International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2019): 7.583

# Cheating Detection and Cancellation in (MDA-ALM) Protocol in Application-Level Multicast Networks

**Rasha Sahawish[1], Mothanna Alkubeily[2]**

[1]Tishreen University, Department of Communications and Electronics Engineering, Tishreen University, Lattakia, Syria
*rasha.m.shaweesh[at]outlook.com*

[2]Tishreen University, Department of Mechanical and Electrical Engineering, Tishreen University, Lattakia, Syria
*mothanna.alkubeily[at]gmail.com*

**Abstract:** ***Membership Duration Aware-ALM (MDA-ALM) protocol was suggested to decrease node rejoin ratio in application-level multicast networks, it depends on the announcement of the expected membership duration for each new user in order to build a stable and efficient tree. Although the performance of (MDA-ALM) protocol is good, but it is based on membership duration parameter and this makes it more sensitive for cheating and non-cooperative nodes. The main goal for the cheating nodes is to improve their position in the tree by trying to get the nearest position to the source node and to avoid having any children in order to relieve its load by manipulating the membership duration information. Our research aims to find the best solution to detect the cheating nodes. The simulation results prove that the proposed scheme detects effectively the cheating nodes. Furthermore, we simulate two schemes that we proposed to cancel cheating in order to choose the best between them***.

**Keywords:** Application-level-multicast, Multicast, (MDA-ALM) Protocol, Cheating, Overlay tree, non-cooperative nodes

## 1. Introduction

Application-level multicast network has several advantages [1-6], where the main advantage that it is easy to deploy since it does not require any changes in network layer. In this network, data will be sent through the overlay tree that has been built using a unicast connection between the nodes. Several protocols have been proposed during the past years to build an effective overlay tree. These protocols can be classified into two main categories [3, 7] centralized protocols and distributed protocols.

Centralized protocols require a central node controlling the session Rendezvous Point (RP), which can be defined as a server, this node will collect all the needed information depends on the required input (delay, bandwidth, loss ...) from all session members, then it will build the efficient overlay tree depends on collected information, after finish tree building process the central node will start informing each node about its position within the tree and its neighbors. However, these protocols suffer from a single point of failure problem.

As for the distributed protocols, it requires the presence of a session-controlling node which has less tasks. Each node sends its information toward the central node which has a controlling role by telling the new node about the potential parents has, but here the tree-building process (the decision to join the father) is taken by the new node, not by the controlling node, and this is done based on the chosen protocol. After connecting with the father node, then newnode informs other nodes of its location,therefor this type of protocols suffers from obvious overload.

We proposed Membership Duration Aware-Application-Level Multicast (MDA-ALM) protocol [8],that takes advantage of the announced membership duration of each new joining member in order to construct a more stable and robust overlay tree.

Some nodes in application-level multicast networks behave selfishly in order to improve their position in the tree, become closer to the source node, which improves reception quality, and aims to reduce copying and forwarding operations by accepting few numbers of children.

We studied the impact of cheating nodes on the stability of the overlay tree topology [9]. We analyzed two parameters, ALM tree structure discrepancy and receivers position variation after cheating. We concluded that cheating receivers have a considerable negative effect on the stability of the ALM tree.

We also studied the negative effect of cheating on link stress and link stretch parameters, so we noticed that the tree shape will take one of two shapes: (1) Longitudinal Shape: We noticed here that average link stress will be decreased while average link stretch will be increased because of delay ratio increment towards (RP) node. (2) Transverse Shape: In contrast to the previous, we noticed here that average link stress will be increased while average link stretch will be decreased.

Based on the simulation results, we have also shown that cheating nodes can profit from their cheating behavior in case there are fewnumbers of cheaters in the session, otherwise, the competition between them will be increased and they lose the benefit of cheating.

In this paper, we tried to find a new scheme to detect cheating nodes. It depends on the fact that a cheater node will announce its membership duration very small and it will avoid having any children. Moreover, we introduce two

schemes to cancel cheating nodes effect in order to keep the main benefit of (MDA-ALM) protocol which is decreasing the node rejoin ratio.

The remainder of this paper is organized as follows: In section II, we study cheating in application-level multicast networks with emphasis on (MDA-ALM) protocol. In section III, we propose our cheating avoidance scheme, also we suggest two cheat cancellation schemes in (MDA-ALM) protocol. In section IV, several simulations are conducted to evaluate the efficiency of cheating detection and cancelation on (MDA-ALM) protocol. Finally, conclusion and future work as presented in section V.

## 2. Related Works

### 2.1 Application-Level Multicast Networks:

As the number of Internet users increases [6], [14], [15], data delivery over the internet becomes more challenging as networks get more overloaded and congested. Currently, data exchange through the internet is mainly based on unicast (point-to-point between two computers). So, if millions of users try to stream an important broadcast event like the inauguration of the president instead of broadcasting the data to all users, the data source sends a copy of the data to each of the users so the source keeps transmitting the same packet a million times. This leads to redundant traffic in the network in addition to overloading the data source resulting in inefficient data delivery and an increase in packet loss.

Multicast was introduced as an alternative to unicast in such cases. In multicast, the source sends data to group of interestreceivers in a single transmission each one of those receivers forward the data to a different group of users. There are several types of multicast such as IP, overlay, and application level. In IP Multicast, the multicast process is implemented at the IP level during packet transmission. IP multicast provides an efficient multicast technique for one-to-many and many-to-many real time communication over an IP infrastructure in a network. However, IP multicasting introduces high complexity and serious scaling constraints at the IP layer in order to maintain a state for each multicast group.

