Game Theoretic Resistance to DoS Attacks Using Hidden Difficulty Puzzles

Harikrishna[1], Venkatanathan[1] and Pandu Rangan[2]

[1]College of Engineering Guindy, Anna University Chennai,Tamil Nadu, India

[2]Indian Institute of Technology, Madras, Tamil Nadu, India

Outline

Proof-of-Work

- A good mechanism to counterbalance computational expenditure during a denial of service (DoS) attack.
- Proposed by Dwork and Naor (1992) to control junk mails.
- On receiving a request, server generates a puzzle and sends it to the client.
- The client solves the puzzle and sends a response.
- The server verifies the solution and provides the service only if the solution is correct.

Puzzle Difficulty

- A challenge in the client-puzzle approach is deciding on the difficulty of the puzzle.
- The puzzle difficulty could be adjusted based on the server load (Feng et al. 2005).
- But this would affect the quality of service to legitimate users.
- Instead, the puzzle difficulty could be varied based on a probability distribution.

Game Theory

- A denial of service attack is viewed as a two player game between an attacker and a defending server.
- Bencsath (2003) et al. was the first to model the client-puzzle approach as a strategic game.
- Fallah (2010) extended the work further by using infinitely repeated games.
- Jun-Jie (2008) applied game theory to puzzle auctions.

Outline

Aim of the Paper

- Introduce the notion of **'hidden puzzle difficulty'** in client-puzzles.
- Propose new puzzles that satisfy this property.
- Show that a defense mechanism is more effective when it uses a hidden difficulty puzzle.

Hash Reversal Puzzle

- Hash Reversal Puzzle proposed by Juels and Brainard (1999).
- S - Server Secret, N_S - Server Nonce, M - Session Parameter

Client		Defender
	$\xrightarrow{Request}$	$X = H(S, N_s, M)$ $Y = H(X)$
	$\xleftarrow{(X', Y), N_s}$	$X' = X \ \& \ (0_1, 0_2, ..., 0_k, 1_{k+1}, ..., 1_n)$
Find rp such that $H(rp) = Y$	$\xrightarrow{rp, N_s}$	$X = H(S, N_s, M)$ $H(rp) \stackrel{?}{=} H(X)$

Hidden Difficulty Puzzle 1 – Modified Hash Reversal Puzzle

Hidden Difficulty Property

"The difficulty of the puzzle should not be determined by the attacker without expending a minimal amount of computational effort."

- Some of the first k bits of X are inverted.
- k determines puzzle difficulty, but is **hidden**.

Client		Defender
	$\xrightarrow{Request}$	$X = H(S, N_s, M)$ $Y = H(X)$
	$\xleftarrow{(X', Y), N_s}$	$X' = X \oplus (l_1, l_2, ..., l_{k-1}, 1, 0_{k+1}, ..., 0_n)$
Find rp such that $H(rp) = Y$	$\xrightarrow{rp, N_s}$	$X = H(S, N_s, M)$ $H(rp) \stackrel{?}{=} H(X)$

Hidden Difficulty Puzzle 1 – Modified Hash Reversal Puzzle

Hidden Difficulty Property

"The difficulty of the puzzle should not be determined by the attacker without expending a minimal amount of computational effort."

- Some of the first k bits of X are inverted.
- k determines puzzle difficulty, but is **hidden**.

Client		**Defender**
	$\xrightarrow{Request}$	$X = H(S, N_s, M)$ $Y = H(X)$
	$\xleftarrow{(X', Y), N_s}$	$X' = X \oplus (l_1, l_2, ..., l_{k-1}, 1, 0_{k+1}, ..., 0_n)$
Find rp such that $H(rp) = Y$	$\xrightarrow{rp, N_s}$	$X = H(S, N_s, M)$ $H(rp) \stackrel{?}{=} H(X)$

Outline

Game Model

- An extension of the model proposed by Fallah (2010).
- Defender and Attacker are players in a strategic game.
- The attacker is rational (strongest attacker).
- Legitimate user is not a player in the game.

Defender Actions

- Defender chooses from n puzzles, P_1, P_2, ..., P_n of varying difficulties.
- It can be shown that two puzzles are sufficient for an effective defense mechanism.
- Defender's choice is between P_1 (**Easy**) and P_2 (**Hard**).

Attacker Actions

- **CA** - Correctly answer the puzzle
- **RA** - Randomly answer the puzzle
- **TA** - Try to answer the puzzle correctly, *but give up if it is too hard.*
- In the case of **TA**, the attacker gives a correct answer if the puzzle is solved and a random answer if he gives up.

