www.ijiarec.com

ISSN:2348-

Volume-5 Issue-1

International Journal of Intellectual Advancements and Research in Engineering Computations

An efficient technique for prevention of denial of service attack

[1]E.P.Prakash, [2]S.Anu Tharni,[3]A.Archana Devi,[4]R.Indhupriyaa,[5]J.Kiruthiga Muthulakshmi

[1]Assistant Professor, [2,3,4,5] UG Scholar

Department of CSE, SNSCE, Coimbatore -107, Tamil Nadu, India.

[1]prakash.ep@gmail.com, [2]methh.555@gmail.com,

[3]archanaarchu2710@gmail.com,[4]indhupriyaar@gmail.com,[5]kiruthiga3bala@gmail.com

ABSTRACT

Denial of service (DOS) and distributed denial of service (DDOS) have become major threats in a computer network. The function of a DDOS attack is fundamental to make the system not to respond any requests or providing services. Detection and prevention of DOS/DDOS is a tedious process. In the earlier days, it can be prevented by using the puzzle-solving technique. In puzzle-solving technique first, the server generates some puzzles and it sends to the client. Then the client solves the puzzle and it sends it to the server then the server confirms that the client able to obtain the service from the server. In this type of puzzle-solving technique attackers can merely reply to the server it cannot be identified in this puzzle-solving technique and Graphics Processing Unit (GPU)-inflation DoS doesn't work and it publish puzzle function in advance. It can be enhanced or overcome by using software puzzle-solving technique it dynamically generates puzzle function. This technique exploits the architectural difference between CPU and GPU. If an attacker tries to move a puzzle transition task into CPU either they want to translate into functionally equivalent GPU or it should do at dynamic puzzle generation. It is the time-consuming process to translate or rewriting a software puzzle. This software puzzle has some drawbacks it cannot carry out puzzle solving process on cloud environment and it doesn't construct client side software puzzle. In order to overcome this problem here, we proposed a new client puzzle technique which is based on two puzzles between clients. Moreover, in order to improve the performance of Denial of Service attack brute force attack, command evaluation, and Kullback libeller divergence are utilized.

A.INTRODUCTION

1a. DENIAL-OF-SERVICE (DOS) ATTACK:

In a denial-of-service (DoS) attack, an attacker attempts to prevent legitimate users from accessing any services and information. By attacking your computer and its network connection or sites you are making use of, an attacker may able to prevent you from accessing any of your email, websites, online accounts (banking, etc.), or other services that rely on the affected computer.

1a.1. TYPES OF DOS ATTACKS IN THE INTERNET

1.1.1. Network-Based Attacks

- TCP SYN Flooding
- ICMP Smurf Flooding
- UDP Flooding
- Intermittent Flooding

1b.DISTRIBUTED DENIAL-OF-SERVICE(DDOS) ATTACK:

In a typical DDoS attack, the assailant begins by exploiting the vulnerability in one computer system and making it the DDoS master. The attack master also called as the botmaster, it identifies and infects other vulnerable systems with malware. The assailant instructs the particular controlled machines to launch an attack against a specified target.

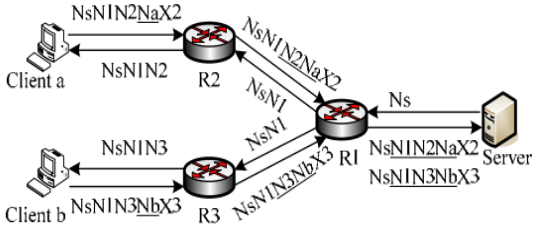
1b.1. TYPES OF DDOS:

There are two types of DDoS attacks: a network-centric attack which overloads a service by Increasing bandwidth and an application-layer attack which overloads a service or database with application calls.

