COMPUTATIONAL PUZZLES FOR REPUDIATION OF MISBEHAVING USERS IN ANONYM ZING NETWORK

T. Shanmugapriya, C. Magesh Kumar, S. P. Kavya, M. Nandhini
Department of Information Technology,
SNS College of Technology, Coimbatote, Tamilnadu, India

ABSTRACT

Anonymizing network provides web services to the users and also hide the client's IP address from the server. All data is wrapped with several layer of encryption. The success of this network, hackers can easily deface popular web site. If the users misbehave, blocking particular IP addresses is difficult. Nymble system is a credential system in which servers can blacklist misbehaving users in anonymizing networks, without compromising their anonymity. This system could block the IP address. However it is possible for a user to attack from another IP. To limit the number of credentials obtained by a single individual by raising the cost of acquiring credentials, we include client puzzles as a resource for obtaining credential where users are required to perform a certain amount of computation. This paper utilizes game theory to propose a number of puzzle-based defenses against flooding attacks.

KEYWORDS- *Game Theory, Nymble, Privacy.*

I. INTRODUCTION

Anonymzing networks route the message traffic through a serious of separate routers administrated by a separate domain. Here the user's identity and his activities cannot be monitored by anyone. By using this advantage the hackers can easily deface any popular website. There are several solutions to this problem, each providing some degree of accountability. The solutions are Group signatures, subjective blacklisting. Subjective blacklisting is also better suited to servers such as Wikipedia, where misbehaviors such as questionable edits to a Webpage, are hard to define in mathematical terms. Nymble system is used to blacklist the misbehaving users. This system does not protect the server from the DOS attacks. A DoS attack is characterized by a malicious behavior, which prevents the legitimate users of a network service from using that service. The resources exhausted by a flooding attack revive when the attack flood stops. A logic attack such as Ping-of-Death or Teardrop forges a fatal message accepted and processed by the victim's vulnerable software and leads to resource exhaustion at the victim. Unlike flooding attacks, the effects of a logic attack remain after the attack until some appropriate remedial actions are adopted. A logic attack can be thwarted by examining the contents of messages received and discarding the unhealthy ones. This is due to the fact that an attack message differs from a legitimate one in contents. In flooding attacks, on the contrary, such a distinction is not possible. This causes defense against flooding attacks to be an arduous task.

II. OUR SOLUTION

2.1 Puzzle Approach

Client puzzle is used to solve the flooding attacks. It can be effectively studied through game theory. This is mainly owing to the several trade-offs existing in a flooding attack- defense scenario. For an attacker, there is a trade-off between the severity of his attack and the amount of resources he uses to do so; the more damage an attacker intends to cause, the more amounts of resources he should spend.

We will assume that all client machines have the same processing power to devote to puzzle solving and we view an attacker as a compromised client machine. The puzzle difficulty (determined by the range of possible puzzle solutions) will be set low enough that every client machine will be guaranteed to solve at least one puzzle. Clients must present the solution to the puzzle along with a previously issued cookie that the server attached with the puzzle. To verify correctness, the server pulls out the server timestamp, indexes into the server nonce table to obtain the corresponding nonce, checks the expiry time, performs a hash of the client's answer with the nonce, and compares it against the cookie.

2.2 Defense Strategies

This section employs the solution concepts of infinitely repeated games with discounting to design the optimum puzzle-based defense strategies against flooding attacks. In general, the strategies prescribed by such solutions are divided into two categories: history independent (open loop) and history dependent (closed loop).
The concept of Nash equilibrium is often used in a descriptive way, where it describes the players' optimum strategies in a game. In this sense, it makes predictions about the behaviors of rational players. In this section, on the contrary, the concept of Nash equilibrium is employed in a prescriptive way in which the defender picks out a specific Nash equilibrium and takes his part in that profile.
The attacker may know this, but the best thing for him to do is to be in conformity with the selected equilibrium. If he chooses another strategy, he gains less profit (the attacker's payoff function reflects the attacker's profit from a flooding attack). In the defense mechanisms proposed in this section, the defender adopts the Nash equilibrium prescription that brings him the maximum possible repeated game payoff while preventing the attack. In this way, the defense mechanism would be optimal.

2.3 Considerations for Distributed Attacks

The optimal puzzle-based defense strategies are developed. More specifically, four defense mechanisms are proposed. PDM1 is derived from the open-loop solution concept in which the defender chooses his actions regardless of what happened in the game history.
This mechanism is applicable in defeating the single-source and distributed attacks, but it cannot support the higher payoffs being feasible in the game. PDM2 resolves this by using the closed-loop solution concepts, but it can only defeat a single-source attack. PDM3 extends PDM2 and deals with distributed attacks.
This defense is based on the assumption that the defender knows the size of the attack coalition. Finally, in PDM4, the ultimate defense mechanism is proposed in which the size of the attack coalition is assumed unknown.
PDM1 treats a distributed attack as a single-source attack, where the attackers are modeled as a single attacker with the capabilities of the corresponding attack coalition. The same approach can be adopted for closed-loop solutions, but some further issues should be considered there. In a distributed attack, the requests come from different machines, and it is no longer reasonable to assume that the defender receives only a small number of requests before receiving the correct or random answer to an issued puzzle. Indeed, a large number of requests are produced by the attack coalition, whereas a small proportion of them are of a single machine. Therefore, in the time a machine is involved in computing the answer, the defender may receive a large number of requests from the other machines in the coalition.