Notations

Term	Meaning
T	Reference time period.
α_m	Fraction of T to provide the service.
α_{PP}	Fraction of T to produce a puzzle.
α_{VP}	Fraction of T to verify the solution.
α_{SP_1}	Fraction of T to solve P_1.
α_{SP_2}	Fraction of T to solve P_2.

- Defender chooses P_1 and P_2 such that $\alpha_{SP_1} < \alpha_m < \alpha_{SP_2}$.

Attacker Payoff

- Assume attacker receives puzzle P_i.
- If his response is CA, his payoff is

$$\alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_i}$$

- If his response is RA, his payoff is

$$\alpha_{PP} + \alpha_{VP}$$

- If his response is TA, his payoff depends on when whether he gives up or not.

Attacker Payoff (Contd.)

- Assume the puzzle difficulty is known.

Attacker Payoff (Contd.)

- Assume the puzzle difficulty is known.
- The attacker's best response to puzzle P_1 is CA as $\alpha_{SP_1} < \alpha_m$.

$$u_2(P_1; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

$$u_2(P_1; RA) = \alpha_{PP} + \alpha_{VP}$$

Attacker Payoff (Contd.)

- Assume the puzzle difficulty is known.
- The attacker's best response to puzzle P_1 is CA as $\alpha_{SP_1} < \alpha_m$.

$$u_2(P_1; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

$$u_2(P_1; RA) = \alpha_{PP} + \alpha_{VP}$$

- Positive

Attacker Payoff (Contd.)

- Assume the puzzle difficulty is known.
- The attacker's best response to puzzle P_1 is CA as $\alpha_{SP_1} < \alpha_m$.

$$u_2(P_1; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

$$u_2(P_1; RA) = \alpha_{PP} + \alpha_{VP}$$

- Positive
- The attacker's best response to puzzle P_2 is RA as $\alpha_{SP_2} > \alpha_m$.

$$u_2(P_2; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_2}$$

$$u_2(P_2; RA) = \alpha_{PP} + \alpha_{VP}$$

Attacker Payoff (Contd.)

- Assume the puzzle difficulty is known.
- The attacker's best response to puzzle P_1 is CA as $\alpha_{SP_1} < \alpha_m$.

$$u_2(P_1; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

$$u_2(P_1; RA) = \alpha_{PP} + \alpha_{VP}$$

- Positive
- The attacker's best response to puzzle P_2 is RA as $\alpha_{SP_2} > \alpha_m$.

$$u_2(P_2; CA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_2}$$

$$u_2(P_2; RA) = \alpha_{PP} + \alpha_{VP}$$

- Negative

Attacker Payoff – Try and Answer

- *TA* is relevant only if the puzzle difficulty is hidden.

Attacker Payoff – Try and Answer

- *TA* is relevant only if the puzzle difficulty is hidden.
- The attacker puts in the minimal effort required to solve P_1 and gives up when he realizes the puzzle is P_2.

Attacker Payoff – Try and Answer

- TA is relevant only if the puzzle difficulty is hidden.
- The attacker puts in the minimal effort required to solve P_1 and gives up when he realizes the puzzle is P_2.
- If the puzzle sent is P_1, he would send the correct answer.

$$u_2(P_1; TA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

Attacker Payoff – Try and Answer

- *TA* is relevant only if the puzzle difficulty is hidden.
- The attacker puts in the minimal effort required to solve P_1 and gives up when he realizes the puzzle is P_2.
- If the puzzle sent is P_1, he would send the correct answer.

$$u_2(P_1; TA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

- If the puzzle sent is P_2, he would give up after expending α_{SP_1} amount of effort.

$$u_2(P_2; TA) = \alpha_{PP} + \alpha_{VP} - \alpha_{SP_1}$$

Attacker Payoff – Try and Answer

- *TA* is relevant only if the puzzle difficulty is hidden.
- The attacker puts in the minimal effort required to solve P_1 and gives up when he realizes the puzzle is P_2.
- If the puzzle sent is P_1, he would send the correct answer.

$$u_2(P_1; TA) = \alpha_{PP} + \alpha_{VP} + \alpha_m - \alpha_{SP_1}$$

- If the puzzle sent is P_2, he would give up after expending α_{SP_1} amount of effort.

$$u_2(P_2; TA) = \alpha_{PP} + \alpha_{VP} - \alpha_{SP_1}$$

- Minimal Effort

Defender Payoff

- Unlike the attacker, a legitimate user always gives the correct answer.
- The defender seeks to maximize the effectiveness of the defense mechanism and minimize the cost to a legitimate user.
- We introduce a balance factor $0 < \eta < 1$ that allows him to strike a balance between the two.
- Payoff:
  $u_1 = (1 - \eta)(-\text{attacker payoff}) + \eta(-\text{legitimate user cost})$.