1c. .DEFENCE USING PUZZLE

Client puzzles have been proposed to defend against DoS attacks in TCP, SYN, cookie authentication protocols, TLS, graphic Turing test, application traffic control, etc. In detail, when a client requests a service, the server first asks the client to solve a problem before accepting the service request. Then, the client needs to send an answer back to the server. Usually, solving the problem takes few amount of computing resource, while resource demand for verifying an answer is negligible. The server will discard its request for service either if the client does not reply or the answer is incorrect. In DPM, a typical puzzle is composed of a moderately-hard function (like hash function MD5); solving the puzzle requires a brute-force search in the solution space. The hash takes three parameters: a server nonce Ns created by the congested router, a

client nonce Nc created by the client, and a solution X computed by the client. The solution of the puzzle satisfies that the first d bits of h(Ns, Nc, X) are zeros. d is called the puzzle difficulty. DPM requests routers on the puzzle distribution paths to generate their own path nonce's and attach them to the congestion notification during the puzzle distribution phase. By using path nonce's, DPM gives different responders different puzzles (no once sequences), thus preventing the adversary from replaying the solutions via different paths. For example, R2 and R3 treat NsN1 as the server nonce and append their own nonces to the nonce sequence; R1 treats N2Na and N3Nb as the client nonces from R2 and R3. Once a link adjacent to a router is congested, the router requires the clients to solve the puzzle and thereby imposes a computational burden on clients who transmit via this router. The computation demand is accompanied to the bandwidth consumed by a puzzle-based rate-limiter implemented in the router.



1c.1 Defence Using Puzzle

B. LITERATURE SURVEY

1. R. Shankesi, O. Fatemieh, and C. A. Gunter, **"Resource inflation threats to denial of service countermeasures,"** Dept. Comput. Sci., UIUC, Champaign, IL, USA, Tech. Rep., Oct. 2010.

2. T. J. McNevin, J.-M. Park, and R. Marchany, **"pTCP: A client puzzle protocol for defending against resource exhaustion denial of service (DoS)attacks,"** Virginia Tech Univ., Dept. of Elect. Comput. Eng., Blacksburg, VA, Tech. Rep. TR-ECE-04-10, October. 2004.

3. E. Kaiser and W.-C. Feng, **"mod_kaPoW: Mitigating DoS with transparent proof-of-work,"** Proc. ACM Co-NEXT Conf., 2007.

4. R. L. Rivest, A. Shamir, and D. A. Wagner, **"Time-lock puzzles and timed-release crypto,"** Dept. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. MIT/LCS/TR-684, Feb. 1996.

C. METHODOLOGY AND SYSTEM IMPLEMENTATION

1a. Existing System

GPU is used for graphics processing and for general purpose computing. nVidia GPU is used which contains many streaming processors. Software puzzle is proposed to prevent GPU from being used to accelerate the puzzle-solving process which includes dynamically generated software. It doesn't reveal the puzzle function in advance like as data puzzle. Additionally, it used Kerckhoffs's Principle to construct software puzzle. In software puzzle framework it consists of code block warehouse which contains various software instructions and there are two modules one for puzzle generation and another one for hiding the puzzle for security purpose. Software puzzle aims to prevent GPU from being used in the puzzle-solving process.

1a.1. Disadvantages of existing system

- Cloud-inflated DOS attack fails
- Time consumption is high to generate puzzles on server side
- Code obfuscation may be not satisfactory in long-term software defense.

1b. Proposed System

In order to overcome the overhead problem of the server by introducing new client puzzle technique to prevent DOS attack. In this, every client in a network has to solve two puzzles to contact with another client. The first one is Discrete Logarithm Problem (DLP) which is solved by a client and produces a solution based on the trust level and the second puzzle is created based on the solution of DLP as well as a random number which is done by connection initiator. The solution of DLP puzzle can become more difficult or easy depends on the trust level of the client. Every client needs to follow the same procedure to find a malicious attack. If one client wants to communicate with other client senders the receiver generates a random prime number based on sender ID, a hello message, and timestamp. Then set a difficulty level of the puzzle to the sender. The receiver creates the strategy for the game and finds the optimum one. Then the receiver client verifies optimum strategy and set of strategies. In this approach, the server needs not verify the puzzle instead of that the clients in the communication verify the puzzle based on the strategy. If the strategy satisfies the client the communication can be granted over the several systems.