As a result of the non-acceptance of IP Multicast[2], the Application layer multicast (ALM) approach was proposed. ALM was proposed as an alternative implementation of the multicast technique to the IP Multicast implementation. ALM builds a virtual topology on top of the physical Internet to form an overlay network. Each link in the virtual topology is a unicast link in the physical network. Therefore, the IP layer provides a unicast datagram service, while the overlay network implements all the multicast functionality such as dynamic membership maintenance, packet duplication, and multicast routing.

The main advantages of an application layer solution are the following[2],[3]:

**1) Easily deployed**
Application layer multicast does not require any changes at the network layer. The construction of a logical structure hides routing complications, the tree will be consisting of end nodes instead of routers. It doesn't require any router support since it used logical topology to hide physical topology.

**2) Absence of multicast routers**
ALM builds a virtual topology on top of the physical Internet to form an overlay network. These networks work at application layer, so they can exploit the capabilities of lower-layer protocols in providing reliability, congestion control, flow control, or security according to the needs of the application.

### 2.2 Membership Duration Aware Application-Level Multicast (MDA-ALM) protocol:

In all the solutions proposed in ALM [8], there is a need to trigger a rearrangement process after each leave event in the session. Indeed, when a member leaves the group, each one of its children should rejoin the overlay tree. In the case of highly dynamic sessions where members frequently join and leave the group, this rearrangement process may be very expensive and the communication within the group may be highly disturbed.

We have proposed a Membership Duration Aware Application-Level Multicast (MDA-ALM) protocol to overcome the rearrangement problem in ALM protocols [8]. The main idea of (MDA-ALM) technique is to take advantage of the announced expected membership duration of each member in order to construct an efficient, robust, and stable overlay tree.

#### 2.2.1 Join process
The new member joining process can be describe as below:
1) The new member sends a join request message to the central node. This message contains the new member IP address and its announced expected membership duration MD.
2) The central node calculates the Candidates List which contains all members that will remain in the session after the departure of new member. Here, the Candidates List consists of the candidate node's IP addresses and corresponding RTT (RTT: Round Trip Time) values from the central node.
3) The central node sends the Candidates List to new member.
4) The new member measures the RTT to all candidate nodes in the candidate list. It then adds measured RTTs to corresponding RTTs in the candidate list.
5) The new member determines the best candidate which gives the minimum RTT from the RP to serve as its parent.
6) The new member contacts its parent and starts receiving data from it.

#### 2.2.2 Leave process
When a member Mi wishes to leave the multicast group, it should be a leaf node in the overlay tree. Mi sends a leave

message to its parent which simply deletes it from the list of its downstream children.

### 2.3 Cheating in Membership Duration Aware Application-Level Multicast (MDA-ALM) Protocol:

#### 2.3.1 Cheating Model

The performances of (MDA-ALM) protocol are highly based on the trust in membership duration (MD) information delivered by each new member. This makes the protocol very sensitive to cheating or no-cooperative nodes. The goal of a cheating member is to improve its position in the ALM tree by trying to be connected as close as possible to the source node.

In addition, it will try to prevent other joining nodes from taking it as a parent. By doing this, a cheater will receive the session traffic with a minimum end-to-end delay and will have less replication and forwarding overhead. In order to achieve these two goals [16], an (MDA-ALM) cheater will announce a very small expected membership duration(MD ~ 0) in its joining message that has been sent to the source.

The source will respond with a parent Candidate List containing all nodes in the overlay since they will all be assumed to leave the session after this joining member. The cheater will, then, choose the best parent to be as close as possible to the source.

At the same time, by announcing a Membership Duration (MD ~ 0), no other joining node will take the cheater as a parent.

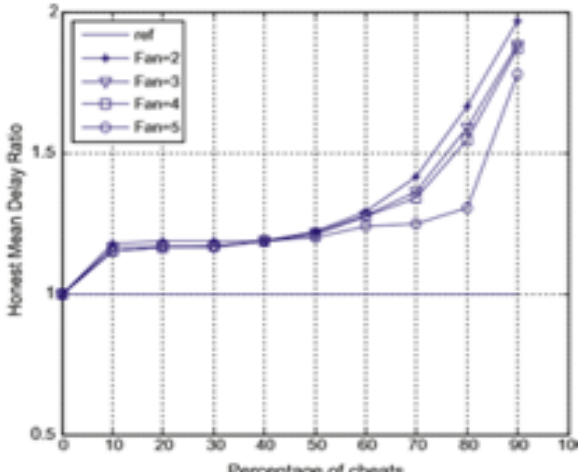
#### 2.3.2 Impact of Cheating in (MDA-ALM) Protocol

In Figure.1 we plot the mean end-to-end delay ratio for honest and cheating members for various values of the fanout while varying the percentage of cheaters among members in the session [16]. In order to study the impact of cheating nodes on the performances and on the stability of overlay tree constructed by our (MDA-ALM) protocol, we considered our simulation will run on a fixed group size to 150 members and varied the maximum allowed fanout from 3 to 5.

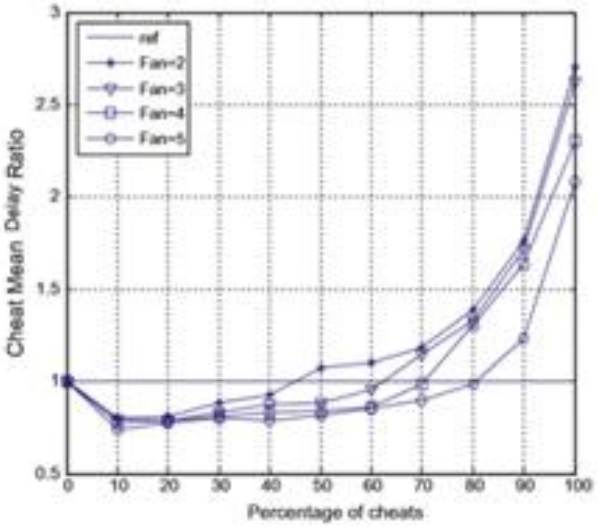
We varied the percentage of cheating members in the session from 10% to 100%. For each set of simulations, we calculated the mean end-to-end delay ratio for honest members only and also calculated the same parameter for cheating members only, a reference session is carried out using only honest nodes.

