DNS based DDoS mitigation solution

Review of CID2 and develop future plan

Sukhun Yang

Seoul National University

July 13, 2023

Contents

Table of Contents

DDoS attack

- **Denial of Service Attack (DoS)**
  - Technique to paralyze the target server by generating a large amount of traffic to the target
  - Generally, main purpose is service interruption
- **Distributed Denial of Service Attack (DDoS)**
  - Type of DoS Attack
  - Using zombie PCs acquired by multiple bots or viruses
  - Difficult to defend as attackers are dispersed

DNS server

Domain Name System (DNS)

- Server that connects the domain of the web site and the actual IP address
- **Local DNS**: Actually used by users belonging to the region or telecommunications company
- **Authoritative DNS**: Where relationship between domains and IP addresses is recorded, stored, and changed
- Open DNS: Created by combining multiple DNS servers

Puzzle

In this presentation, puzzle means hash puzzle.
Puzzle is a kind of hash function have to solve for validate query.
Same with have to find answer for problems.

Cloud Flare

Cloud Flare

Communication when a query is requested, the query is frozen for about 5 seconds, and then the query is accept if it is still valid.
⇒ The advantage of being able to block DDoS attack techniques that only send Syn packets.
⇒ If the number of bots increases, can't effectively defend!

Puzzle based DDoS mitigation solution

1. Feng, Wu-chi, Edward Kaiser, and Antoine Luu. "Design and implementation of network puzzles." Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Socie ties.. Vol. 4. IEEE (2005)
2. Abliz, Mehmud, and Taieb Znati. "A Guided tour puzzle for denial of service prevention," 2009 Annual Computer Security Applications Conference. IEEE (2009)
3. Abraham. Anup Mathew, and Shweta Vincent. "Defending DoS Attackers Using a Puzzle-Based Approach and Reduction in Traceback Time towards the Attacker." Global Trends in Computing and Communication Systems.: 4th International Conference, ObCom (2011)

Approach

- **Participation of local DNS server to DDoS protection**
  - Difficult for the target server to respond to distributed attackers
  - Local DNS server manages internal users
- **Rate limiting using puzzle**
  - The key of DDoS defense is to reduce the attack rate
  - Solving puzzles consumes users time and resources

Goals

Establishment of DDoS Mitigation System in which DNS server and target server cooperate!

Goals

1. Design puzzle based DDoS mitigation system
2. Modifying linux kernel for mounting our system
3. Implementing servers and bots to make own testbed

Requirement 1

Design puzzle based DDoS mitigation system

1. Local DNS server presents the seed value of the puzzle to the user.
2. User obtains the solution of the puzzle received from the DNS server and transmits it together when sending a request to the target server.
3. Target server verifies whether the correct puzzle and solution are transmitted, and accepts the connection only if it is correct, otherwise it rejects the connection.

Requirement 2

Make own testbed

1. Implements Client and DNS server using socket programming.
2. Implements bots and put it on the Raspberry pi.
3. Server and bots comunicates with normal TCP and UDP communication.

Requirement 3

Test our DDoS mitigation system

1. Implement puzzle-based DDoS mitigation system by modifying the Linux kernel.
2. Effectively drop the traffic received by the host using our system.
3. Through puzzles, prevent the host server from being down.
4. Adjust the difficulty of the puzzle to enable effective rate limiting of the host.
5. Check the benefits of adding DNS server to system.

Table of Contents

Assumption

1. DNS server is trustworthy and does not participate in attacks.
2. For DNS server, only local DNS server and authoritative DNS server of target server exist.
3. Communication between target server and bot using TCP.
4. Other communication using UDP.