1b.1. Advantages of proposed system

- Time consumption is reduced
- Server overhead reduce
- Secure data transmission
- High efficiency.

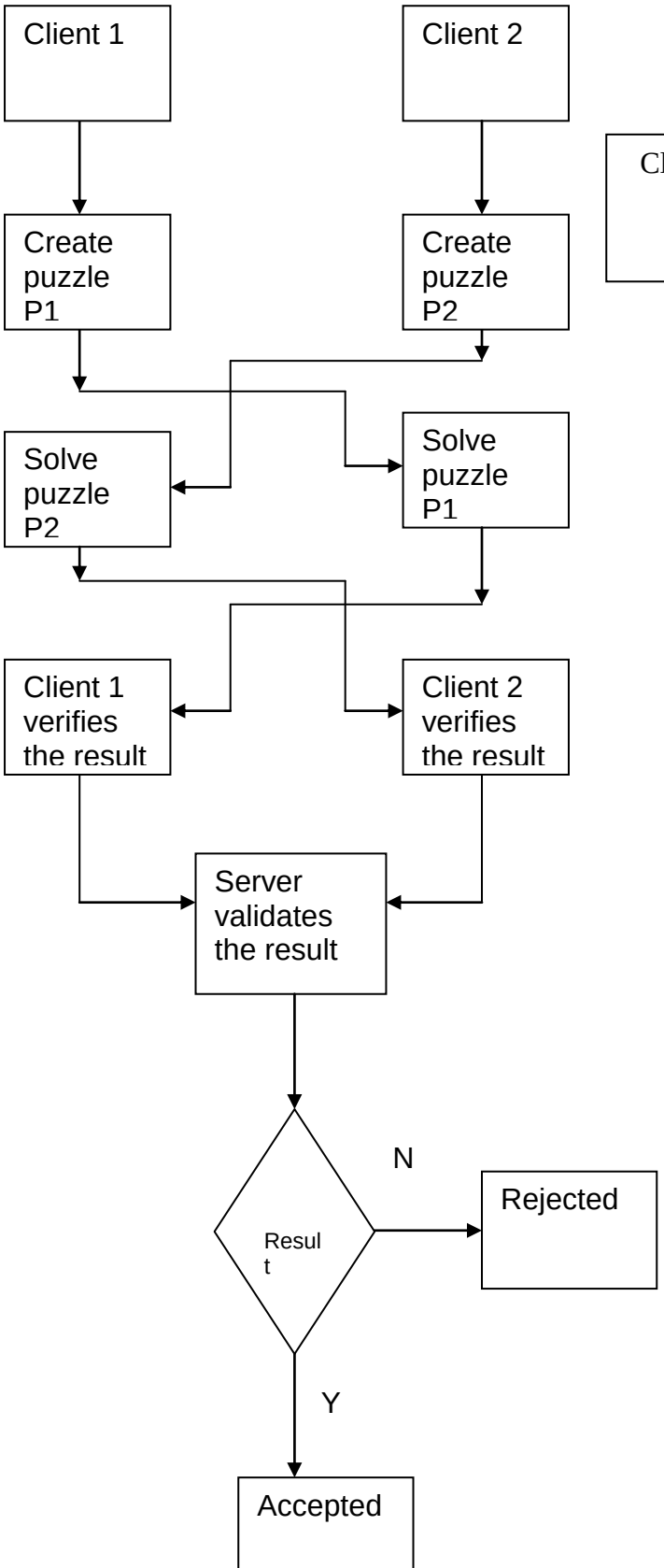
1c. Client puzzle technique for Denial of Service attack

Denial of service attack can be detected and prevented by creating and solving puzzles at the client

side. It reduces the overhead of server-side puzzle generation and verification. In this proposed technique server act as the middle man to exchange the information between clients. It creates a decentralized design where connection initiator in the client side is responsible for puzzle generation and puzzle solving as well as verification of puzzle is also carried over at the client side.