We can notice that the mean end-to-end delay ratio for honest members increases when some cheaters are present in the session. This degradation becomes more important with the increase in the number of cheaters. This is due to the fact that honest members are moved far away from the source, particularly with trees having small fanout.

Indeed, when the number of cheaters is low (less than 50% for fanout = 2), cheating nodes improve their end-to-end delay since they are able to connect close to the source. However, if the number of cheaters increases in the session (more than 50% for fanout = 2), they will no longer be able to benefit from cheating due to the competition between them.




a. Mean end-to-end delay Ratio for Honest members




b. Mean end-to-end delay Ratio for cheating members

**Figure 1:** Mean End-to-End Delay Ratio in MDA [16].

From the above, we conclude that thecheating nodes achieved their goals; so, we tried to find a way to detect cheating in (MDA-ALM) protocol by applying a new schematic depends on membership duration parameter, we should mention that there is no other solution in application-level multicast networks except for ESM (End System Multicast) protocol [17] which applied a schematic that depends on distance parameter.

## 3. Cheating Detection and cancellation in (MDA-ALM) Protocol in Application-Level Multicast Networks

### 3.1 Overview of Our Cheating Avoidance scheme in (MDA-ALM) Protocol:

Our Scheme will assume that if Membership duration is too small and the node has no children then we can say that this is a cheating node,

If (MD (Nj) $<MD_{cheat}$ & & Child (Nj) = 0) then Nj is a cheating node.

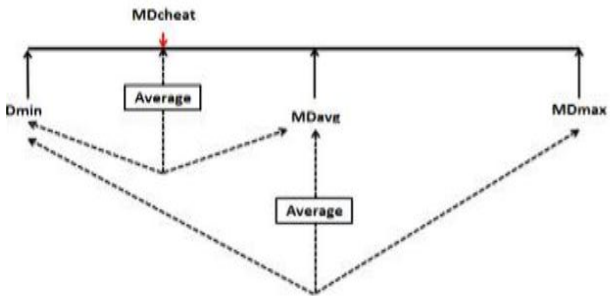



**Figure 2:** Detect cheating boundaries in MDA Protocol.

We suppose the maximum membership duration (MDmax), the minimum membership duration (MDmin), and the average membership duration

$$(MDavg) = (MDmax+MDmin)/2. \quad (1)$$

In order to detect cheating nodes as appeared in figure.2:

1) We will detect cheating boundaries:
   We can say that there is a selfish node $N_j$ if Member ship duration MD ($N_j$) is less than threshold cheating membership duration. But the question here is how to obtain $MD_{cheat}$. We can obtain $MD_{cheat}$ as following:

$$MD_{cheat} = (MDavg+MDmin)/2 \quad (2)$$

2) Choose any member except the source, say $N_j$ for example and repeat:
   If (MD ($N_j$) <$MD_{cheat}$&&Child ($N_j$) =0) then $N_j$ is a cheating node, add it to the cheating members group.
3) Calculate $MD_{avg}$.

### 3.2 Overview of Our Cheating cancellation scheme in (MDA-ALM) Protocol:

Two Schemes have been proposed to cancel cheating:

**1) First Scheme:**

Forcing the cheating node to leave. When we force the cheating node to leave, we guarantee that the cheating cancellation ratio will be high, and the rearrangement overhead ratio will be low because the cheating node has no children.

**2) Second Scheme**

We have noticed that the membership duration of the cheating node is less than $MD_{cheat}$ so, in order to cancel cheating we suggested increasing the membership duration of the cheating node by adding $MD_{avg}$ to its membership duration, it will be reflected positively on the tree building process.

By increasing the membership duration of the cheating node, we guarantee that the cheating node will become one of the nodes in the candidates list, this meaning it will become a parent node, and thus the impact of cheating is minimized as possible.

Through simulations, we will study the two suggested schemes and see which one is the best way to take it into account.

## 4. Results and Analysis

### 4.1. Simulation Environment and Setup

We did the simulation using python simulator [10]. Python is a high-level and general-purpose programming language, whichconstructs an object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python supports multiple programming paradigms [11], including structured, object-oriented, and functional programming.

Python was created in the late 1980s, and first released in 1991, by Guido van Rossum as a successor to the ABC programming language. Python 2.0, released in 2000, introduced new features, such as list comprehensions, and a garbage collection system with reference counting, and was discontinued with version 2.7 in 2020.

Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible and much Python 2 code does not run unmodified on Python 3. With Python 2's end-of-life, only Python 3.6.x and later are supported, with older versions still supporting e.g. Windows 7 (and old installers not restricted to 64-bit Windows).

As of December 2020 Python ranked third in TIOBE's index of most popular programming languages, behind C and Java [12], [13]. In our research, we used python 2.4.2 which is a free license.

The simulations were carried over a set of random flat graphs generated using a modified version of Waxman algorithm written using NetworkX2 [18] python library. This technique constructs graphs having similar properties as the Internet networks.

In our simulations we considered the following case of study: We varied the group size from 25 to 250 members and varied the maximum allowed fanout from 3 to 6. We used models presented in [19]- [21] in order to generate real multicast sessions.

These models can be modeled as follows:

- The users arrive in a multicast group according to a Poisson process with rate λ (arrivals/time unit).
- The membership duration of a member in the group is an exponential distribution with a mean duration equals to 1/μ time units.
- The average number of concurrent users in a subgroup is then given by λ /μ.

