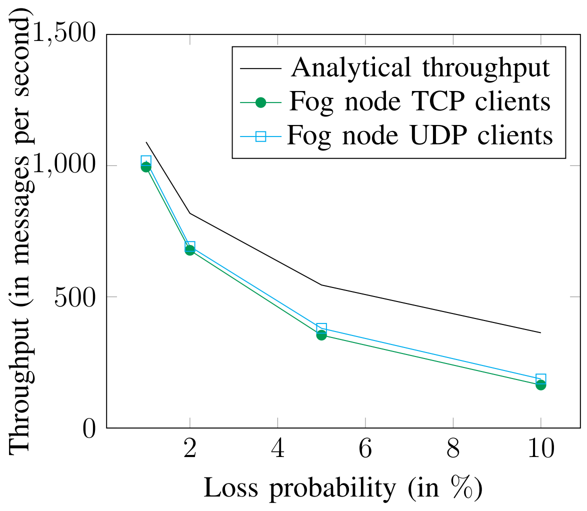

Fig. 6. Throughput performance for respective UDP and TCP clients, with Fog node computing

lower than the Fog nodes throughput performance. Subsequently, the throughput was close to 0, due to the failure of (large number of) TCP handshaking processes attempting to establish connections with the broker.

VII. Conclusion

In this paper, we have proposed an SDN-based Fog computing architecture and developed its working prototype. Subsequently, we have mathematically studied the throughput offered by the Fog node, and our experimental results tend to follow roughly in-line with the analysis. It is also demonstrated that the Fog node delivers at a significantly higher throughput, as compared with the respective traditional client and end-host setup. In future, we plan to study the performance of the considered set up, in the presence of wireless IoTs.

References

[1] S. Yi, C. Li, and Q. Li, "A survey of Fog computing: Concepts, applications and issues," in *Mobidata '15: Proceedings of Workshop on Mobile Big Data*, 2015, pp. 37–42.

[2] K. Sood, S. Yu, and Y. Xiang, "Software defined wireless networking opportunities and challenges for Internet of things: A review," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2015.

[3] "MQTT - Message Queuing Telemetry Transport," http://mqtt.org.

[4] P. Loiseau, P. Gonalves, J. Barral, and P. V.-B. Primet, "Modeling TCP throughput: An elaborated large-deviations-based model and its empirical validation," *Performance Evaluation*, vol. 67, no. 11, pp. 1030–1043, 2010.

[5] A. Hakiri, P. Berthou, A. Gokhale, and S. Abdellatif, "Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications," *IEEE Communications Magazine, IEEE*, vol. 53, no. 9, pp. 48–54, 2015.

[6] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *ASE BD&SI '15: Proceedings of the ASE BigData & Social Informatics*, 2015, pp. 28:1–28:6.

[7] S. Lee, H. Kim, D.-K. Hong, and H. Ju, "Correlation analysis of MQTT loss and delay according to qos level," in *ICOIN '13: Proceedings of the International Conference on Information Networking*, 2013, pp. 714–717.

[8] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*. Jones and Bartlett, Boston, 1993.

[9] J. Barral and P. Loiseau, "Large deviations for the local fluctuations of random walks," *Stochastic Processes and their Applications*, vol. 121, no. 10, pp. 2272–2302, 2011.

[10] "Mosquitto - An open source MQTT broker," http://mosquitto.org.