2.4 The Pseudonym Manager

After solving the puzzle the client must contact with the Pseudonym Manager (PM) (i.e., not through a known anonymizing network) to establish control over a resource for IP-address blocking. We assume the PM has knowledge about Tor routers .For every user who registering at first the pseudo manager generates the key value called pseudo number. This number will use to deposit in the nymble manager to communicate with the server. Actually this pseudo numbers will helps to server to make the blacklist of the misbehave users. Pseudonyms are deterministically chosen based on the controlled

resource, ensuring that the same pseudonym is always issued for the same resource. Note that the user does not disclose what server he or she
Intends to connect to, and the PM's duties are limited to mapping IP addresses (or other resources) to pseudonyms. The user contacts the PM only once per linkability window (e.g., once a day).

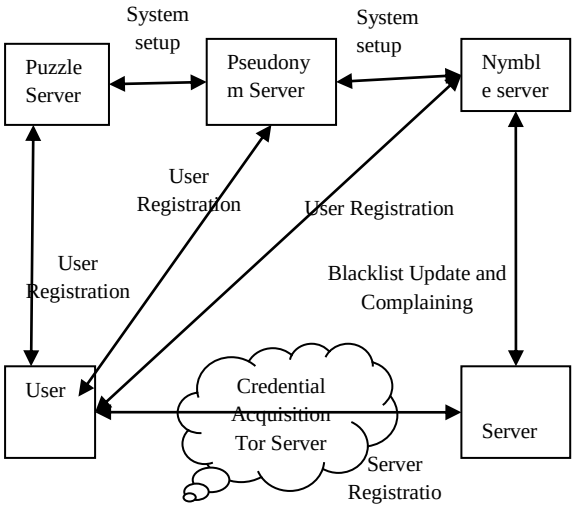

Fig 1.The Nymble System Architecture

2.5 The Nymble Manager

Once the user gets the key value from the pseudo manager, then the next step is to deposit that ticket into nymble. This authority verifies the ticket (key) in the blacklist table and also stores the user IP for the additional process. Next level their connection will be denied further. After receiving a pseudonym from the PM, the user associates to the Nymble Manager (NM) through the anonymizing network, and requests nymbles for access to a particular server (such as Wikipedia). A user's requests to the NM are therefore pseudonymous, and nymbles are yielded using the user's pseudonym and the server's identity. These nymbles are specific to a particular user-server pair. As long as the PM and the NM do not collude, the Nymble system cannot identify which user is connecting to what server; the NM knows only the pseudonym-server pair, and the PM knows only the user identity-pseudonym pair. To provide the necessary cryptographic protection and security properties, the NM encapsulates nymbles within nymble tickets. Servers envelop seeds into linking tokens, and therefore, we will speak of linking tokens being used to link future nymble tickets.

2.6 Time

Nymble tickets are limit to specific time periods. Time is divided into linkability windows of duration W, each of which is split into L time periods of duration T (i.e., W ¼L _ T). While a user's access within a time period is tied to a single nymble ticket, the use of different nymble tickets across time periods grants the user anonymity between time periods.

2.7 Blacklisting

Blacklisting misbehave user is the main process of server. If the users defacing or misbehave with the server response then the server make the note of the ticket (key) and send to the blacklist. Once the blacklist received the key of a user then nymble authority closed the connection of the user IP, which can also get from the blacklist table.

III. DATA STRUCTURES

Nymble uses several important data structures:

3.1 Pseudonym creation:

A pseudonym pnym has two components nym and mac: nym is a pseudorandom mapping of the user's identity, the link ability window w for which the pseudonym is valid, and the PM's secret key nymKeyP ; mac is a MAC that the NM uses to verify the integrity of the pseudonym.

Algorithm 1. PMCreatePseudonym
Input:(uid.w) € H×N
Persistent state: pmState € SP
Output: pnym € P
1: Extract nymKeyP , macKeyNP from pmState
2: nym :=MA.Mac(uid||w,nymKeyP)
3: mac :=MA.Mac(nym||w,macKeyNP)
4: return pnym :=(nym,mac)

3.2 Nymble Tickets and Credentials:

A ticket contains a nymble specific to a server, time period, and linkability window. ctxt is encrypted data that the NM can use during a complaint involving the nymble ticket. In particular, ctxt contains the first nymble (nymble) in the user's sequence of nymbles, and the seed used to generate that nymble. Upon a complaint, the NM extracts the user's seed and issues it to the server by evolving the seed, and nymble helps the NM to recognize whether the user has already been blacklisted.

