# An Experimental Study and Analysis of Crowds based Anonymity

Lokesh Kumar Bhoobalan
Digital Worlds Institute
University of Florida
Gainesville, Florida 32611–5800
Email: lokesh@digitalworlds.ufl.edu

Piyush Harsh
Institute National de Recherche en
Informatique et en Automatique
INRIA Bretagne Atlantique Center
Rennes, France 35700

*Abstract*—**Crowds provides probable innocence in the face of large number of attackers. In this paper, we present the experimental results of the behavior of Crowds in a dense network. We begin by providing a brief description about Crowds followed by the experimental environment in which the simulations were carried out. We then present the results of our simulations and the inferences made out of them. We will also show that the obtained results match the predictions made by others.**

## I. INTRODUCTION

Anonymous communication involves communicating without revealing the identity to each other and to the outside world. There are three types of anonymity namely sender anonymity, receiver anonymity and the unlinkability of sender and receiver. Sender anonymity means that the information about the sender will be hidden while the receiver may not. Receiver anonymity on the other hand hides the information about the receiver. Unlinkability of sender and receiver refers to the phenomenon that, both the sender and the receiver may be found to involve in communication but cannot be identified as communicating with each other.

Many solutions exist to achieve anonymous communication over a network. They can be broadly classified under three heads:

1) Web Proxies
2) Mix based systems
3) Other Communication systems.

In a proxy based system, additional trusted third parties called proxy remain in between the sender and receiver. Requests and responses go through this proxy, by which the identities of the communicating parties are hidden. Some of the available proxies for anonymous web browsing include: Anonymizer [1], Proxify.com [2], and Proxy.org [3].

Mix based system was introduced by David Chaum in 1981 [4]. A mix in short is an enhanced proxy employing public key cryptography to achieve anonymity. It hides the sender's identity by cryptographically altering the messages being exchanged. Mixes utilize techniques such as buffering, and circulation of dummy traffic during idle time, in order to preclude an attacker from retrieving information about the nature and the parties involved in a communication.

Disclaimer: This work was done when Piyush Harsh was at University of Florida.

Other prominent communication systems include Onion Routing [5] and Crowds. In Onion Routing, the sender builds a virtual circuit by determining a path between it and the receiver using layered objects called "Onions". Every layer in the layered object contains information about the session key and next address of the node in the path. These onions that travel down the path are unwrappped using the session keys at each node. When the layers are fully removed, the session keys are destroyed.

The final system of interest and also has been the subject of analysis in this literature is the Crowd. They operate by forming a large group of users who may be geographically distributed. Crowds try to hide the actions of an individual with in that group by forwarding the requests randomly between the members before sending it to the final destination. The rest of the paper is organized as follows - section 2 provides a brief description about crowds, section 3 describes the relevant literature review with respect to crowds research, section 4 describes our simulation environment, section 5 presents the experimental results and the inferences drawn from the data, and then we end this paper with conclusion and future direction our research will take.

## II. CROWDS: A BRIEF DESCRIPTION

Crowds provide a mechanism for anonymous web browsing. Though there were other systems such as mix nets and DC-Nets [6] to accomplish the same, crowds were preferred because of the low latency and less computational overhead.

Every member of the crowd runs a process named jondos that registers itself with the central server called blender. Every jondos knows about every other jondos in this architecture. The blender is responsible for the distribution of symmetric keys between every pair (jondos). When a request for a web page begins at one of the nodes, the jondos running in the originator node forwards the request to one of the randomly chosen node by encrypting the message with the corresponding symmetric key. The latter node then either forward the same to another randomly selected node or to the web sever. The decision is taken based on forwarding probability with which it operates.

When a jondos receives a message, it does limited processing to preclude certain attacks and continues to transmit the message. The request and the response of the message follow
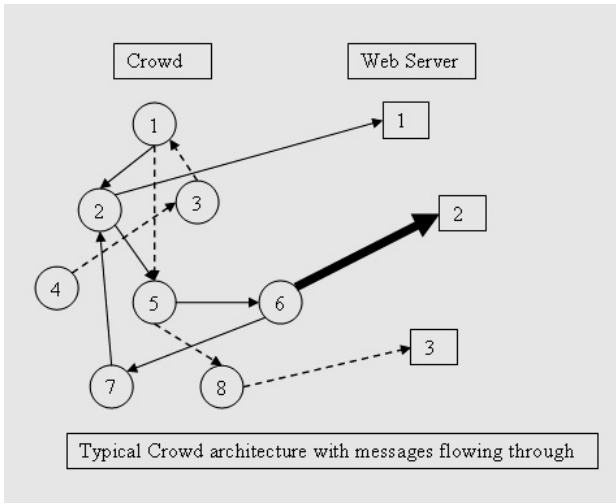
Fig. 1. A Typical Crowd Architecture

the same virtual path. These paths are torn down and new paths are constructed on the regular basis whenever there is a change of member count with in the crowd or there is a node failure. A jondos cannot decide by itself, if the request originated from the preceding node or the one before it.