Outline

Preliminaries – Mixed Strategy

- A mixed strategy is a probability distribution over a players actions.
- The defender could send P_1 with a probability p and P_2 with probability $1 - p$.
- We represent such a mixed strategy as $(p \circ P_1 \oplus (1 - p) \circ P_2; TA)$.
- Similarly, the attacker could choose a lottery over CA, TA and RA.

Nash Equilibrium

- A Nash equilibrium exists if each player has chosen a strategy and no player can benefit by unilaterally changing his strategy.
- Fallah (2010) constructed a defense mechanism by using Nash equilibrium is used here in a prescriptive manner.
- The defender selects and takes part in a specific equilibrium profile and the best thing for the attacker to do is to conform to his equilibrium strategy.

Defense Mechanism 1 - Equilibrium Strategy

- The defender sends P_1 with probability p and P_2 with probability $1 - p$.
- The attacker tries to solve the puzzle (and gives a correct answer only for P_1)

Theorem

In the strategic game of the client-puzzle approach, for $0 < \eta < \frac{1}{2}$, *a Nash equilibrium of the form* $(p \circ P_1 \oplus (1 - p) \circ P_2; TA)$, *exists if*

$$\eta = \frac{\alpha_m}{\alpha_m + \alpha_{SP_2} - \alpha_{SP_1}},$$

$$\alpha_{SP_2} - \alpha_{SP_1} > \alpha_m \text{ and}$$

$$p > \frac{\alpha_{SP_1}}{\alpha_m}.$$

Mitigating DoS Attack

- A Nash equilibrium does not prevent the flooding attack from being successful.
- Let N be the maximum number of requests that an attacker can send in time T (reference time).
- The defender is overloaded when

$$Np\alpha_m > 1.$$

- So to prevent a DoS attack, we must ensure that

$$Np\alpha_m \leq 1 \text{ or } p \leq \frac{1}{N\alpha_m}.$$

Comparison with Previous Work

- HDM1 - Defense mechanism using hidden difficulty puzzles.
- PDM1 - Defense mechanism using known difficulty puzzles (Fallah 2010).
- Expected payoff of the attacker in HDM1 is

$$\alpha_{PP} + \alpha_{VP} + p\alpha_m \boxed{-\alpha_{SP_1}}.$$

- Expected payoff of the attacker in PDM1 is

$$\alpha_{PP} + \alpha_{VP} + p\alpha_m \boxed{-p\alpha_{SP_1}}.$$

- The expected payoff of an attacker in HDM1 is lower than in PDM1.
- The payoff of the defender is the same in both defense mechanisms.

Outline

Repeated Games

- Two flavors of game theory:
- **Strategic games:** A single-shot game where a decision-maker ignores the decisions in previous plays of the game.
- **Repeated games:** A multi-period game where a player's decision is influenced by decisions taken in all periods of the game.
- During a denial of service attack, the attacker repeatedly sends requests to the defender.
- The scenario is modeled as an **infinitely repeated game**.

Threat of Punishment

- In a repeated game, a player would be willing to take sub-optimal decisions if it would give him a higher payoff in the long run.
- Deviation of a player from a desired strategy can be prevented if he is threatened with sufficient punishment in the future.
- A Nash equilibrium with high payoff can be achieved if a player is patient enough to see long term benefits over short term gains.

The Folk Theorem

- The **minmax payoff** of a player is the minimum payoff that he can guarantee himself in a game, even when the opponents play in the most undesirable manner.
- A player's minmax strategy against an opponent would reduce the opponent's payoff to the minmax payoff.
- A Nash equilibrium where each player receives an average payoff above his minmax payoff is possible through the threat of punishment (Fudenberg and Maskin 1986).

Two Phase Equilibrium

Normal Phase (A)

- The defender and attacker choose a strategy profile, where each of them receive a payoff greater than the minmax payoff.
- If either of them deviate, the game switches to the punishment phase (B).

Punishment Phase (B)

- Each player chooses a minmax strategy against the other player for τ periods, after which the game switches to the normal phase.
- Any deviation from this strategy would restart the phase.

Minmax Strategies

- Defender's Minmax Strategy

Theorem

In the game of the client-puzzle approach, when $\alpha_{SP_2} - \alpha_{SP_1} < \alpha_m$, one of the defender's minmax strategy against the attacker is

$$p_1 \circ P_1 \oplus (1 - p_1) \circ P_2,$$

where $p_1 = \dfrac{\alpha_{SP_2} - \alpha_m}{\alpha_{SP_2} - \alpha_{SP_1}}$.