Algorithm 2.NMCreateCredential
Input: (pnym,sid,w) € P ×H×N
Persistent state: nmState €S_N
Output: cred € D
1. Extract $macKey_{NS}$, $macKey_N$, $seedKey_N$, $encKey_N$ from keys in nmState
2.$seed_0$:=f(Mac(pnym||sid||w,$seedKey_N$))
3. nimble*:= g($seed_0$)
4. for t from 1 to L do
5. $seed_t$:=f($seed_{t\_1}$)
6. $nymble_t$:= g($seed_t$)
7. $ctxt_t$:=Enc.Encrypt(nymble*||$seed_t$, $encKey_N$)
8.$ticket_t$:=sid||t||w||nymblet||$ctxt_t$
9. $mac_{N,t}$:= MA.Mac(ticket'$_t$, $macKey_N$)
10. $mac_{NS,t}$:= MA.Mac(ticket'$_t$||$mac_{N,t,}$ $macKey_{NS}$)
11. tickets[t] := (t, $nymble_t$,$ctxt_{t,}$ $mac_{N,t,,}$ $mac_{NS,t}$)
12. return cred := (nymble*, tickets)

3.3 Blacklists

A server's blacklist is a list of nymbles corresponding to all the nymbles that the server has complained about. Users can quickly check their blacklisting status at a server bychecking to see whether their nymble appears in the server's blacklist.

Algorithm 3: UserCheckIfBlacklisted
Input:(sid; blist)€ H×B_n, n,l € N_0
Persistent state: usrState €Su
Output: b €{true; false}
1: Extract nymble*from cred in usrEntries[sid] in usrState
2: return (nymble*€blist)

IV. CONCLUSION

This paper utilizes game theory to propose a number of puzzle-based defenses against flooding attacks. It is shown that the interactions between an attacker who launches a flooding attack and a defender who counters the attack using a puzzle-based defense can be modeled as an infinitely repeated game of discounted payoffs. Then, the solution concepts of this type of games are deployed

to find the solutions, i.e., the best strategy a rational defender can adopt in the face of a rational attacker. In this way, the optimal puzzle-based defense strategies are developed. A complete flooding attack solution is likely to require some kind of defense during the attack traffic identification. The mechanisms of this paper can provide such defenses. On the other hand, the estimations made by a reactive mechanism can be used in tuning the mechanisms. This section discusses some aspects of the puzzle-based defense mechanisms proposed in this paper and outlines future researches in the game-theoretic study of the client- puzzle approach. It also compares these mechanisms with some of the earlier puzzle-based defenses against flooding attacks. If the game continues at each period with a probability less than the unity, it is also of discounted payoffs, where the future payoffs are lowered using a discount factor.

REFERENCE

[1]. C. Cornelius, A. Kapadia, P.P. Tsang, and S.W. Smith, "Nymble: Blocking Misbehaving Users in Anonymizing Networks," Technical Report TR2008-637, Dartmouth College, Computer Science, Dec. 2008.
[2]. Douglas Stebila,Lakshmi Kuppusamy, Jothi Rangasamy,Colin Boyd,"Stronger Difficulty Notions For Client Puzzle And Denial Of Service Resistant Prorocol"Information Security Institute, Queensland University Of Technology,Australia,Dec 21, 2010.
[3]. A. Lysyanskaya, R.L. Rivest, A. Sahai, and S. Wolf, "Pseudonym Systems," Proc. Conf. Selected Areas in Cryptography, Springer, pp. 184-199, 1999.
[4]. P.P. Tsang, M.H. Au, A. Kapadia, and S.W. Smith, "Blacklistable Anonymous Credentials: Blocking Misbehaving Users without TTPs,"

AUTHORS BIOGRAPHY

T.Shanmugapriya was born in Tamilnadu, India in 1984. She Received her B.E. Degree in Computer Science and Engineering from Anna University Chennai. She completed her M.Tech Information Technology in Anna University of Technology, Coimbatore.

C.MageshKumar was born in Tamilnadu, India in 1987. He Received her B.Tech. Degree in Information Technology from Anna University Chennai. He pursuing M.Tech Information Technology in Anna University of Technology, Coimbatore.

S.P.Kavya was born in Tamilnadu, India in 1989. She Received her B.E. Degree in Computer Science and Engineering from Anna University of Technology Coimbatore. She is pursuing her M.Tech Information Technology in Anna University Chennai.

M.Nandhini was born in Tamilnadu, India in 1989. She Received her B.Tech. Degree in Information Technology from Anna University Chennai. She completed her M.E. Computer Communication Engineering in Anna University Chennai