Puzzle

Puzzle

$$H(\text{Nonce},\ \text{Seed},\ \text{Source IP},\ \text{Local DNS IP}) \leq \text{Threshold}$$

- Nonce: solution, have to find
- Seed: given hash value for DNS
- Threshold: limiting factor

Step 1

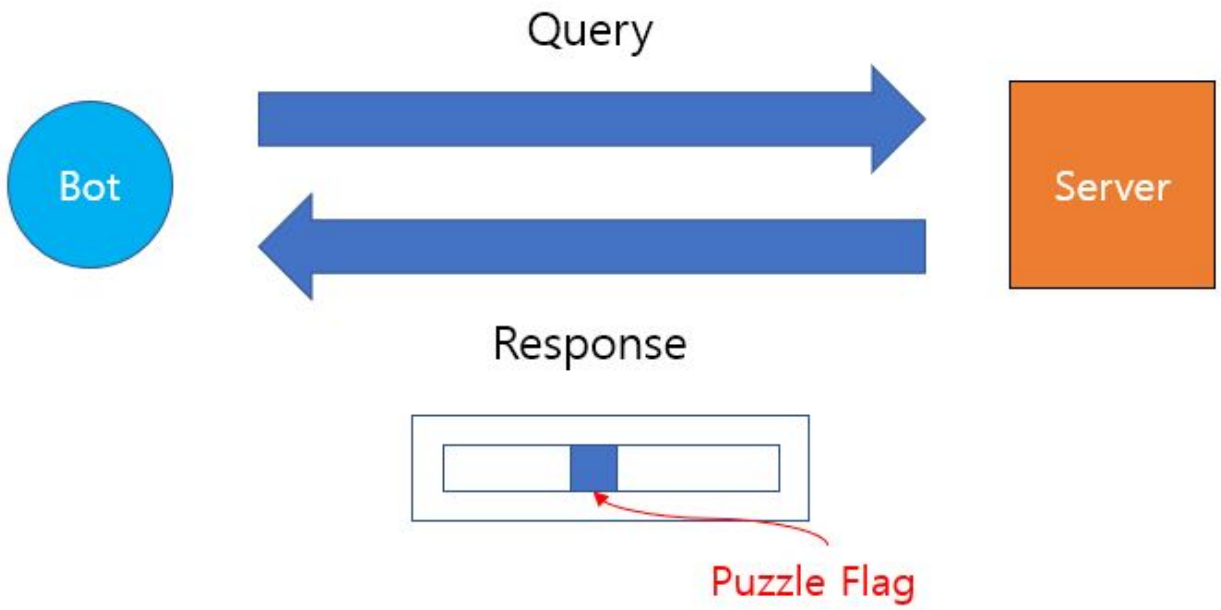

Figure 1: Step 1

Step 1. Target server detects DDoS attack and sets puzzle flag

Step 2

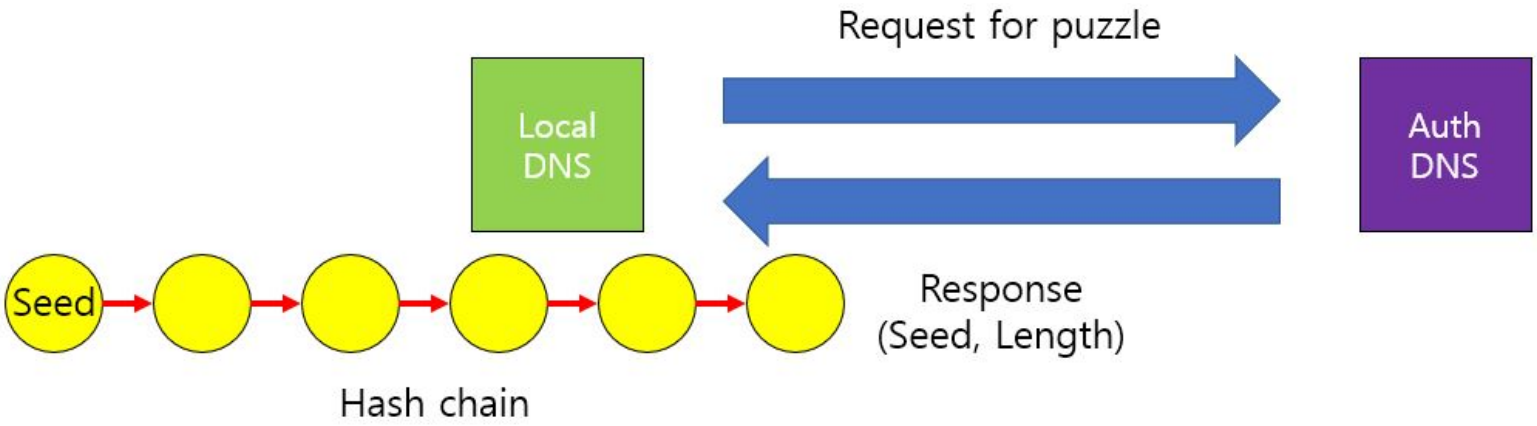

Figure 2: Step 2

Step 2. Local DNS server get seed and length from authoritative DNS server and generate