Minmax Strategies (Contd.)

- Attacker's Minmax Strategy

Theorem

In the game of the client-puzzle approach, when $\alpha_{SP_2} - \alpha_{SP_1} < \alpha_m$ *and* $0 < \eta < \frac{1}{2}$*, the attacker's minmax strategy against the defender is*

$$p_2 \circ CA \oplus (1 - p_2) \circ RA,$$

where $p_2 = \frac{\eta}{1 - \eta}$.

Defense Mechanism

- **Punishment Phase:** The defender chooses the mixed strategy

$$p_1 \circ P_1 \oplus (1 - p_1) \circ P_2,$$

while the attacker chooses the mixed strategy

$$p_2 \circ CA \oplus (1 - p_2) \circ RA.$$

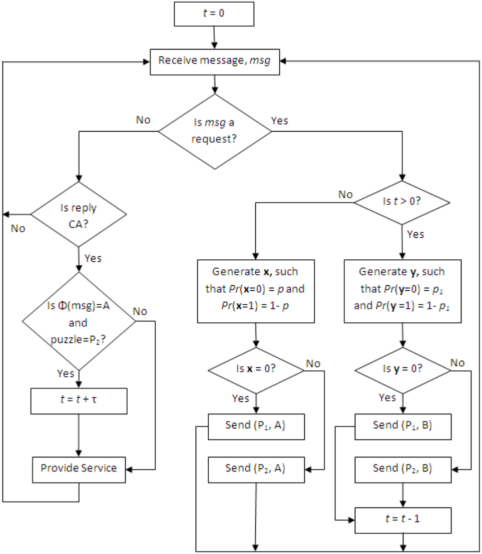
- **Normal Phase:** The defender chooses the mixed strategy

$$p \circ P_1 \oplus (1 - p) \circ P_2,$$

while the attacker chooses the strategy TA.

- The defender receives higher payoff in the Nash equilibrium of the repeated game than in the Nash equilibrium of the single-shot strategic game.

Flow Chart



t = 0
Receive message, msg
Is msg a request?
No
Yes
Is reply CA?
No
Yes
Is Φ(msg)=A and puzzle=P₂?
No
Yes
t = t + τ
Provide Service
Is t > 0?
No
Yes
Generate x, such that Pr(x=0) = p and Pr(x=1) = 1- p
Generate y, such that Pr(y=0) = p₁ and Pr(y =1) = 1- p₁
Is x = 0?
No
Yes
Is y = 0?
No
Yes
Send (P₁, A)
Send (P₂, A)
Send (P₁, B)
Send (P₂, B)
t = t - 1

Comparison with Previous Work

- HDM2 - Defense mechanism based on repeated game using hidden difficulty puzzles.
- PDM2 - Defense mechanism based on repeated game using known difficulty puzzles (Fallah 2010).
- The minmax payoff of the defender in HDM2 is
$$(1-\eta)(-\alpha_{PP}-\alpha_{VP})-\boxed{\eta\alpha_m}.$$
- The minmax payoff of the defender in PDM2 is
$$(1-\eta)(-\alpha_{PP}-\alpha_{VP})-\boxed{\eta\alpha_{SP_2}}.$$
- The minmax payoff of the defender in HDM2 is higher than that in PDM2.

Comparison with Previous Work (Contd.)

- The minmax payoff of the attacker is the same in both defense mechanisms.
- Since the minmax payoff is a lower bound on the defender's payoff, the defender is better off in HDM2.
- In PDM2, only P_2 puzzles are sent in punishment phase.
- In HDM2, a lottery over P_1 and P_2 is adopted.
- A legitimate user is hurt less in the punishment phase of HDM2.

Distributed Attacks

- The computational power of the attacker increases proportionally with the size of the attack coalition.
- When s machines are used, the attacker can send sN requests in time T.
- The conditions for the first defense mechanism to handle distributed attacks are

$$\frac{\alpha_{SP_1}}{s} < \frac{1}{N} < \alpha_m < \frac{\alpha_{SP_2}}{s},$$

$$\alpha_{SP_2} - \alpha_{SP_1} > s\alpha_m,$$

$$\eta = \frac{\alpha_m}{\alpha_m + \alpha_{SP_2} - \alpha_{SP_1}} \text{ and}$$

$$\frac{\alpha_{SP_1}}{s\alpha_m} < p < \frac{1}{N\alpha_m}.$$

Outline

Properties of HDPs

- **Hidden Difficulty:** The difficulty of the puzzle should not be determined without a minimal computations.
- **High Puzzle Resolution:** The granularity of puzzle difficulty must be high allowing us to fine tune the system parameters.
- **Partial Solution:** Submission of partial solutions should be possible (to differentiate between *RA* and *TA*.)

