Game-based Analysis of Denial-of-Service Prevention Protocols

Ajay Mahimkar

Class Project: CS 395T

Overview

- Introduction to DDoS Attacks
- Current DDoS Defense Strategies
- Client Puzzle Protocols for DoS Prevention
- Distributed Approach
- Game-based Verification using MOCHA
- Conclusions and future work

DDoS Attacks

- What is a Denial-of-Service Attack?
  - Degrade the service quality or completely disable the target service by overloading critical resources of the target system or by exploiting software bugs
- What is a Distributed Denial-of-Service Attack?
  - The objective is the same with DoS attacks but is accomplished by a set of compromised hosts distributed over the Internet

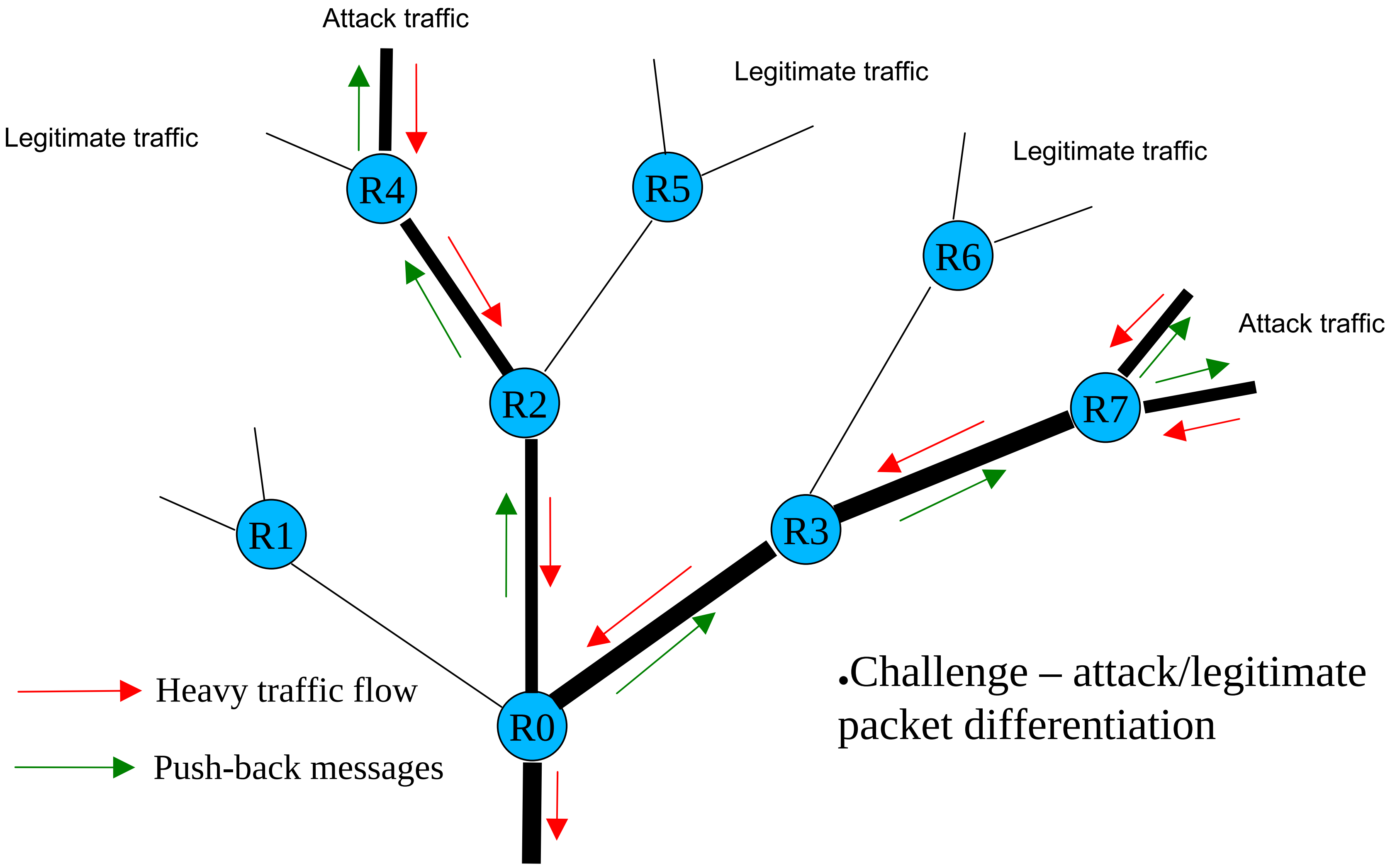
Defense Mechanisms (1)