Step 3

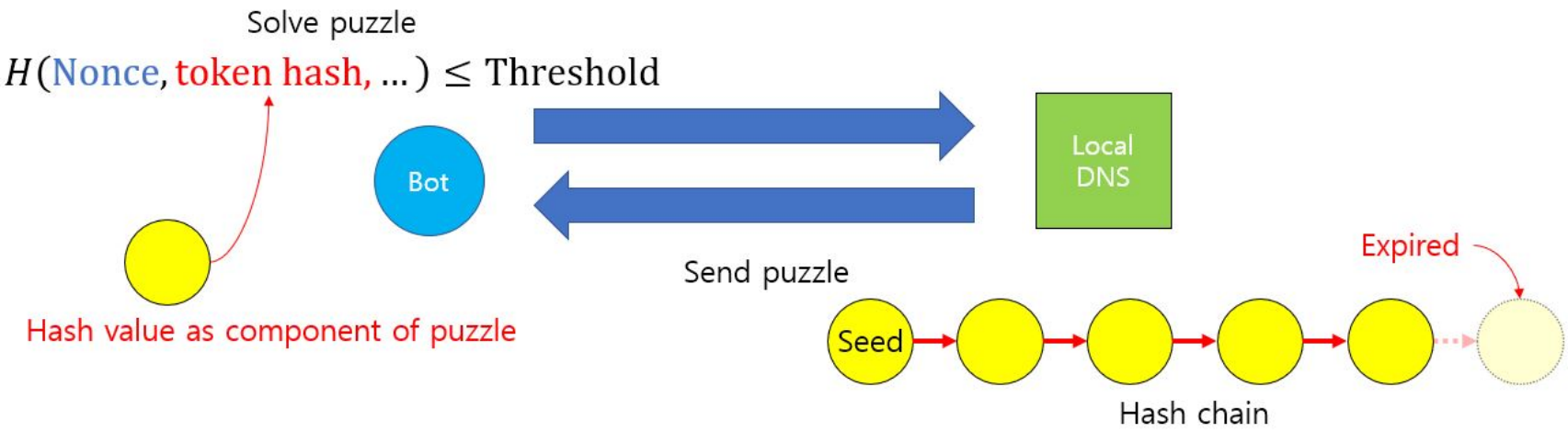

Figure 3: Step 3

Step 3. Local DNS server give seed value to user when user request information

Step 4

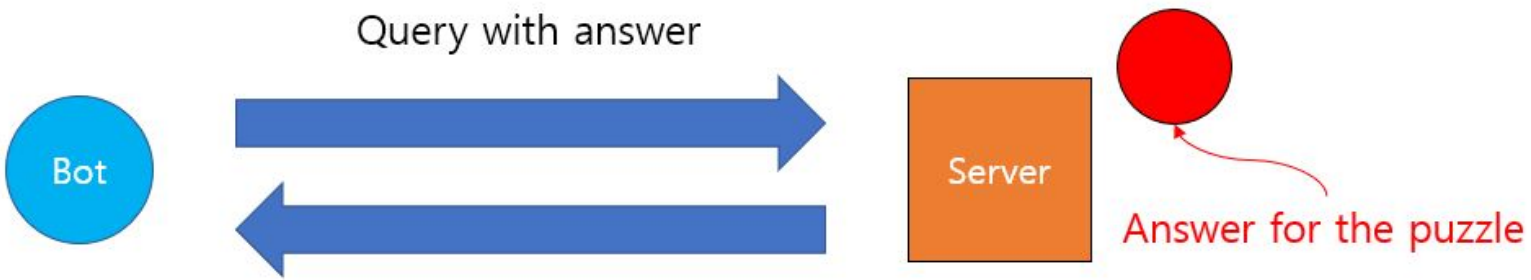

Figure 4: Step 4

Step 4. User solve the puzzle and send answer to target server