Hidden Difficulty Puzzle 2

Client		Defender
	Request $\longrightarrow$	$X = H(S_1, N_s, M)$
		$Y = H(X)$
		$a = H(S_2, N_s, M) \bmod D + I$
		$X' = X - a$
		$Z = H(X')$
	$\longleftarrow$ $(X'', Y, Z), N_s$	$X'' = X' \oplus (I_1, ..., I_{k-1}, 1, 0_{k+1}, ..., 0_n)$
Find $rp1$ such that $H(rp1) = Z$. Find a' such that $H(rp2) = Y$, where $rp2 = rp1 + a'$.	$rp1, rp2, N_s$ $\longrightarrow$	$X = H(S_1, N_s, M)$
		$a = H(S_2, N_s, M) \bmod D + I$
		$H(rp1) \stackrel{?}{=} H(X - a)$
		$H(rp2) \stackrel{?}{=} H(X)$

Hidden Difficulty Puzzle 3

Client		Defender
	$\xrightarrow{Request}$	$X = H(S_1, N_s, M)$ $Y = H(X)$ $a = H(S_2, N_s, M) \bmod D_a + l$ $X' = X - a$ $Z = H(X')$
	$\xleftarrow{(X'', Y, Z), N_s}$	$X'' = X' - b$
Find b' such that $H(rp1) = Z$, where $rp1 = X'' + b'$. Find a' such that $H(rp2) = Y$, where $rp2 = rp1 + a'$.	$\xrightarrow{rp1, rp2, N_s}$	$X = H(S_1, N_s, M)$ $a = H(S_2, N_s, M) \bmod D_a + l$ $H(rp1) \stackrel{?}{=} H(X - a)$ $H(rp2) \stackrel{?}{=} H(X)$

Hash Computations

- We present the number hash computations required for generating, verifying and solving the proposed puzzles.

Puzzle	Generation	Verification (max)	Solving (avg)	Partial Solution
HDP1	2	3	$\frac{(2^k+1)}{2}$	No
HDP2	4	6	$\frac{(2^k+1)\ +\ (D+1)}{2}$ $(I = 1)$	Yes
HDP3	4	6	$\frac{(D_a+1)\ +\ (D_b+1)}{2}$ $(I = 1)$	Yes

Term	Meaning
H	Hash Function
S	Server Secret
N_S	Server Nonce
M	Session Parameter
I	Random Binary Number
k	No. of bits to be inverted

Outline

Conclusions

- We have given emphasis on hiding the difficulty of client-puzzles from a denial of service attacker.
- Three concrete puzzles that satisfy this requirement have been constructed.
- Using game theory, we have developed defense mechanisms that are more effective than the existing ones.
- Future direction of work would be to incorporate the defense mechanisms in existing protocols and to estimate its effectiveness in real-time.

Outline

References

- Dwork, C., Naor, M.: Pricing via processing or combatting junk mail. In: *Brickell, E.F. (ed.) CRYPTO 1992. LNCS*, vol. 740, pp. 139–147. Springer, Heidelberg (1993).
- Juels, A., Brainard, J.: Client puzzles: A cryptographic countermeasure against connection depletion attacks. In: *Proceedings of NDSS 1999 (Networks and Distributed Security Systems)*, pp. 151–165 (1999)
- Bencsath, B., Vajda, I., Buttyan, L.: A game based analysis of the client puzzle approach to defend against dos attacks. In: *Proceedings of the 2003 International Conference on Software, Telecommunications and Computer Networks*, pp. 763–767 (2003).

References (Contd.)

- Feng, W., Kaiser, E., Luu, A.: Design and implementation of network puzzles. In: *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, March 2005, vol. 4, pp. 2372–2382 (2005).
- Lv, J.-J.: A game theoretic defending model with puzzle controller for distributed dos attack prevention. In: *2008 International Conference on Machine Learning and Cybernetics*, July 2008, vol. 2, pp. 1064–1069 (2008)
- Fallah, M.: A Puzzle-Based Defense Strategy Against Flooding Attacks Using Game Theory. In: *IEEE Trans. Dependable and Secure Computing*, vol. 7, no. 1, pp. 5–19 (2010).