- Victim-end
  - Most existing intrusion detection systems and DDoS detection systems fall in this category
  - Used to protect a set of hosts from being attacked
  - Advantages
    - DDoS attacks are easily detected due to aggregate of huge traffic volume
  - Disadvantages
    - Attack flows can still incur congestion along the attack path
  - Filtering of attack flows using IP Traceback

Defense Mechanisms (2)

- Intermediate Network
  - Routers identify attack packet characteristics, send messages to upstream routers to limit traffic rate
  - Attack packets filtered by Internet core routers
  - Advantages
    - Effectiveness of filtering improved
  - Disadvantages
    - Internet-wide authentication framework is required
  - Example
    - Push-back Mechanism

Push-back Mechanism



- Challenge – attack/legitimate packet differentiation

Defense Mechanisms (3)

- Source-end
  - Attack packets dropped at sources
  - Prevents attack traffic from entering the Internet
  - Advantages
    - Effectiveness of packet filter is the best
  - Disadvantages
    - It is very hard to identify DDoS attack flows at sources since the traffic is not so aggregate
    - Requires support of all edge routers

Problems

- In DDoS Attack Mitigation techniques, filters do not accurately differentiate legitimate and attack traffic
  - Mechanisms like IP Traceback, Push-back could drop legitimate traffic
  - Dropping legitimate traffic serves the purpose of the attacker
- Question is
  - How to differentiate legitimate and attack traffic behavior?
  - Solution
    - Use Client Puzzles

Client Puzzles

- Force each client to solve a cryptographic puzzle for each request before server commits its resources
  - In other words, “*Make client commit its resources before receiving resource*”
- Client puzzles defends against Distributed DoS attacks
  - Study shows that existing DDoS tools are carefully designed not to disrupt the zombie computers, so as to avoid alerting the machine owners
- Filter packets from clients that do not solve puzzles
  - This differentiates legitimate users from attackers

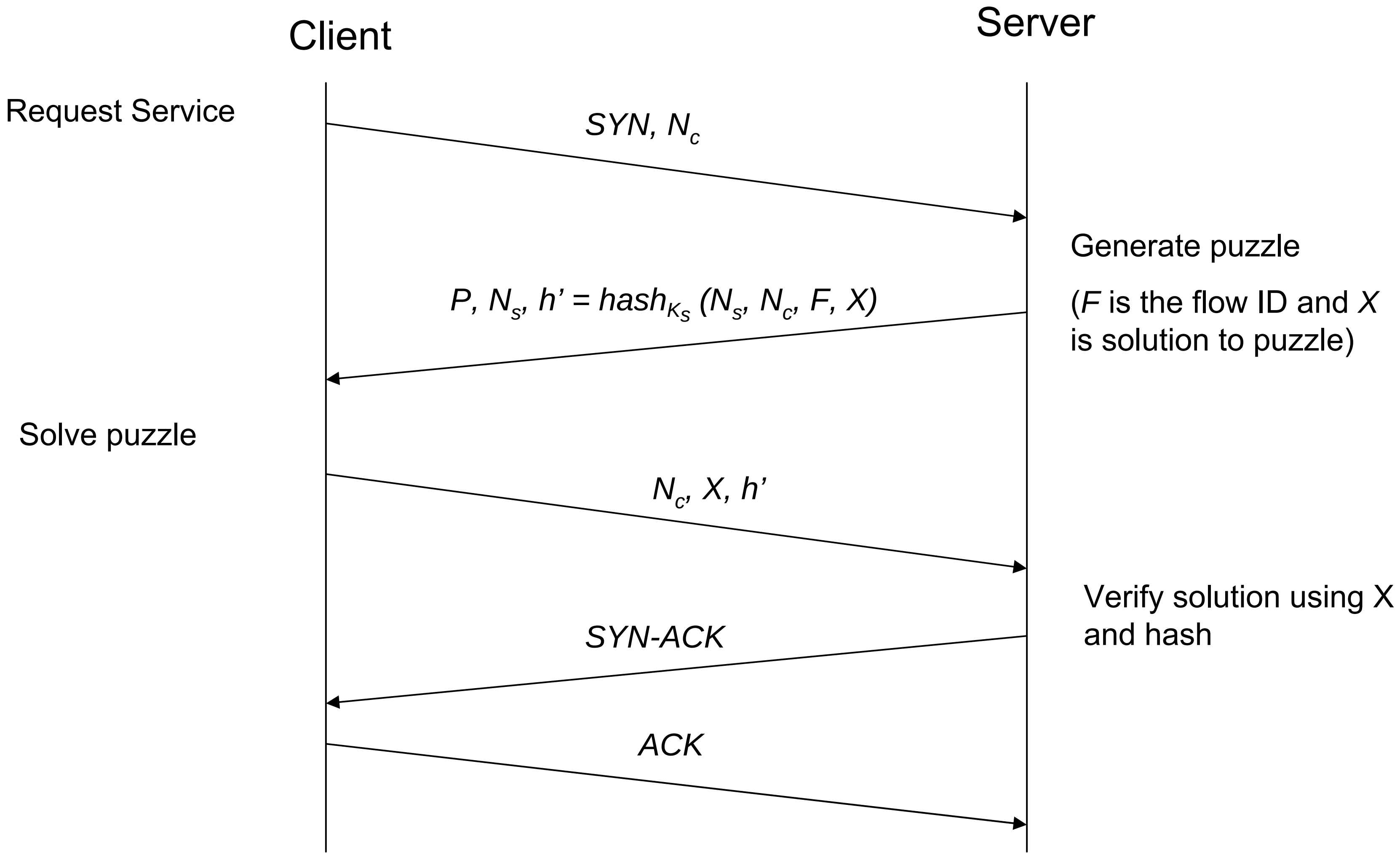
Client Puzzle Protocols (1)