Step 5

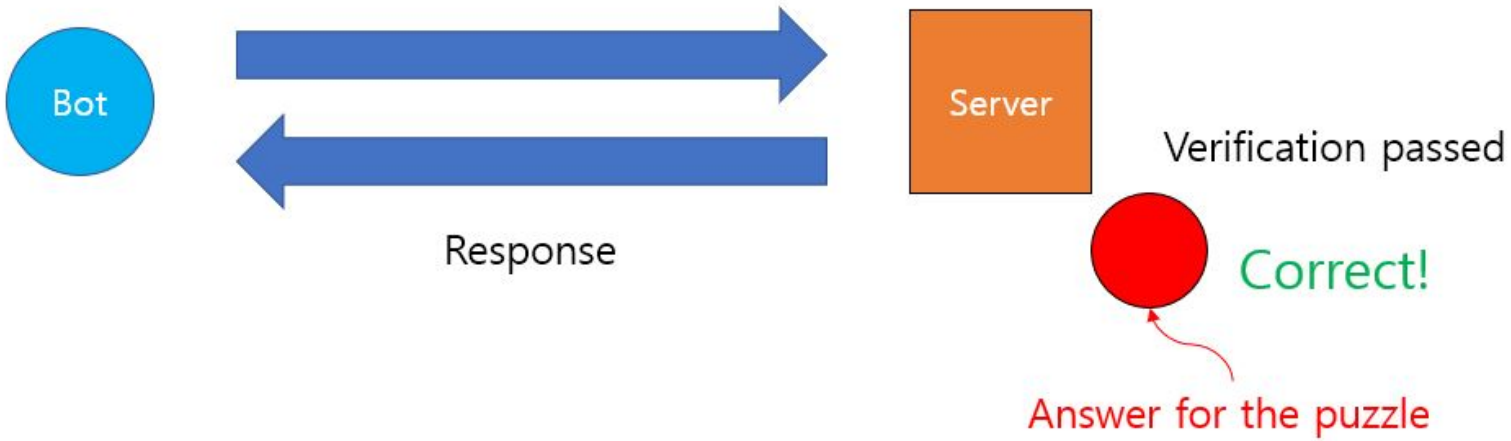

Figure 5: Step 5

Step 5. Query processing after checking whether the Puzzle and Solution are valid in the target server

Logical diagram

Below figure show whole logical steps for our DDoS mitigation system.

