An Improved Strategy for Detection and Prevention IP Spoofing Attack

Huda Basim Said
University of Mosul
Mosul/Iraq

Turkan Ahmed Khaleel, PhD
University of Mosul
Mosul/Iraq

ABSTRACT

The Spoofing Attack is dangerous and complex to networks and clouds; an attacker fakes a legitimate user address and launches his attack. Those who control the cloud have a big role to play in preventing and detecting these attacks.

This research focuses on enhancing an algorithm called HCF (Hop Count Filtering Algorithm) helps to get rid of the weaknesses of this algorithm.

General Terms

Addresses to Hop Count (Addr2HC) table, Improved Hop Count Filtering (IHCF).

Keywords

Cloud Computing, IP Spoofing, Spoofing Attack, time to live, IP2HC table, Hop Count Filtering (HCF).

1. INTRODUCTION

Cloud computing is a service that provides resources shared by all customers (resources are increased or down to customer demand) and only the use of these resources is possible without attention to management, confidentiality and resource allocation [7] [14], and it is available over internet [9]. Cloud computing is described as a technology that encapsulates some of the relevant existing technologies such as network computing, computational computing, cluster computing and computing distribution to provide users with pooled resources as a service [12] [15].

IP Spoofing: is a major threat to network security in which the attacker impersonates legitimate agents or devices addresses or identity, and thus get all the properties submitted on behalf of the victim. That means; IP Spoofing is to create IP packets using someone else's IP address [5] [16].

The most effective used method for preventing and detecting IP Spoofing Attack is *Hop Count Filtering Method* bybuilding *IP2HC Table.* This Table suffers many vulnerabilities and holes; in this research, this research offers a satisfactory solution in order to use this table better, with a simple addition to this table.

2. BACKGROUND

2.1. The Difference between TTL Field and Hop Count

TTL (Time to Live) field is one of the IP header, a field determines the validity period of the sent packet [11], it is 8_bits space (in other word that 255 is the maximum value for this field [1] [2].

Hop Count is defined as the number of hops of a packet when the travel from the sender to the receiver, which means Hop Count depends on the TTL field of an incoming packet, it is the initial TTL subtracted by the number of intermediate hops [4].

The different Operating System use different initial TTL value, such as 30, 32, 60, 64, 128, and 255 [6] [8], it is the purpose of using the number of hops instead of using the *Time to Live* field value directly

2.2. Vulnerabilities in IP2HC Table

IP2HC Table is an effective way and more successive technique for guessing any duplication in source addresses; but there are holes in this method, as:

- The distance between the victim and the target must be different from the distance between the attacker and the target, in order to discover it.
- Operating systems do not use all TTL space (2^8) [17] [18].
- Initial ***Time to live*** value had known easily [20]; a hacker has another chance to complete its purpose.
- Building a table with “Multiple path possibility” is cancelled [10].

2.3. Literature Review

The most paper used the matching of IP address with TTL fields (from IP Header) for guessing Illegal user try entering the cloud system (or even the internet because The cloud is part of this environment):

- Researchers H. Wang and et al. used in their research [8] a table linking the source address with the number of packet’s hops travel until it reaches a detection node (using *Hop-Count Filtering method),* and 90% of spoofed IP packets have discovered.

- The DDoS mitigation method was provided by B. T. Swain et al. in [3] based on the value of the number of hops, and calculation of the number of malicious packets by using an improved probability policy and then filtering them. This method succeeded in preventing about 85% of the attacking packets, which reduced the computational memory and time during package processing.

- Shahid Akhter and et al. use the Hop-Count Defense Mechanism. By using static addresses to whole the network nodes, and assume 255 is the initial TTL field value [19].

- In 2017, P. Indu and et al. suggest an enhancement to HCF method by adding TCP port numbers, IP addresses and hop count values. Here has been well detected. But if the attacker is the server (same port number) and has the same hop counts with the victim then; the attack will be done [13].

3. PROPOSED METHODOLOGY

This section describes the flow of IHCF working strategy, and the brief description of data structure and functions used in this detection and prevention strategy.