- Puzzle Auctions Protocol
  - Before initiating session, client solves a puzzle of some difficulty level and sends request along with puzzle solution to the server
  - Depending upon the server utilization and the puzzle difficulty level
    - The server sends an accept and continues with the session communication or,
    - It sends a reject and asks client to increase the puzzle difficulty level
      - If client can solve puzzle with higher difficulty level, it gets service
    - Legitimate clients can solve puzzles of high difficulty, whereas attackers have an upper bound
      - Thus attacker cannot prevent legitimate users from accessing service

Client Puzzle Protocols (2)

- Challenge-Response Type Client Puzzle Protocol
  - When server receives request from client, depending upon the current utilization it asks the client to solve a puzzle of some difficulty level
  - Server allocates resources only if it receives solution from the client
  - Server does not maintain information about the puzzles
    - Avoids denial-of-service attacks on the puzzle generation

Basic Client Puzzle Protocol

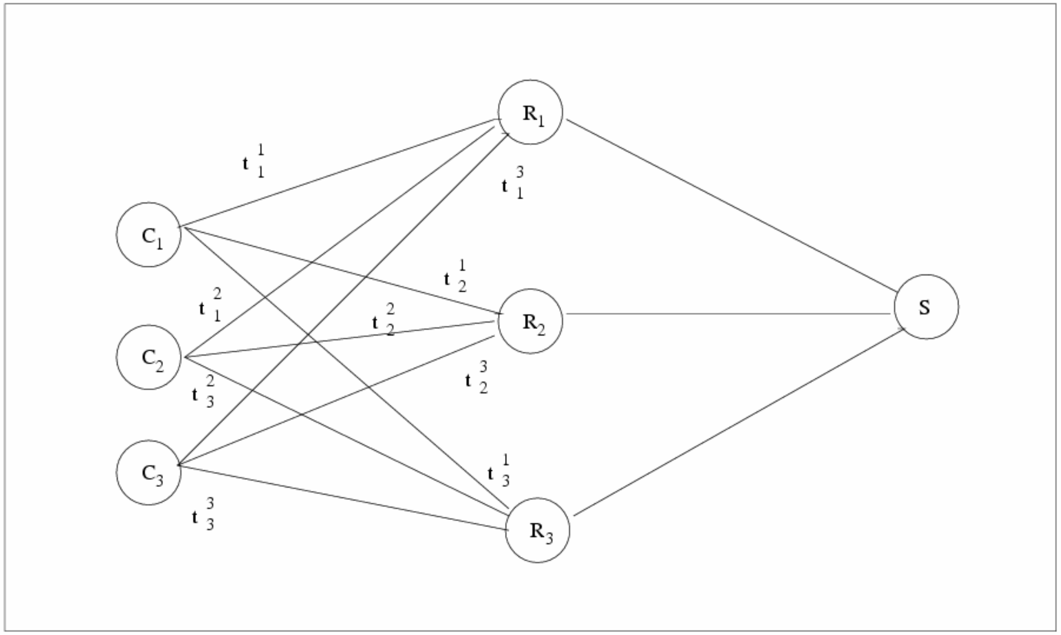

Client
Server
Request Service
SYN, N_c
Generate puzzle
(*F* is the flow ID and *X* is solution to puzzle)
$P, N_s, h' = hash_{K_S} (N_s, N_c, F, X)$
Solve puzzle
N_c, X, h'
Verify solution using X and hash
SYN-ACK
ACK

Distributed Approach (1)

- The two protocols solve *Resource-exhaustion* DDoS attacks
  - Cannot prevent the attacker from flooding the link to the server, thereby exhibiting *Bandwidth-consumption attacks*
- I propose a new approach that shifts puzzle distribution and verification from server to intermediate routers or monitoring nodes
  - Intermediate routers collaborate and determine the total traffic to a certain destination
  - They adapt the difficulty level depending on traffic information
  - Packets from clients that fail to solve puzzles of appropriate difficulty levels are filtered in the intermediate network

Distributed Approach (2)

t^{i}_{j} is the traffic on a link from client *i* to router *j*



$$D^i = \sum_j t^i_j$$

Analysis of the Protocols (1)

- Protocol Properties
  - *Liveness*
    - If a server has enough resources to handle connection requests, then it should allocate resources to clients (genuine or legitimate) that solve puzzles of any difficulty level
  - *Availability*
    - A set of attackers should not be able to prevent legitimate users from accessing the service
  - *Client Authentication*
    - Server allocates resources after authenticating the clients by verifying the solution to the puzzle
  - *Adaptability*
    - Puzzle difficulty level should be in proportion to the traffic levels going to a server

Analysis of the Protocols (2)

- Game-based verification using MOCHA
  - Situation between the attacker and the server modeled as a two-player strategic game
  - Server's strategy is characterized by the complexity of the puzzle that it generates
  - Attacker's strategy is characterized by the amount of effort he invests in solving the received puzzles

Liveness in ATL

$$\underbrace{\langle\langle\ \Sigma\ \rangle\rangle}_{}\ \underbrace{\Box\ ((\neg \mathrm{full})}_{} \rightarrow (\underbrace{(\mathrm{requestC} \rightarrow \mathrm{allocatedC})}_{} \wedge \underbrace{(\mathrm{requestA} \rightarrow \mathrm{allocatedA}))}_{}$$