In this architecture, the adversary can observe the server receiving messages. He/she cannot determine the source of the message. Similarly, when the server transmits the response back in the same path, it is difficult to ascertain the final receiver. In addition, even if the attacker finds that there are clients and servers communicating, he /she cannot find out as to which client talks to which server. Hence the anonymity goals that the crowds achieve include sender anonymity, receiver anonymity and as well as the unlinkability of sender and receiver. However all of these depend on the kind of attacker that we are talking about while determining the anonymity. For example, there is no sender anonymity against a local eaves dropper and, receiver anonymity against the end server. In addition, none of these anonymities work against a global eavesdropper if the scope of the crowd is only with in LAN. This in turn necessitates spanning the crowd across multiple administrative domains.

## III. LITERATURE REVIEW

In this section we will provide brief descriptions and summaries from literatures significant in the domain of crowds. Reiter and Rubin [7](Crowds: Anonymity for Web Transactions), were the first to introduce the concept of Crowds. They discussed the ways by which crowds can be formed and operated. Measuring anonymity provided by Crowds by employing the concept of degree of anonymity was also discussed. Finally, it was proved that the expected length of hop count for a message to reach the end server is $\frac{1}{1-P_f} + 1$, and the expected participant payload in a crowd of 'n' nodes is bounded by $O(\frac{1}{(1-P_f)^2} \times (1+\frac{1}{n}))$.Here $P_f$ is the forwarding probability. They also provided information regarding design,

implementation, security, performance and scalability of the system.

The bounds for the participant payload proposed by Reiter and Rubin was further improved in [8](The cost of becoming anonymous: on the participant payload in Crowds). This paper provides a precise formula that expected payload of a participant also tends to $\frac{1}{1-P_f} + 1$. In addition, the authors also showed that participant payload in Crowds is entirely independent of its size which in turn made evident that the Crowds possess good scalability feature.

In [9] Towards measuring anonymity by Claudia, Stephen, Joris and Bart, the author discusses about measuring the degree of anonymity of systems including Crowd through entropy $H(X)$. In this literature, $H(X)$ for Crowds is measured as,

$$\frac{N-p_f(N-C-1)}{N}log_2[\frac{N}{N-p_f(N-C-1)}] + p_f\frac{N-C-1}{N}log_2[\frac{N}{p_f}]$$

Here $N$, $p_f$ and $C$ are the total number of crowd members, probability of forwarding to the another member, total number of collaborators respectively. This measure is in addition to the suggestion made in [7] where the degree of anonymity is defined as $(1 - P_{sender})$ where $P_{sender}$ is the probability assigned by the attacker to a particular user.

Trust plays a major role in deciding your forwarder. Hence Vladimiro, Ehab and Sardaouna in their paper titled [10] Trust in Crowds: probabilistic behavior in anonymity protocols, proposes a Crowds-Trust protocol that uses trust information to achieve the desired level of anonymity. They also derive expressions for different level of anonymity required.

## IV. SIMULATION ENVIRONMENT

Network topology consisting of 2500 nodes was generated using Georgia Tech Network Topology Generator (GT-ITM) [11].The output of the same was converted to a understandable format using the utility sgb2alt that accompanies the software. The output comprised of source,destination node and the path length between them. The path length was interpreted as delay in the simulation. This was followed by computing the shortest path between all pairs of nodes using Floyd-Warshall's Algorithm [12]. This is how we generated the network topology:

```
# <method keyword> <number of graphs> [<initial seed>]
# <n> <scale> <edgemethod> <alpha> [<beta>] [<gamma>]
geo 3
2500 2500 3 .03
```

Included here is a portion of the output that was generated by the topology generator:

```
GRAPH (#nodes #edges id uu vv ww xx yy zz):
2500 188354 geo(0,{2500,2500,3,0.030,0.000,0.000}) 2500

VERTICES (index name u v w x y z):
0 0 805 682
1 1 1134 268
2 2 2181 925
3 3 1670 310
4 4 793 291
5 5 1747 917
6 6 220 945
7 7 183 1775
```

Crowd network was an overlay network on top of this generated network. Hence although there remain direct connections between two nodes in a Crowd network, the message passes through numerous other nodes in the underlying network before reaching the destination. The delay that was precomputed at the end of topology generation was taken as the delay between the nodes due to the underlying network configuration. The simulations were run for different node count and probabilities. Sampling of nodes that are part of Crowd network for every simulation were generated from the underlying topology using a randomized algorithm.