3.1. Flow Diagram of IHCF Algorithm

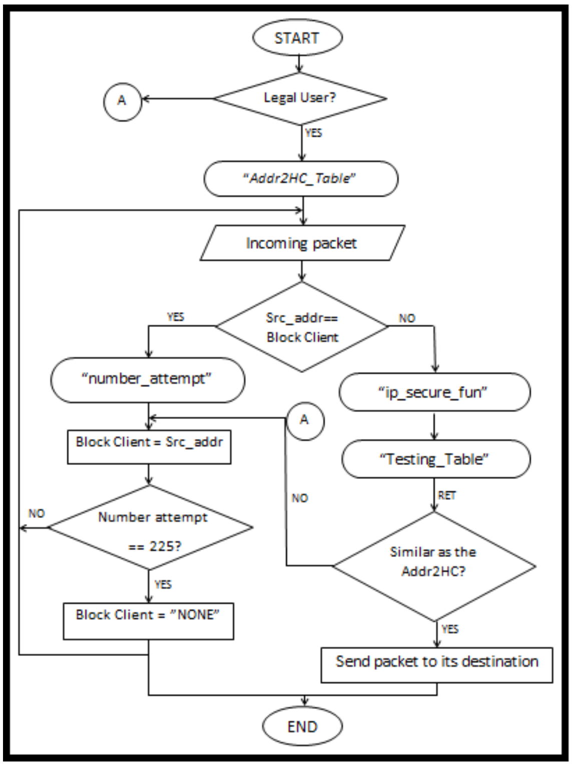
As explained in section 2.2 about the vulnerabilities of IP2HC table; here proposed a simple addition to this table.

An *Address to TTL Table* (Add2HC) is matching IP, Physical addresses, Hop Count value (HC), with TCP and UDP source port numbers. Show table 1.

	IP src address	**HC value**	**Src HW Address**	**TCP port**	**UDP port**
1	192.0.5.1	30	47:20:1B:2E:08:AE	--	25
2	192.0.6.1	2	11:15:1D:2E:0A:50	210	--
. . .	. . .	. . .	. . .	. . .	. . .
n	192.0.9.1	15	EE:C7:10:A0:2E:03	21	--

The matching of these objects may do a great role in detection and prevention phases and make a challenge at the hacker to fake all these IP address fields as in legal user packets.

The Flow Diagram of IHCF methodology appears in the figure 1 below.



As in figure 1: IHCF Algorithm needs one data structure and 3 functions; after testing the status of *legal user*.

Legal user is the identification number ***ID_Cloud*** for each client login in the cloud. Any other client has not ***ID_Cloud*** and will block.

Add2HC_Table: It seems as in table1. This table builds initially after executing the system, and updated when the routing protocol did. If the system is sensing an attack; no updating occurs.

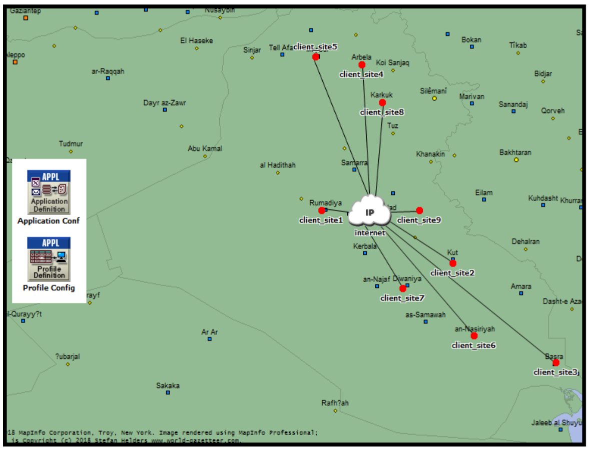
The ***'number_attempt'*** function examines if the source address of incoming packets is blocked in '*Block Client*', drop these packets directly. Else the function '***ip_secure_fun***' extracts addresses and hop count values from each incoming packet.

The '***Testing_Table***' the function tests the information obtained from the received packet with the table '***Add2HC'***.

Since the attacker uses a legitimate user address, at detection phase, both the legitimate user and the attacker are blocked. In order to restore the validity of the legitimate user after blocking it, IFHC calculates the number of times this address is attempted to contact the cloud, when the maximum number of attempts is reached, this blocked address will be released, however, all packets remain under the examination at all the time.

3.2.Experimental Setup

Here, you can show a proposed cloud network in Iraq, has 8 sites in different cities(have Web and File Clients), while the centralization in capital (Baghdad) which has three servers(file, Web, and Database serves as well as Database Clients which represent as private cloud). This implemented using OPNET Modeler 14.5A. The network shown in Figure2:



In the second scenario, Spoofing Attack station in the client_site8 will be added, which implements ping to all the servers in client_site9, and the victim station is client_site3_FTP Clients.

While in the third scenario, the detection and prevention mechanism depending on the *Add2HC* table and *IHCF* strategy will be applied in a cloud node (in child called *ip_rte_cloud* process model).