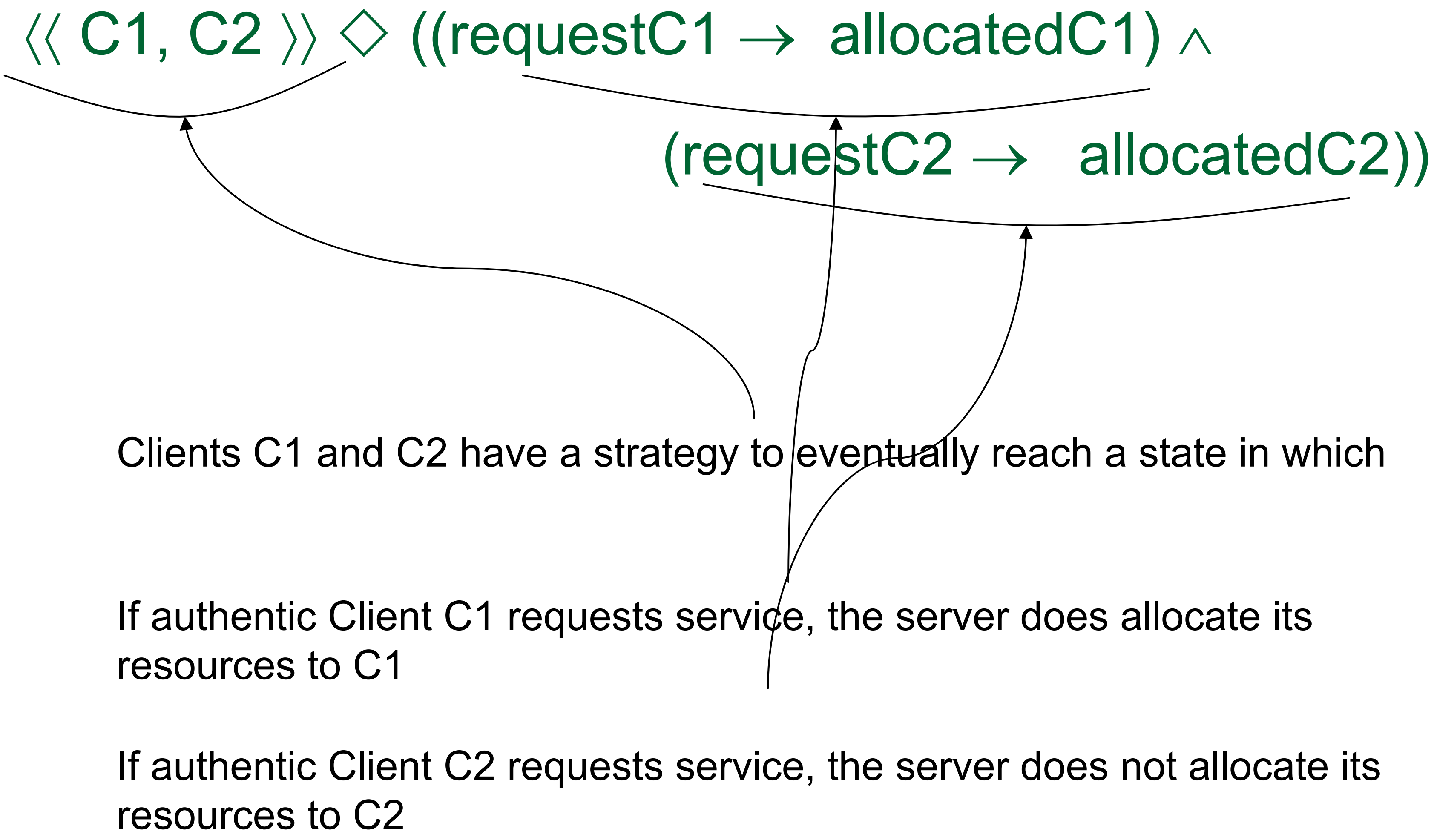
It is always true in all states that

If the server has not committed all of its resources

Request from client C implies that it would be allocated server resources

Request from attacker A implies that it would be allocated server resources

Availability in ATL



$$\langle\langle \text{C1, C2} \rangle\rangle \Diamond ((\text{requestC1} \rightarrow \text{allocatedC1}) \wedge (\text{requestC2} \rightarrow \text{allocatedC2}))$$

Clients C1 and C2 have a strategy to eventually reach a state in which

If authentic Client C1 requests service, the server does allocate its resources to C1

If authentic Client C2 requests service, the server does not allocate its resources to C2

Client Authentication in ATL

$$\langle\langle \Sigma \rangle\rangle \; \Box \; ((allocatedC \rightarrow XC) \wedge (allocatedA \rightarrow XA))$$

It is always true in all states that

If a client is allocated server resources, then it must have solved the puzzle

If an attacker is allocated server resources, then it must have solved the puzzle

Adaptability in ATL

$$\underbrace{\langle\langle\, \Sigma \,\rangle\rangle} \Diamond \underbrace{(\text{difficulty\_levelC} = \text{pkts1}[0] + \text{pkts2}[0])}$$

There exists a state in which

Difficulty level of the puzzle to be solved by a requesting entity C is equal to sum of the packets transmitted from C to the intermediate routers (in this case 1 and 2)

Conclusions and Future Work

- Verified properties of DDoS prevention protocols using game-based tool, MOCHA
    - Liveness, Availability, Client Authentication, Adaptability
- The Distributed approach solves Bandwidth Consumption attacks
    - Adaptation in the puzzle difficulty level using router collaboration and the traffic flow information
- Distributed approach needs to be made more generic to incorporate several flow definitions
- System design and building for Distributed Prevention of DDoS in the network