The simulator that we used, was written in Java and ran on Linux machines. The server (blender) was started and made to wait for a predefined period (specified in the configuration file) to accept connections from the clients (members interested in becoming part of the network). At the end of this registration phase, every member was provided information about itself and every other member. The number of client processes to spawn, depending on the node count, were performed using a batch file. These client processes were ran on different machines. The parameters for the client such as the probability with which to operate, time at which to generate and send messages and as well as the delay incurred in transmitting message to its successive member were specified in the client configuration file. Every scenario was run for 100 iterations and the data (hop count, total delay) collected.

## V. Experiments and Outcomes

The simulations were run for member counts ranging from 10 to 1000 with predefined intervals in between this range. The forwarding probability was allowed to vary from 0.1 to 0.95. Measurements were taken after running every scenario for 100 iterations.

The graphical outputs for some of the runs are presented below. The graphs are plotted between the following parameters:

a Count of nodes participating in Crowd and delay incurred in transferring message.

b Count of nodes participating in Crowd and the measured hop count.
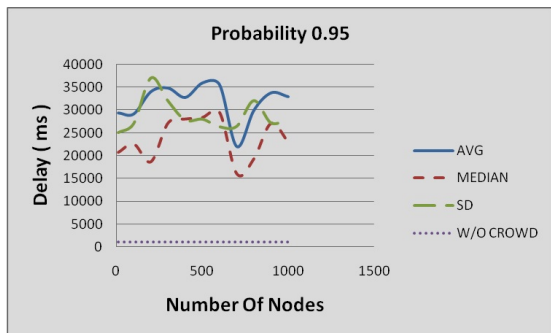


Fig. 2.    Transmission delay for probability 0.95
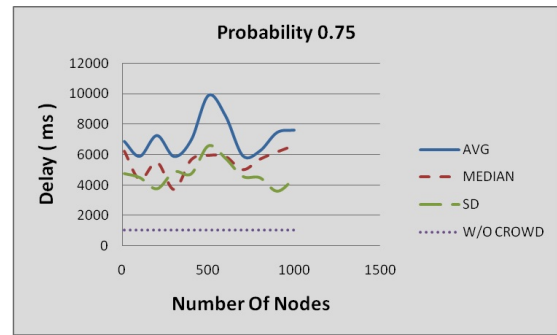


Fig. 3.    Transmission delay for probability 0.75
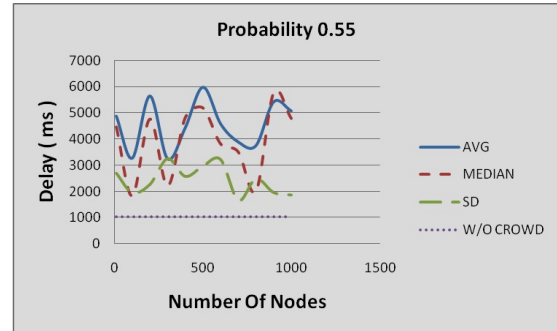


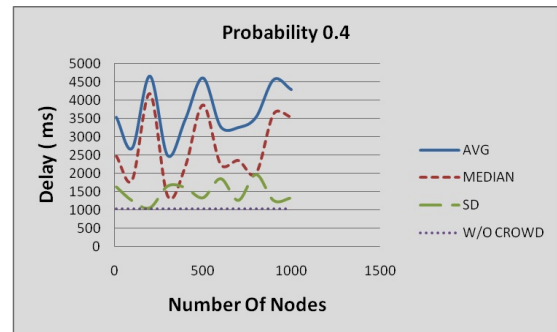Fig. 4.    Transmission delay for probability 0.55



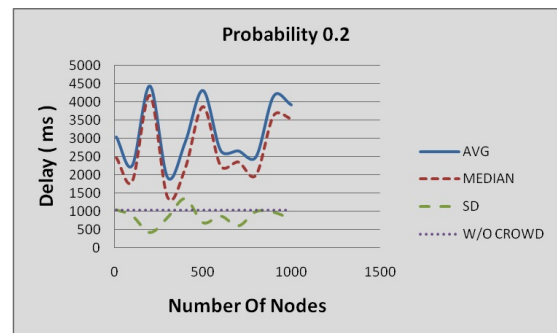Fig. 5.    Transmission delay for probability 0.40



Fig. 6.    Transmission delay for probability 0.20

Figures 2, 3, 4, 5, and 6 show the message transmission delays incurred in milliseconds between the source and destination when transmitted through the crowds network with

varying transmission probability and with crowds composed of different node counts.