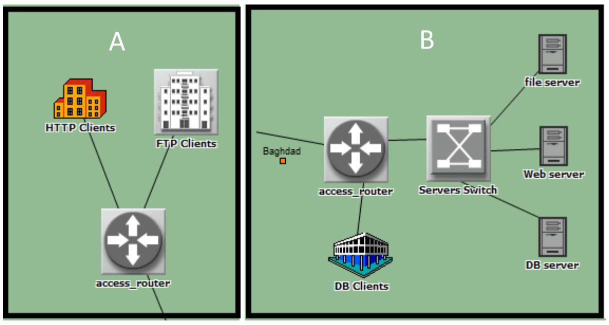


Fig. 3: The Client Sites.
A: All Client Sites1-8, B: Client Site9

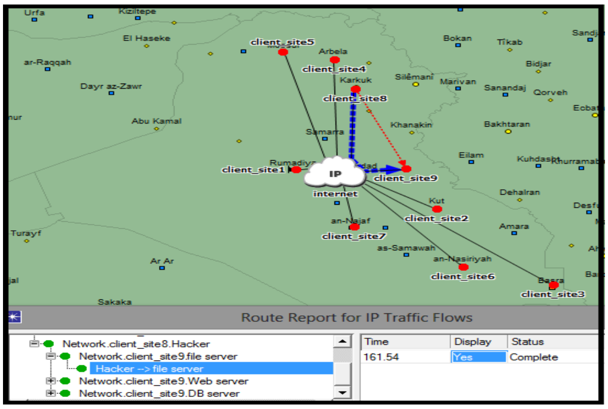
4. SIMULATIONS AND RESULTS

In this section, a network analysis is performed. There are three network models configured and run as follows: the first scenario without a spoofing attack, the second with a spoofing attack, and the third having spoofing attack and detection. The proposed topology is based on the use of RIP (Routing Information Protocol) as the routing protocol.

In two figures (figure2-3) clearly appear that all clients have the same distance from the cloud (two steps).Then blocking an attacker even though the legitimate client and the attacker have the same address and TTL value.

4.1. Hacker-Servers Demands

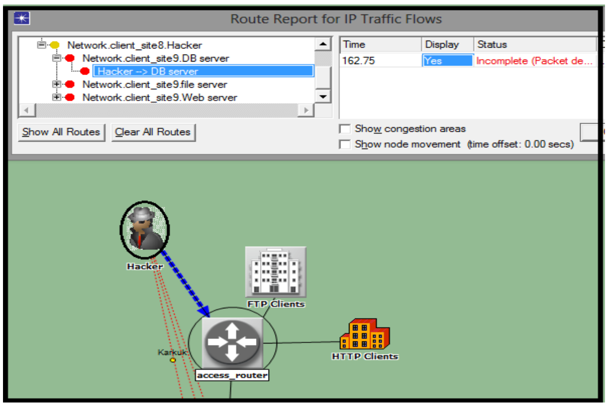
After running the program, figure 4-5 appear clearly the path (by using *ip_traffic_flow* demand) from Hacker to the server is active in attack scenario (figure 4), On the other hand, it breaks in the detection scenario (figure 5).



Route Report for IP Traffic Flows

Network.client_site8.Hacker
Network.client_site9.file server
Hacker --> file server
Network.client_site9.Web server
Network.client_site9.DB server

Time	Display	Status
161.54	Yes	Complete


Route Report for IP Traffic Flows

Network.client_site8.Hacker
Network.client_site9.DB server
Hacker --> DB server
Network.client_site9.file server
Network.client_site9.Web server

Time	Display	Status
162.75	Yes	Incomplete (Packet de...

Show All Routes   Clear All Routes

☐ Show congestion areas
☐ Show node movement (time offset: 0.00 secs)

Figure5 shows the demand is blocked at the router not at the cloud, because after assigning 'Block Client' in the cloud (as discussed in section 3.1), then any packets come from this client will be discarded directly (it seems as that no link between the router and cloud).

4.2. Aborted Connections

Measuring connection numbers that have been aborted, which means the total number of illegal communications have been blocked as well as dropped (in Figure 6, notice that all the neglected packets are at detection period).