1d. METHODOLOGY:

1d.1ARCHITECTURE DIAGRAM:



1e. Puzzle construction and verification at client side

In this proposed technique the puzzles are created and handled at client side through game theory which is used for decision making. There are different and various numbers of players involved in game theory represent the number of clients and each and every player has set of strategies and a payoff for each combination strategies. It can be explained in figure

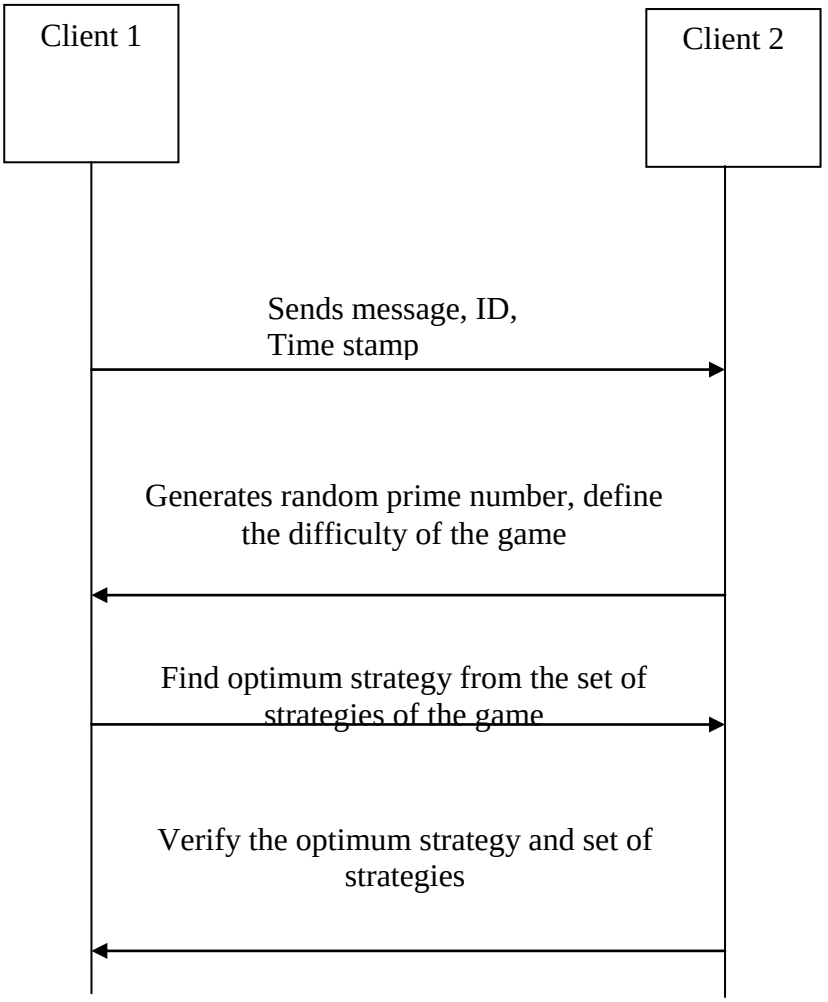


Fig 1e.1 Workflow of puzzle construction and verification

1f. Puzzle construction

Puzzle construction is the processes of creating puzzles at client side which are want to communicate each other. In figure 1e.1, shows the workflow of puzzle generation and puzzle verification. If client 1 wants to communicate with client 2, client 1 sends three parameters are hello message, its ID and time stamp. After receiving these parameters from client 1, client 2 creates random prime number R, random integer y and

a generator h of. Then client 2 set the difficulty level l of the puzzle that client 1 will have to solve the puzzle. In the next step client, 2 determine the value of S and utilize baby step giant step algorithm to find the value of h, over. S is calculated by using following equation:

$$S = h^{y} mod\ R \quad (1)$$

Algorithm 1:

Baby step giant step algorithm

Input: A cyclic group h of order m having a generator Y and an element β.