During our simulation, we have noticed that there were some outliers in membership duration values, which had a major impact on data static analysis, so the anomalies were diagnosed by Tukey [22] method taking into consideration one variable status.

Also, we applied a specific technique for cheating: we varied the percentage of cheaters from (10-100) % taking into consideration that when the percentage of cheaters exceeds 60% the network becomes ineffective

### 4.2 Performance evaluation metrics

In order to evaluate the performances of the constructed overlay tree [7], [8] and to investigate the impact of cheating nodes on our MDA protocol, we used the following performance metrics:

1) The Rearrangement Overhead (RO): This metric measures the stability of the overlay tree. It is calculated as the total number of rearranged nodes in the session. These are nodes forced to rejoin the overlay tree after a leave event.
2) The Mean End-to-End Delay (ME2EDR): This metric measures the average end-to-end delay from the source to each group member. This metric is calculated as the ratio of the ME2ED in a session with cheating receivers over the ME2ED in a reference session with only honest receivers. This ratio is measured for cheating and honest receivers separately.
3) The Link Stretch Ratio: It measures the penalty paid by a node for receiving data on an application-level tree rather than directly from the source (unicast path). It is defined as the ratio

$$D_{ALM}/D_U \qquad (4)$$

Where $D_{ALM}$ is the latency from the source to the receiver observed along the overlay tree, and $D_U$ is the Unicast delay from the source to the receiver.

Similarly, to the mean end-to-end delay ratio, we study here the link stretch ratio for cheating and honest members separately.

### 4.3 Simulation results

#### 4.3.1 Cheating detection efficiency simulation results:

In order to study cheating detection efficiency, we considered two cases: In the first, we fixed the fanout to 3 and varied the mean group size from 25 to 250 members. In the second, we fixed the group size members [group size is constant and pre-determined during one simulation process] and varied maximum allowed fanout from 3 to 6. In these two cases we are varying the percentage of cheating members from (10 to 100) %.

**(a) Fixed Fanout=3**

In figure.3 we study cheating detection ratio for various values of group size while varying the percentage of cheaters among members in the session. We noticed that our cheating detection algorithm has shown excellent results, because it was able to detect cheating 100% while cheating ratio varied between (10 to 50) %, on other hand cheating detection ratio decreased with the increasing of cheating ratio. As we can see cheating detection ratio reached 50% when cheating ratio was 70%. This percentage keep decreasing with the increasing of cheating ratio, and that is acceptable taking into consideration the network becomes in effective after cheating ratio exceeds 60% for fanout =3.

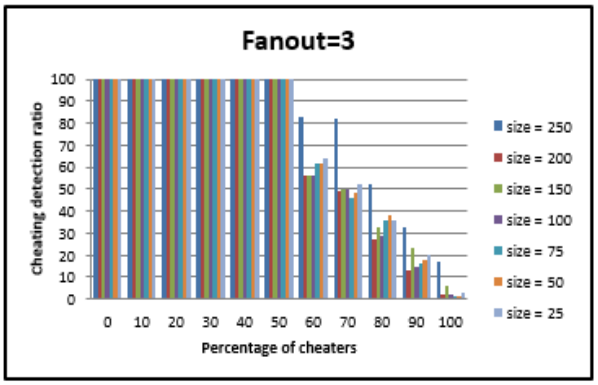



**Figure 3:** Cheating detection ratio in Scenario 1 for different group sizes

**(b) Fixed group size**

In figure.4, figure.5 and figure.6 we study cheating detection ratio for various values of fanout members [group size is constant and pre-determined during one simulation process] while varying the percentage of cheaters among members in the session. We noticed that our cheating detection algorithm has shown good results, but this algorithm has proven efficiency with the increment of fanout when group size is fixed and it was able to detect cheating 100% while cheating ratio varied between (10 to 50) % when fanout equal to 3, but cheating detection ratio increased with the increase in a number of children (fanout), where the detection ratio was 100% when cheating ratio was 60% and for fanout =4.