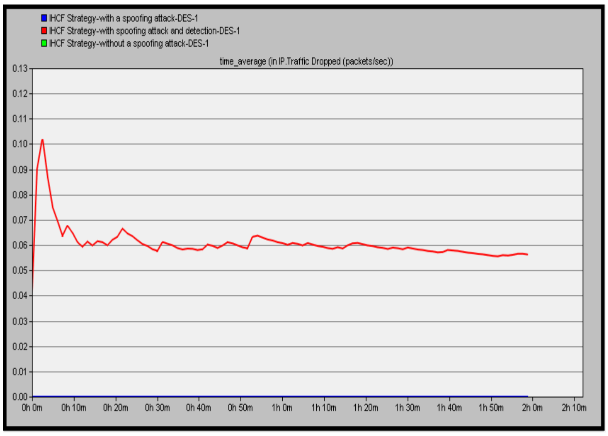


Fig. 6: Number of Aborted Connections.

4.3. DB Response Time

The time in the second scenario was greater because of the presence of the attacker, which increases the load on each network. Show figure 7.

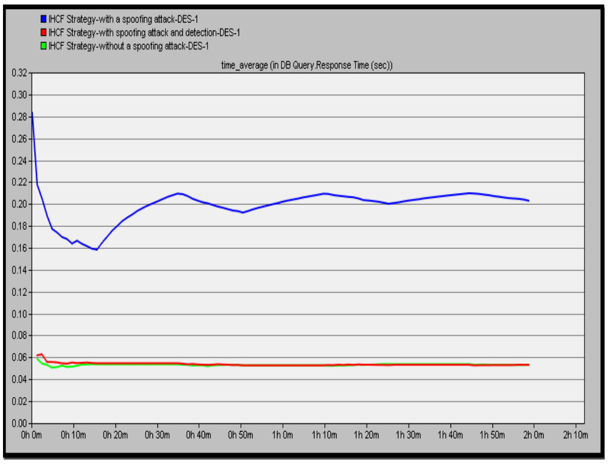


Fig. 7: DB Response Time.

4.4. Traffic Received and Sent of DB Query

Here notes that the most sent and received packets are in the second scenario (when an attacker is present). But at the detection phase, traffic returned to his normal situation as no attacker, as in figures 8-9.

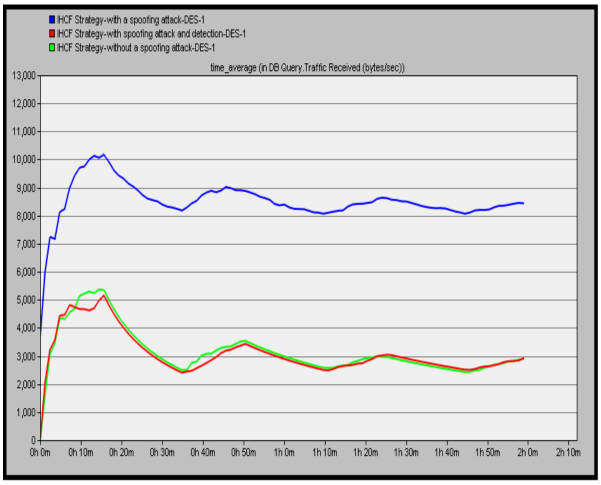


Fig. 8: Traffic Received of DB Query.

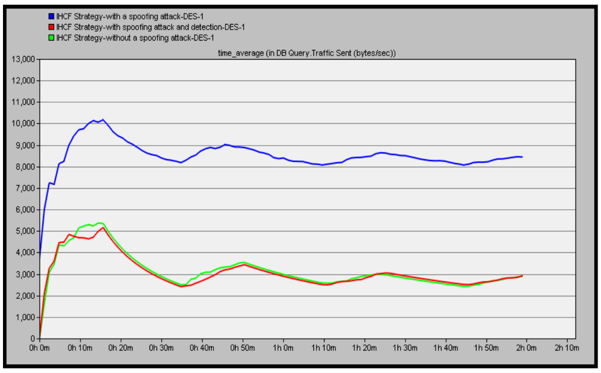


Fig. 9: Traffic Sent of DB Query.

5. CONCLUSION

This research is working on an enhancement of algorithm based on linking the number of hops with the source address called HCF, by building a table that links the source addresses (IP, Physical addresses and source TCP or UDP port for different applications) with the number of hops and saves that table in the cloud.

This table updates after the routing table is updated, and if there is no attack on the network.

After the implementation of this algorithm, the attacker was blocked and the network traffic returned closely as it was before the attack appears.

6. ACKNOWLEDGMENTS

Special thanks to *Dr. Faris Alghareb* (University of Central Florida) for his helpful, constructive comments and valuable tips.