Output: value satisfying $Y^{x} = \beta$

1. n= ceiling ($\sqrt{m}$)
2. i=0 to n
3. for all i
4. calculate Y^{i} and store the pair (i, Y^{i}) in a teble
5. End for
6. Calculate Y^{-n}
7. μ= β
8. for j=0 to n-1
9. Check if μ is the second component (Y^{i}) of any pair in the table.
10. If $Y^{i}$is the second component
11. return jn + i.
12. else μ = μ • Y^{-n}
13. end if
14. end for

From the algorithm the value of Y_h^* was found which is simple modification of trial multiplication for computing discrete logarithm. The value of Y^{i} is calculated and store it in a table and verify μ is the second component return the value jn +i otherwise return μ • Y^{-n}.

The calculated values of S, h, R and Y_h^* send to client 1then it calculates number z that is needed to create the strategies of the puzzle by the integer division of random number and difficulty level. Then the client 1 generates and solves the puzzle after generating z and finds the optimum strategy by calculating all strategies. The strategies of each client are described as the addition of three positive integers (z1, z2, z3) which are in non descending order. It can be given by,

z1+z2+z3=z (2)

Each puzzle will have the number of strategies based on the client. The number of strategies in each puzzle is calculated by using equation 3. For all Z ϵ the number of strategies is:

Π(z)=1/2×[z/3]×z+1/8×〖(-1)〗^(z-3[z/3)+1/8×〖(-1)〗^(1+z)-3/4×[z/3]^2 (3)

The client 1 found all possible strategies of the puzzle through the equation 3 and it can able to find the optimal strategy from the set of strategies. It can be found by client 1 with the help of utility function where every strategy of each puzzle plays against all other strategies. The input of utility function is two different strategies where the client 1 compares each strategy by itself. The utility function is performed until all possible combinations have been played. The two different strategies are represented by M= (z1, z2, z3), (,) and the possible results of utility function are M, DRAW, and. It returns M as the optimum strategy if strategy M has at least two higher coordinates than the corresponding ones of strategy and it returns DRAW when the two input strategies have one equal and two different coordinates. It returns only if has at least two higher coordinates than the corresponding ones of strategy M. Thus the client 1 finds the optimum strategy and sends it back to client 2 along with the set of strategies, random number y, and its ID. Like as the client 2 generate puzzle and set of strategies and found the optimum strategy and it sends back to client 1 to verify the puzzle.

1g. Puzzle verification

In the puzzle verification process the created puzzle, set of strategies and the optimum strategies are verified by another client. If the verification of puzzle is validated then the information is sent through the server otherwise it won't send the information to requested client i.e. it found attacker in the network by the verification process. Client 2 verifies that equation (3) holds for all possible strategies and used an algorithm 2 to verify client 1 has correctly found set of strategies and optimum strategy.

Algorithm 2:

Verifying puzzle at client side

Input: random integer, set of strategies, optimum strategy OPT, ID, X(set of strategies, OPT)

Output: connect the client or disconnect the client

1. if(OPT[0]>$X_i[0]$and (OPT[1]>$X_i[1]$ or OPT[2]>$X_i[2]$))or (OPT[1]>$X_i[1]$and (OPT[0]>$X_i[0]$ or OPT[2]>$X_i[2]$))
2. OPTwins ++
3. end if
4. if $X_{i-1}[1]$< $X_i[1]$
5. sum+=$X_i[0]$
6. end if
7. else
8. exsum+=$(X_i[1] - j).X_i[0]$
9. if exsum==sum & OPTWINS≥ $\frac{\Pi(Z)}{2}$
10. give resources
11. end if
12. else
13. drop connection
14. end else
15. end else
16. end begin

From the algorithm 2 the created puzzles are verified at the end of both clients if it satisfies the condition the connection will be granted between the two clients otherwise the connection will be dropped by the server.