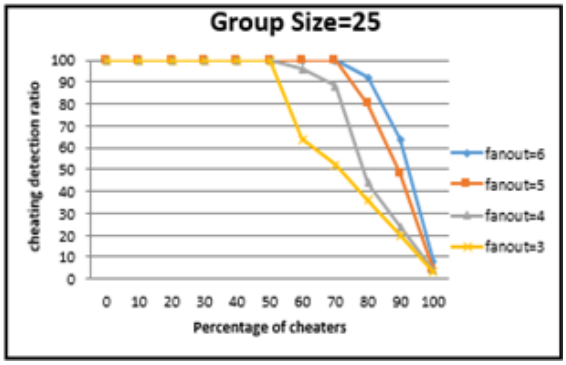



**Figure 4:** Cheating detection ratio in Scenario 2 for group size = 25 and different fanout.

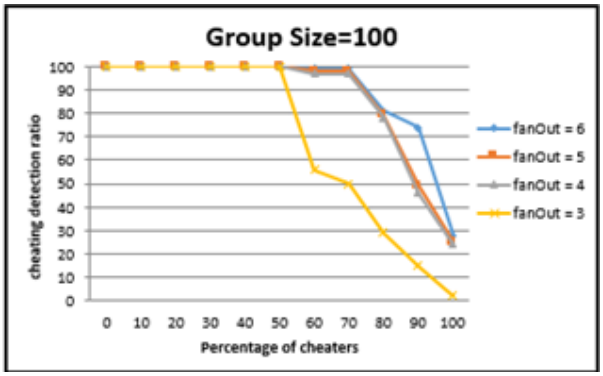



**Figure 5:** Cheating detection ratio in Scenario 2 for group size = 100 and different fanout.

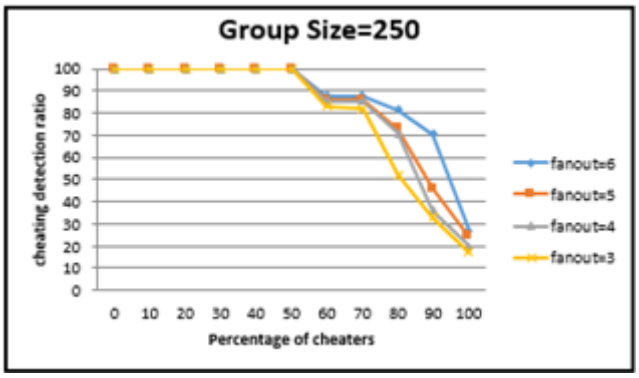



**Figure 6:** Cheating detection ratio in Scenario 2 for group size = 250 and different fanout.

The cheating detection ratio keeps increasing while the number of children increases, the cheating detection ratio reached about 90% when cheating ratio reaches 80% with fanout =6 and for group size =25 members. We should note that the previous results were the same regardless of group size.

**4.3.2 Cheating cancelation efficiency simulation results:**
In order to evaluate ofboth proposed cheating cancelation schemes efficiency, we consider the following case, we varied the group size and we do model for both proposed schemes and for study metrics, we run the simulation for varied values of group size and fanout while we are varying the percentage of cheating members from (10 to 100) %.

**(a) Scheme 1 results**
Figures (7-14) present simulation results for scheme 1. Simulation results show that the mean end-to-end delay for honest members decreased while the mean end-to-end delay for cheating members increased. This is mean that honest nodes get the benefit from applying the first proposal in contrast to the cheating nodes.

Also, we noticed that link stretch decreased until the cheating ratio reaches 50%, after that, it will increase and that's because cheating detection efficiency becomes lower while cheating detection ratio increased and all of this effect link stretch ratio.

We should mention that rearrangement overhead increased and that's ok because our proposal will disconnect the nodes which will try to rejoin again.