7. REFERENCES

[1] Ballmann, B. 2012. “Understanding Network Hacks: Attack and Defense with Python”. Springer-Verlag Berlin Heidelberg, pp.10-11.

[2] Kumar, B. K., Kumar, P. K. and Sukanesh, R. 2010. “Hop Count Based Packet Processing Approach to Counter DDoS Attacks”. International Conference on Recent Trends in Information, Telecommunication and Computing. pp. 271-273.

[3] Swain, B. R. and Sahoo, B. 2009. “Mitigating DDoS attack and Saving Computational Time using a Probabilistic approach and HCF method”. IEEE International Advance Computing Conference (IACC). pp. 1170-1172.

[4] Jin, C; Wang, H. and Shin, K. G. 2003. “Hop-Count Filtering: An Effective Defense Against Spoofed Traffic”. 10th ACM Conference Computer and Comm. pp. 30-41.

[5] Duan, Z.; Yuan, X. and Chandrashekar, J., 2008. “Controlling IP Spoofing through Interdomain Packet Filters”. IEEE Transactions on Dependable and Secure Computing. 5 (1).

[6] Devi, G. U.; Priyan, M. K.; Balan, E. V.; Nath, C. G. and Chandrasekhar, M., 2015. "Detection of DDoS Attack using Optimized Hop Count Filtering Technique". Indian Journal of Science and Technology , 8(26) .

[7] Ramachandra, G.; Iftikhar, M. and Khan, F. A. 2017. “A Comprehensive Survey on Security in Cloud Computing”. Elsevier B.V. ,pp. 466.

[8] Wang, H.; Jin, C. and Shin, K. G., 2007. "Defense Against Spoofed IP Traffic Using Hop-Count Filtering". IEEE/ACM Transactions on Networking (ToN), 15 (1).

[9] Zahid, H.; Arshad, A.; Khalid, M.; Saeed, B. and Rafique, A., 2013. "Implementation of Cloud Computing over the MAN and Analysis of DoS Attack". International Journal of Science and Advanced Technology(IJSAT), 3 (10).

[10] Patil, S., 2.15. "Implementation Of Updated Hop Count Filtering Using Time to live Probing". International Journal of Advanced Research in Science Management and Technology (ijarsmt), 1 (1).

[11] Babatunde, O. and Al-Debagy, O., 2014. "A Comparative Review Of Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6)". International Journal of Computer Trends and Technology (IJCTT), 13 (1).

[12] Osanaiye, O..A. 2015. “Short Paper: IP Spoofing Detection for Preventing DDoS Attack in Cloud Computin”. 2015 18th International Conference on Intelligence in Next Generation Networks.

[13] Indu, P.; Joseph, S. E.; Sreelakshmi, M.C. and RemyaNair, T., 2017. "Enhancement of HOP Count Filtering Mechanism-An ANTI-IP Spoofing Technique". International Journal of Pure and Applied Mathematics, 114 (12).

[14] Patil, D.; Patil, P. and Patil, P., 2017. "Establishing Cloud Computing Security in trust-based Cloud Service Provider". International Journal of Engineering Technology Science and Research (IJETSR). 4 (4).

[15] Kumar, S. and Goudar, R. H., 2012. "Cloud Computing – Research Issues, Challenges,Architecture, Platforms and Applications: A Survey". International Journal of Future Computer and Communication. 4 (1).

[16] Rani, S.; Abhilasha, E. and Jindal, E.S., 2015. "Implementation and Analysis of Identity Spoofing Attack Using Epidemic Routing Protocol in DTN". International Journal of Current Engineering and Scientific Research (IJCESR). 2 (12).

[17] Zander, S.; Armitage, G. and Branch P. 2006. “Covert Channels in the IP Time To Live Field”. Australian Telecommunication Networks & Applications Conference (ATNAC).

[18] Zander, S.; Armitage, G. and Branch, P. 2007. “An Empirical Evaluation of IP Time To Live Covert Channels”. 15th IEEE International Conference on Networks.

[19] Akhter, S.; Myers, J.; Bowen, C.; Ferzetti, S.; Belko, P. and Hnatyshin, V. 2013. “Modeling DDoS Attacks with IP Spoofing and Hop-Count Defense Measure Using OPNET Modeler”. Technical Report. Rowan University at Department of Computer Science.

[20] Wang, X.; Li, M. and Li, M. 2009. “A scheme of distributed hop-count filtering of traffic”. IET International Communication Conference on Wireless Mobile and Computing (CCWMC 2009).