The next set of figures shows the hop count a message has to travel before reaching the destination if sent through the crowds network for networks composed of different number of node counts and with crowd node varying forwarding probabilities.
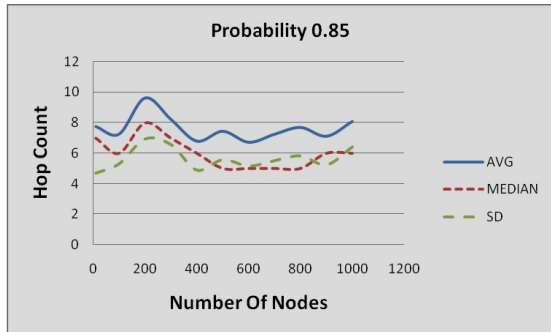
**Probability 0.85**

Fig. 7.    Number of hops traveled for probability 0.85

Figures 7, 8, 9, 10, and 11 shows the experimental results we collected for the parameters shown. Each experiment was repeated 100 times and the average, median, and the standard deviations have been plotted.

## VI. RESULTS INTERPRETATION

When the forwarding probability associated with the nodes increased, the average hop count that the message took to reach the destination increased and mostly followed the derived entity $\frac{1}{1-P_f} + 1$ except under very high probability. This is clearly evident from the fact that the lesser the likelihood of reaching the target, the more it takes to reach it. Also since the crowds neither generate cover traffic nor increase the work load of CPU by encrypting and decrypting the content, the very slow increase of hop count for huge increase in the member count is beneficial for community adopting this service.

Though the hop count is minimum,the delay is more since the hop count overlooks the underlying nodes in between the member nodes.As the result, this service cannot be adopted for systems that require faster response.The delay in the response may not be acceptable for interacting users even for queries such as the one that provides some location information.

The standard deviation of the hop count increased on increasing probability. The standard deviation of the hop count decreased on increasing the number of nodes for the same probability. The hop count increases gradually for every increase in probability except at above 0.9 where there is a huge increase in the hop count.

Crowds neither generate cover traffic nor increase the work load of CPU caused due to a series of encryptions and decryptions. The encryption and decryption performed using path key is very minimal in Crowds. Hence they are preferred over mix nets and onion routing specially in situations where security of the content is not of prime importance. From the simulations, it is observed that the hop count increases very
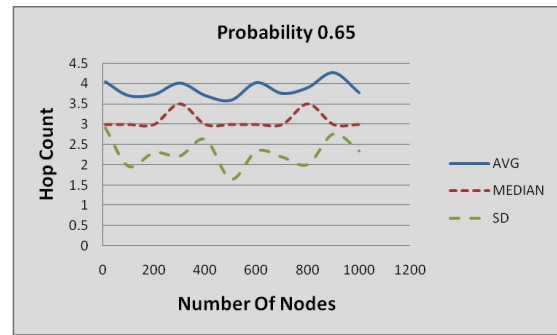
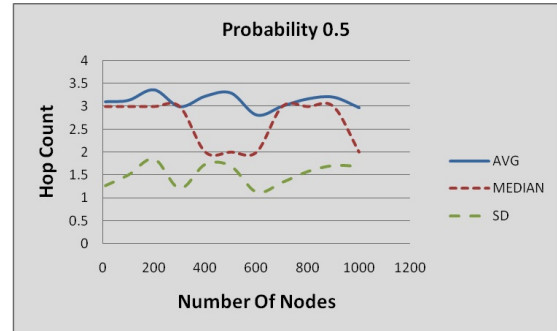**Probability 0.65**

Fig. 8.    Number of hops traveled for probability 0.65

**Probability 0.5**

Fig. 9.    Number of hops traveled for probability 0.50
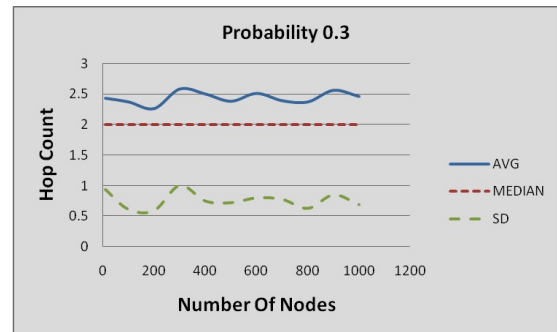
**Probability 0.3**