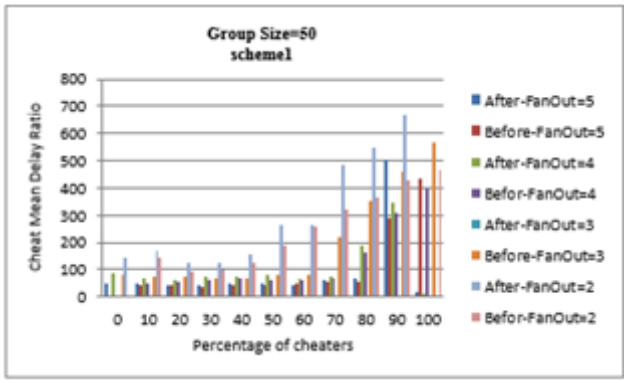



**Figure 7:** Mean End-to-End Delay Ratio for cheating members for group size =50 after applying scheme1

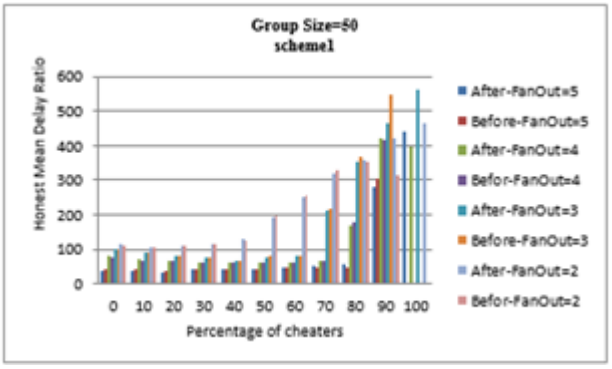



**Figure 8:** Mean End-to-End Delay Ratio for honest members for group size =50 after applying scheme1

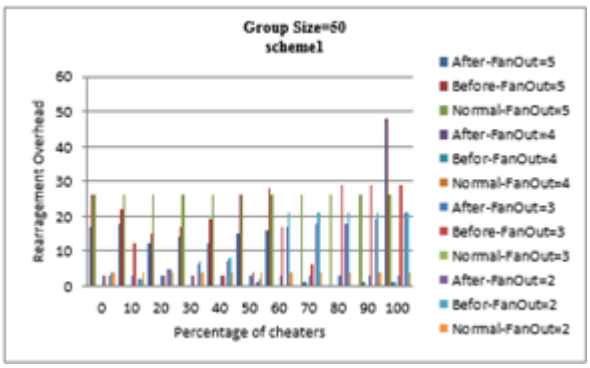



**Figure 9:** Rearrangement Overhead for group size =50 after applying scheme1

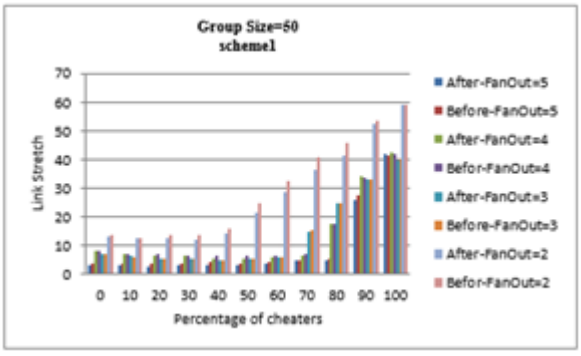



**Figure 10:** Link Stretch Ratio for group size =50 after applying scheme1

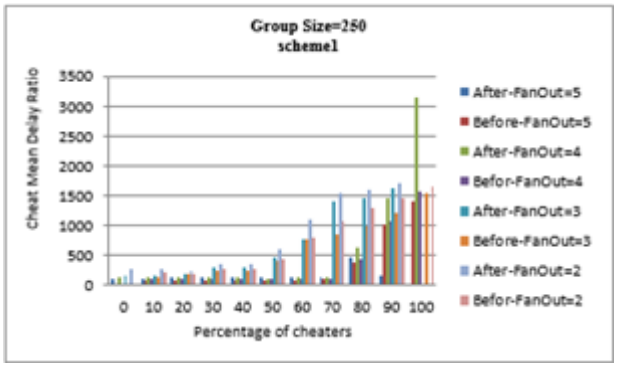



**Figure 11:** Mean End-to-End Delay Ratio for cheating members for group size =250 after applying scheme1

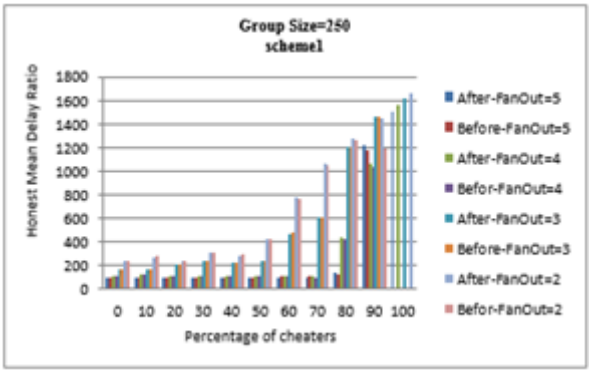



**Figure 12:** Mean End-to-End Delay Ratio for honest members for group size =250 after applying scheme1

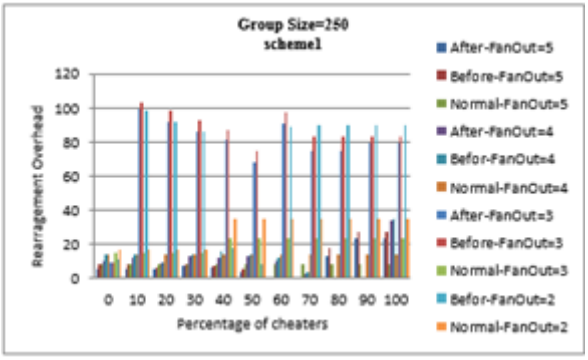



**Figure 13:** Rearrangement Overhead for group size =250after applying scheme1

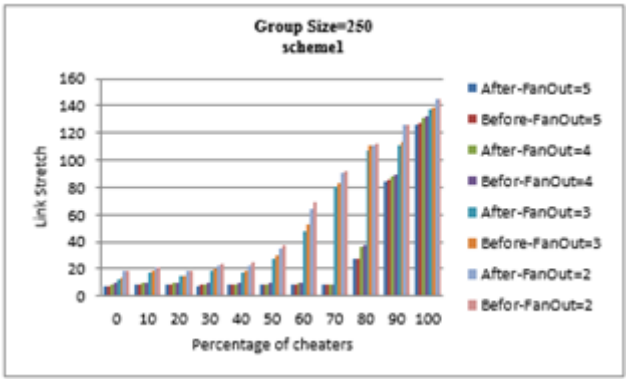



**Figure 14:** Link Stretch Ratio for group size =250 after applying scheme1

**(b) scheme 2 results**

Figures (15-22) present simulation results for scheme 2. We also found that simulation results for the second proposal are similar to the first proposal, but the difference between them is that link stretch ratio decreased without being affected by the cheating ratio. As for rearrangement overhead, we noticed that for the second proposal it was higher compared to the first proposal.

We should mention that both proposals demonstrated the same effect for different group sizes. Also, we found that the number of children (fanout) did not affect the performance of the two proposals and also did not affect all study metrics while varying group size between (50 to 250) members.