D.FEATURES:

1a. Brute Force Attack

Brute force attacks work by calculating every possible combination that could make up a password and test it to see if it is the correct password. As the password's length increases, to find the correct password the amount of time on average increases exponentially. This means short passwords can usually be discovered quite quickly, but longer passwords may take decades. This can be eliminated by using Renew after Fork SSP (RAF SSP) technique.

The main properties of RAF SSP are:

1. It can be used just pre-loading shared library. The source code need not be modified to the networking servers nor recompile the server nor modify neither system libraries nor the compiler.
2. It prevents brute force attacks against canary stack protection mechanism.
3. The SSP byte-for-byte attack is no longer applicable to the RAF SSP
4. The overhead introduces is negligible.

1b. Command Evaluation

Within the Command Evaluation, first, the Command Identification analyzes the statistical data of encrypted connections to identify which commands are entered by a user. For this purpose with the use of clusters, encrypted network packets are categorized where a cluster consists of the encrypted packets of one command and the corresponding response of the server. The beginning and end of each cluster are identified by the transport direction, the timings, and the memory of the payloads. Within the next step, the similarity of these clusters to the corresponding probabilities is saved for each cluster and different commands (generated in advance) is calculated. For our work, attack trees are designed and used to identify the probability that a sequence of commands is used for executing an attack. A part of an attack tree describing possible actions for a privilege escalation. Three different approaches are analyzed: the use of exploits, searching for configuration errors, and brute-force related attacks. Every branch of the tree can then be split into further paths, where every path can consist of several commands which can be used to execute a potential malicious action. The concept of attack trees is implemented in conjunction with the knowledge about commands used during an attack to constructing prototypical sessions. It also requires attack tree to evaluate if a sequence of commands can be used for an attack. This is evaluated by the Sequence Evaluation. If the commands are able to accomplish a path in an attack tree a (possible) malicious behaviour will be identified and an alarm can be raised.

To identify the user of a connection, similarity measurements are used again to detect the degree of correlation of the data stream with profiles of employees. Therefore we have integrated a module User Identification in our architecture. The Keystroke module receives the encrypted packets timings of the keystrokes connection.

List of Executed Commands by respective Attackers

If the following commands are used more than the preferred number of times then those nodes are identified as attackers

Ranking	Command	Times executed
1	At	103
2	Reg	31
3	Wmic	24
4	Wusa	7
5	netsh advfirewall	4
6	Sc	4
7	rundll32	2

Ranking	Command	Times executed	Option
1	Dir	903	
2	net view	225	
3	Ping	197	
4	net use	192	
5	Type	116	
6	net user	73	

Ranking	Command	Times executed	Option
7	net localgroup	34	
8	net group	19	
9	net config	16	
10	net share	10	
11	dsquery	6	
12	csvde	5	/f /q
13	nbtstat	5	-a
14	net session	3	
15	Nltest	3	/dclist
16	Wevtutil	2	

1c. Kullback libeller divergence

We use information based metric Kullback libeller divergence or skew divergence for detecting the attacks. The probability-based measures are information theoretic measures which are applied to any type of data type. Information distance is a measure of the difference between different probability distributions. However, the K-L divergence does not give the probability of similarity between the two distributions but it gives a relative value. So, the value of K-L divergence indicates an attack to be defined. Kullback–Libeller distance metrics for the intrusion detection which are in the terms of early detection, lower false positive rates, and stabilities.

We can also define the Kullback–Libeller distance as,

(P, Q) = (P||Q) + (Q||P)

(P, Q) = (log ())

Since and land, this means that P and Q have to be continuous absolute probability distributions. Here denoted as another client flow. The features of clients are extracted from the network monitoring tool.