Fig. 10.    Number of hops traveled for probability 0.30
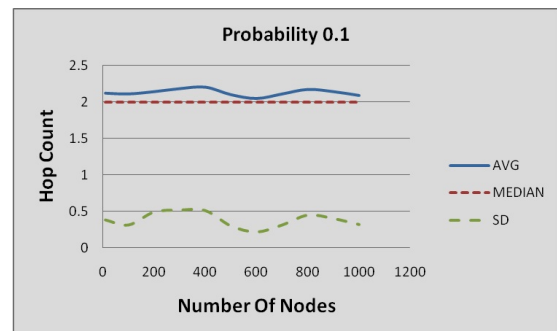
**Probability 0.1**

Fig. 11.    Number of hops traveled for probability 0.10

slowly and reaches the value 8 until the probability value hit 0.85, after which it increases drastically even for the large number of nodes. This seems to be reasonable in the light of

the service that it provides.

Although Crowd is meant to provide efficient service, the delay associated in sending a message to the destination is significant. This is evident from the graphs above where delay value is quite high even for minimum node count.

## VII. CONCLUSION

This paper provides the experimental results that we carried out to validate the operation of a crowd anonymity network. We have described our simulation startegy and provided the results we got. The results are in line with theoretical predictions made in several of the pioneering work in this field. In the very near future, we are planning on analyzing crowds and the degree of anonymity it provides by incorporating it within a utility function that would include the notion of cost versus degree of anonymity, transmisson detay and other relevant parameters. It is still a work in progress and would take some time before we can comment on the startegy here in this paper. We are also planning on using the same utility function to compare other anonymity schemes such as onion routing, mix nets, etc.

## ACKNOWLEDGMENT

## REFERENCES

[1] L. Cottrell. (2011) Your IP Address is Your ID. [Online]. Available: http://www.anonymizer.com/

[2] (2011) Proxify anonymous proxy protects your online privacy. [Online]. Available: http://proxify.com

[3] (2011). [Online]. Available: http://proxy.org

[4] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, pp. 84–90, February 1981. [Online]. Available: http://doi.acm.org/10.1145/358549.358563

[5] M. Reed, P. Syverson, and D. Goldschlag, "Anonymous connections and onion routing," *Selected Areas in Communications, IEEE Journal on*, vol. 16, no. 4, pp. 482 –494, May 1998.

[6] C. A. Melchor and Y. Deswarte, "From dc-nets to pmixes: Multiple variants for anonymous communications," in *Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 163–172. [Online]. Available: http://portal.acm.org/citation.cfm?id=1157739.1158219

[7] M. K. Reiter and A. D. Rubin, "Crowds: anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, pp. 66–92, November 1998. [Online]. Available: http://doi.acm.org/10.1145/290163.290168

[8] H. Sui, J. Wang, J. Chen, and S. Chen, "The cost of becoming anonymous: on the participant payload in crowds," *Inf. Process. Lett.*, vol. 90, pp. 81–86, April 2004. [Online]. Available: http://portal.acm.org/citation.cfm?id=989519.989524

[9] C. Díaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Proceedings of the 2nd international conference on Privacy enhancing technologies*, ser. PET'02. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 54–68. [Online]. Available: http://portal.acm.org/citation.cfm?id=1765299.1765304

[10] V. Sassone, E. ElSalamouny, and S. Hamadou, "Trust in crowds: probabilistic behaviour in anonymity protocols," in *Proceedings of the 5th international conference on Trustworthly global computing*, ser. TGC'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 88–102. [Online]. Available: http://portal.acm.org/citation.cfm?id=1893701.1893709

[11] E. W. Zegura. (2011) Georgia Tech Internetwork Topology Models (GT-ITM). [Online]. Available: http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html

[12] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, pp. 345–, June 1962. [Online]. Available: http://doi.acm.org/10.1145/367766.368168

**Lokesh Bhoobalan** received his M.S in Computer Science from University of Florida in April 2010. He is currently a research assistant at the Digital Worlds Institute, University of Florida. His research interests include distributed systems and their architectures, computer networks, and anonymity.



**Piyush Harsh** received his doctorate in Computer Engineering from University of Florida on August 2010. His current research interests are in Computer Overlay Networks, Autonomous and Cognitive Networks, Distributed Computing, and Cloud Computing Architectures. In the past he researched on issues in User Authentication mechanisms, Anonymity, Covert Channels and Cryptography. He is currently working as R&D Engineer at INRIA Bretagne-Atlantique Center on Federation of Clouds as part of European Union project funded under FP7 program. He is a professional member of ACM and in the past was student member of IEEE and ACM.