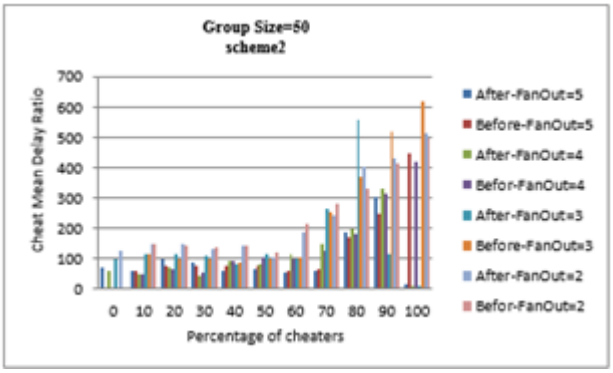



**Figure 15:** Mean End-to-End Delay Ratio for cheating members for group size =50 after applying scheme2

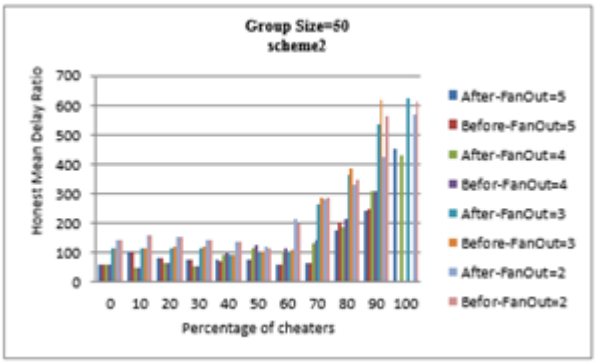



**Figure 16:** Mean End-to-End Delay Ratio for honest members for group size =50 after applying scheme2

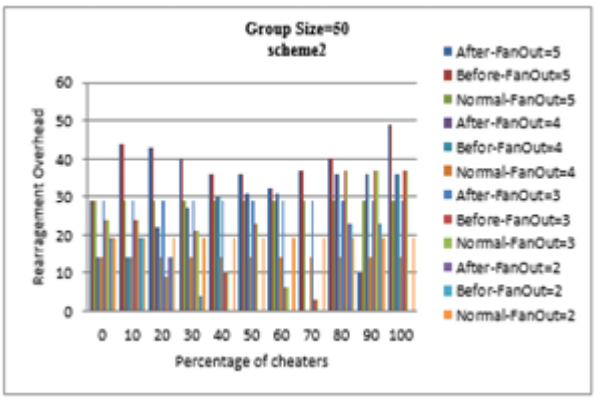



**Figure 17:** Rearrangement Overhead for group size =50 after applying scheme2

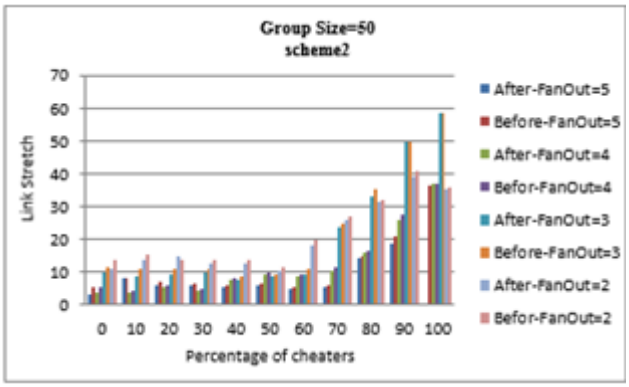



**Figure 18:** Link Stretch Ratio for group size =50 after applying scheme2.

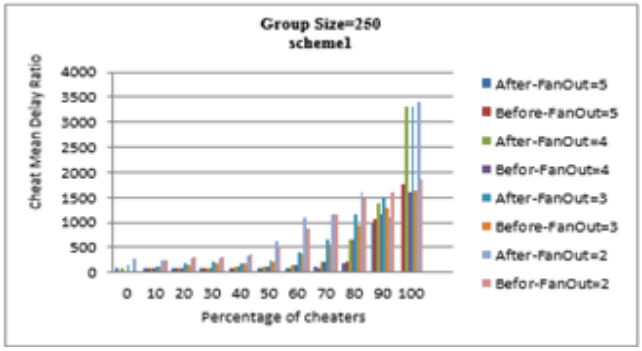



**Figure 19:** Mean End-to-End Delay Ratio for cheating members for group size =250 after applying scheme2

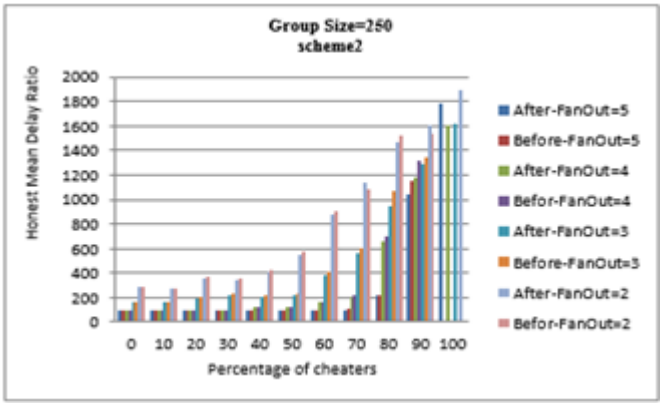



**Figure 20:** Mean End-to-End Delay Ratio for honest members for group size =250 after applying scheme2.

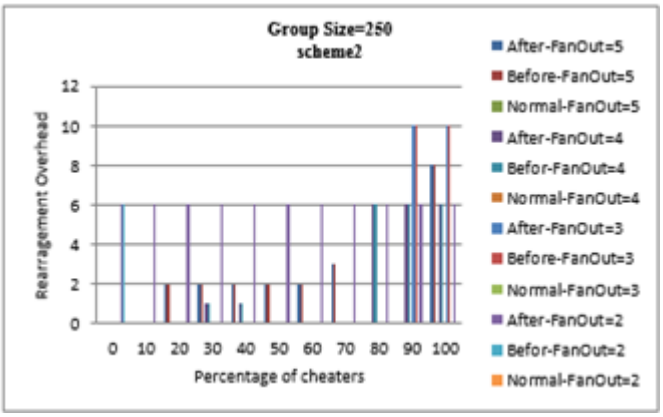



**Figure 21:** Rearrangement Overhead for group size =250 after applying scheme2

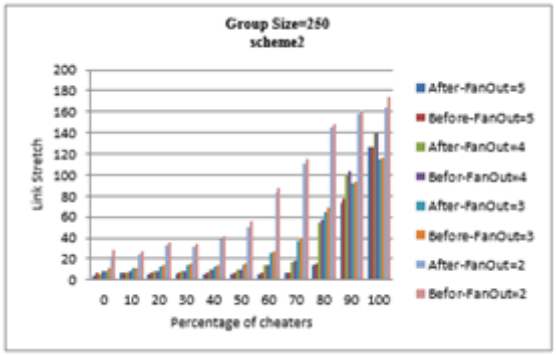



**Figure 22:** Link Stretch Ratio for group size =250 after applying schem2

## 5. Conclusion and Future Work

In this paper we applied a new scheme to detect cheating in (MDA-ALM) Protocol. We run the simulation on python simulator and we noticed that cheating detection scheme showed good results, it was able to detect cheating up to 100% until 50% cheating percentage ratio, but the detection ratio decreased while cheating percentage ratio keeps increasing. Moreover, it was able to detect cheating up to 100% until 80% cheating percentage ratio while the number of children keeps increasing. We should mention also that cheating detection algorithm has been equally effective on all group sizes.

Also, we proposed two cheating cancelation schemes, and we noticed that both proposed cheating cancelation schemes showed good results on the study metrics while varying group size and fanout also. The first proposal is preferable in terms of rearrangement overhead while the second proposal is preferable in terms of link stretch. Based on the result, we found that both proposed schemes achieved the improvement that required from it and it is recommended to use both cancelation schemes with all network sizes for different fanout values.