E.CONCLUSION AND FUTUREWORK

This paper presented an efficient technique called client puzzle which is used to detect and prevent the malicious threat denial of service attack. This also reduces the overhead occurred at server side by creating and verifying puzzles at the client side. The server act as middle man between clients to transmits the information. Based on the verification results of client puzzles the resources are allocated to the client otherwise it will drop the connection between clients. The experimental results prove that the proposed client puzzle technique works more efficiently than the existing technique in terms of time and the number of served clients. The paper can be extended on GPU-inflation attack; its idea can be implemented to thwart DoS attackers which will exploit other inflation resources such as Cloud Computing. For example, suppose the server inserts some anti-debugging codes for detecting Cloud platform into software puzzle when the puzzle is running, the software puzzle will reject to carry on the puzzle-solving processing on Cloud environment such that the Cloud-inflated DoS attack fails.

F.REFERENCES:

[1]. Jason Ansel, Petr Marchenko," Language-Independent andboxing ofJust-In-Time Compilation and Self-Modifying Code", PLDI'11, June 4–8, 2011, San Jose, California, USA.

[2]. J. Black and P. Rogaway, "Ciphers with arbitrary finite domains," inTopics in Cryptology (Lecture Notes in Computer Science), vol. 2271. Berlin,Germany: Springer-Verlag, 2002.

[3]. X. Wang and M. K. Reiter, "Mitigating bandwidth-exhaustion attacks using congestion puzzles," in Proc. 11th ACM Conf. Comput. Commun.Secur., 2004.

[4]. J. Green, J. Juen, O. Fatemieh, R. Shankesi, D. Jin, and C. A. Gunter,"Reconstructing Hash Reversal based Proof of Work Schemes," in Proc. 4thUSENIX Workshop Large-Scale Exploits Emergent Threats, 2011.

[5]. R. Shankesi, O. Fatemieh, and C. A. Gunter, "Resource inflation threatsto denial of service countermeasures," Dept. Comput. Sci., UIUC, Champaign,IL, USA, Tech. Rep., Oct. 2010.

[6]. T. J. McNevin, J.-M. Park, and R. Marchany, "pTCP: A client puzzle protocol for defending against resource exhaustion denial of service attacks,"Virginia Tech Univ., Dept. Elect. Comput. Eng., Blacksburg, VA, USA, Tech.Rep. TR-ECE- 04-10, Oct. 2004.

[7]. E. Kaiser and W.-C. Feng, "mod_kaPoW: Mitigating DoS with transparent proof-of- work," in Proc. ACM CoNEXT Conf., 2007.

[8]. R. L. Rivest, A. Shamir, and D. A. Wagner, “Time-lock puzzles and timed-release crypto,” Dept. Comput. Sci., Massachusetts Inst. Technol., Cambridge,MA, USA, Tech. Rep. MIT/LCS/TR-684, Feb. 1996

[9] N.Shyamambika, N.Thillaiarasu “A Survey on Acquiring Integrity of Shared Data with Effective user Termination in the Cloud”International Conference on Intelligent Systems and Control (ISCO16), DOI: 10.1109/ISCO.2016.7726893, IEEE Explore, 2016.

[10] N.Thillaiarasu, Dr.S.Chenthur Pandian “Enforcing Security and Privacy over Multi– loud Framework Using Assessment Techniques”, International Conference on Intelligent Systems and Control (ISCO16), DOI: 10.1109/ISCO.2016.7727001, IEEE Explore, 03 N2016.

[11] N. Shyamambika and N. Thillaiarasu,"Attaining Integrity, Secured Data Sharing and Removal of Misbehaving Client in the Public Cloud using an External Agent and Secure Encryption Technique" Advances in Natural and Applied Sciences. 10(9); Pages: 421-431.