For future work, we plane to develop cheating detection method to be applied on other protocols.

## References

[1] Amad, Mourad, "Application Layer Multicast Based Services on Hierarchical Peer to Peer Architecture." Applied Mechanics and Materials, Trans Tech Publications, Ltd., vol. 892, pp. 64–71, June 2019.

[2] Christelle al hasrouty, Mohamed Lamali, Vincent Autefage, Cristian Olariu, , Damien Magoni and John Murphy, "Adaptive Multicast Streaming for Videoconferences on Software-Defined Networks", Computer Communications, Vol.132, pp: 42-55, 2018.

[3] Mojtaba Hosseini, Dewan Tanvir Ahmed, Shervin Shirmohammadi and Nicolas D. Georganas, "A survey of application-layer multicast protocols", IEEE Commun. Surv, vol. 9, pp: 58-74, 2007.

[4] Shubha Shukla and Akhilesh Kosta, "A Relevant and Survivable Scheme for Application Layer Multicast Routing", Department of Computer Science & Engineering K.I.T, Kanpur, India, vol.2, no.8, pp: 43-54, August 2013.

[5] Ayman El-Sayed. "Application-Level Multicast Transmission Techniques Over The Internet". PhD thesis, INRIA RhneAlpes, March (2004).

[6] Zhiye Huang, Jinxiang Peng and Jian Zhang, "The application-level Multicast Technique Algorithms Oriented to P2P video", Applied Mechanics and Material, vol.8, no. 303-306, pp: 2260-2264, January 2013.

[7] Krzysztof Stachowiak, Tytus Pawlak and MaceijPiechowiak, "Performance Evaluation of Multicast Overlay Routing Protocols", Image Processing & Communication, vol. 17, no. 1-2, pp: 19-32, January 2013.

[8] M. Alkubeily, H. Bettahar, A. Bouabdallah, "A new Application-Level Multicast technique for stable,

robust and efficient overlay tree construction", Computer Networks, vol.55, no.15, pp: 3332-3350, 2011
[9] D. Li, Yong Cui, Ke Xu, and Jianping Wu, "Impact of Receiver Cheating on the Stability of ALM Tree", In IEEE Global Telecommunications Conference, GLOBECOM '05, St Louis, Missouri, USA, vol.2, pp: 667–671, November- December 2005.
[10] Rossum, Guido Van (20 January 2009). "The History of Python: A Brief Timeline of Python". The History of Python. Retrieved 5 March 2021.
[11] Peterson, Benjamin (20 April 2020). "Python Insider: Python 2.7.18, the last release of Python 2". Python Insider. Retrieved 27 April 2020.
[12] TIOBE index (December 2020). "TIOBE Index for December 2020". TIOBE.com. Retrieved 20 December 2020.
[13] Stephen Cass, Nick Diakopoulos and Joshua J. Romero," The Top Programming Languages", IEEE Spectrum's 2014 Ranking, July 2014.
[14] Anitha, M and Yogesh, P, "A Contemporary Study of Application Layer Multicast Protocols in aid of Effective Communication", International Journal of Computer Science & Information Technology, Vol. 5, Issue 3, pp:3823, May 2014.
[15] John Buford, Mario Kolberg, Thomas C. Schmidt and Matthias Wählisch, "Application Layer Multicast Extensions to RELOAD", IRTF Internet Draft - work in progress, no.1, July 2010.
[16] M. Alkubeily, H. Bettahar and A. Bouabdallah, "Impact of Cheating and NonCooperation on the Stability and the Performances of Application Level Multicast Sessions", Proceedings of the 4th International Conference on Information Assurance and Security, Naples, pp: 141-146, September 2008.
[17] Ayman EL-SAYED, "End-System Multicast Protocols (ESM): Principles, Improving scalability/Robustness, Cheating, and Security Techniques", LAP Lambert Academic Publishing (2012-04-08), ISBN-13: 978-3-8484-9383-8, ISBN-10:3848493837, EAN: 9783848493838, pp: 172, 08 April 2012.
[18] https://networkx.github.io
[19] http://www.networkx.lanl.gov/wiki
[20] K. Almeroth, M. Ammar, "Collecting and modelling the join/leave behaviour of multicast group members in the MBone", in: 5th International Symposium on High Performance Distributed Computing (HPDC'96), Syracuse, NY, USA, pp: 209–216, August 1996.
[21] K. Almeroth, M. Ammar, "Multicast group behaviour in the internet's multicast backbone (Mbone)", IEEE communications Magazine, vol. 35, pp: 124–129, 1997.
[22] Tukey, John W.: Exploratory Data Analysis. Addison-Wesley Publishing Company Reading, Mass. — Menlo Park, Cal., London, Amsterdam, Don Mills, Ontario, Sydney 1977, XVI, 688 S, BIOMETRICAL JOURNAL, Vol. 23, Issue 4, 1981, Pages: 413–414, H. Beyer, Article first published online: 18 JAN 2007.

## Author Profile

**Rasha Shawish** is currently a Faculty member at American university of Middle East, Egaila, Kuwait, and she is Ph.D. candidate at Syrian Virtual University. She obtained here B.Sc. degree in communication engineering in 2011 from Tishreen University and also obtained her M.S. degree in communication Engineering from Tishreen University in 2015. Her research interest includes Application-Level Multicast networks, Key management, Security.

**Mothanna Alkubeily** is currently manager of National Center for the Distinguished, Latakia, Syria, he is also a Faculty member at Tishreen University, Latakia, Syria. He received the M.S. degree and PhD degree in Application-Level Multicast Protocols from the University of Technology of Compiegne (UTC, France) in 2006 and 2009 respectively. His research interest includes Wireless Sensor Networks, VANET and Application-Level Multicast.