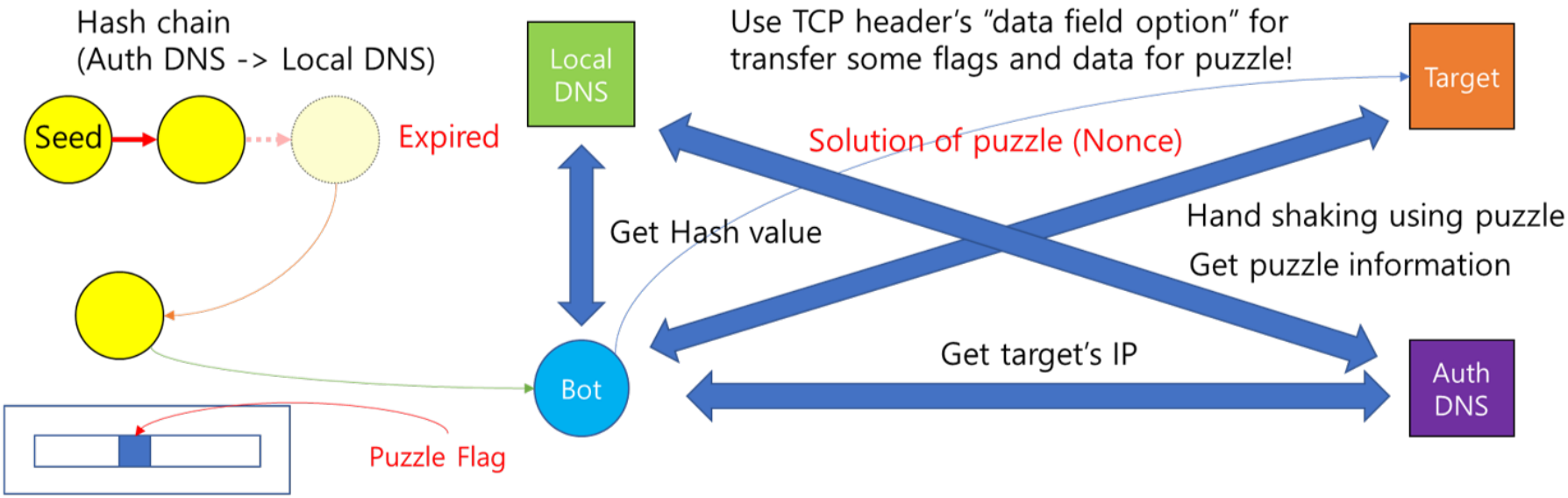

Figure 6: Logical diagram

Table of Contents

TCP handshaking

Slightly modified for transmission and verification of puzzle data in the TCP handshaking process.
In the figure on the right, the black line is the existing TCP handshaking process, and the red line is the newly added process.

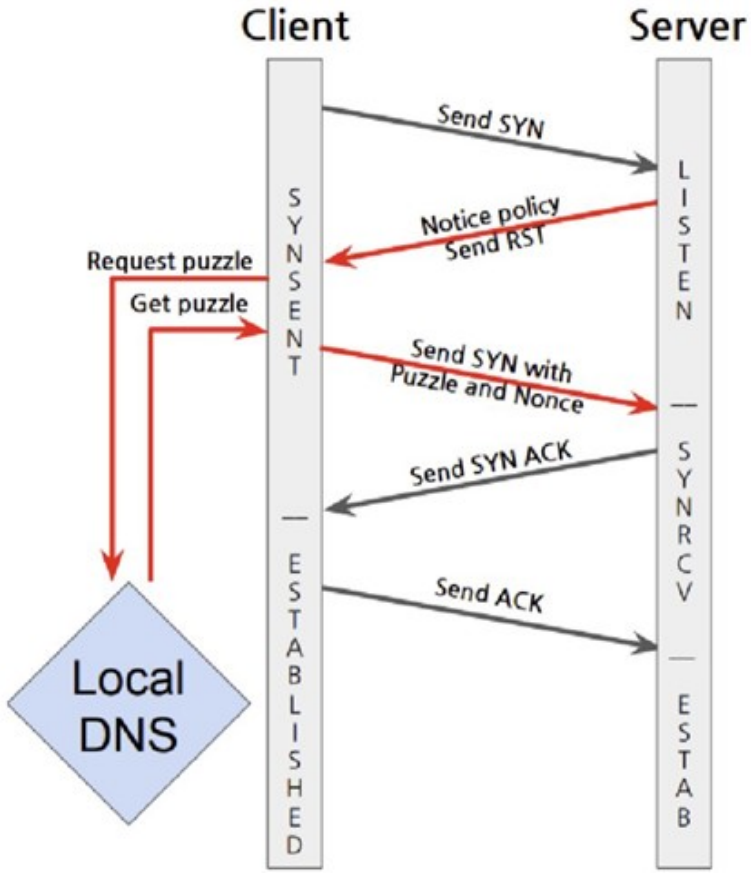

Figure 7: TCP modified handshaking

TCP header

TCP segment header

Offsets		0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0 0				CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	128	Checksum																Urgent pointer (if URG set)															
20	160	Options (if *data offset* > 5. Padded at the end with "0" bits if necessary.)																															
⋮	⋮																																
56	448																																

Figure 8: TCP segment header

By adding information on Puzzle Flag, Puzzle, Solution, etc. to the TCP header, the corresponding information is naturally transmitted during communication.
Assign 1 word to Token hash (Puzzle) and 1 word to nonce (Solution)

Linux kernel

Changed TCP communication process.
Modified below files. (omit details)

- net/ipv4/tcp_ipv4.c
- net/ipv4/ip_output.c
- net/ipv4/tcp_input.c
- net/ipv4/tcp_output.c
- net/puzzle.c

Whole codes are in github!

Implementation issues

1. Hooking using module
   ⇒ Unable to access tcp handshaking due to security regulations
2. Modify many functions and structures directly at linux kernel
   ⇒ Basically, it is call-by-value, so puzzle options are omited
3. In the case of TCP communication, this communication goes up to the IP layer, and rather, the values of IP header and buffer are used for actual communication.

Testbed

Use own testbed using raspberry pi as servers.

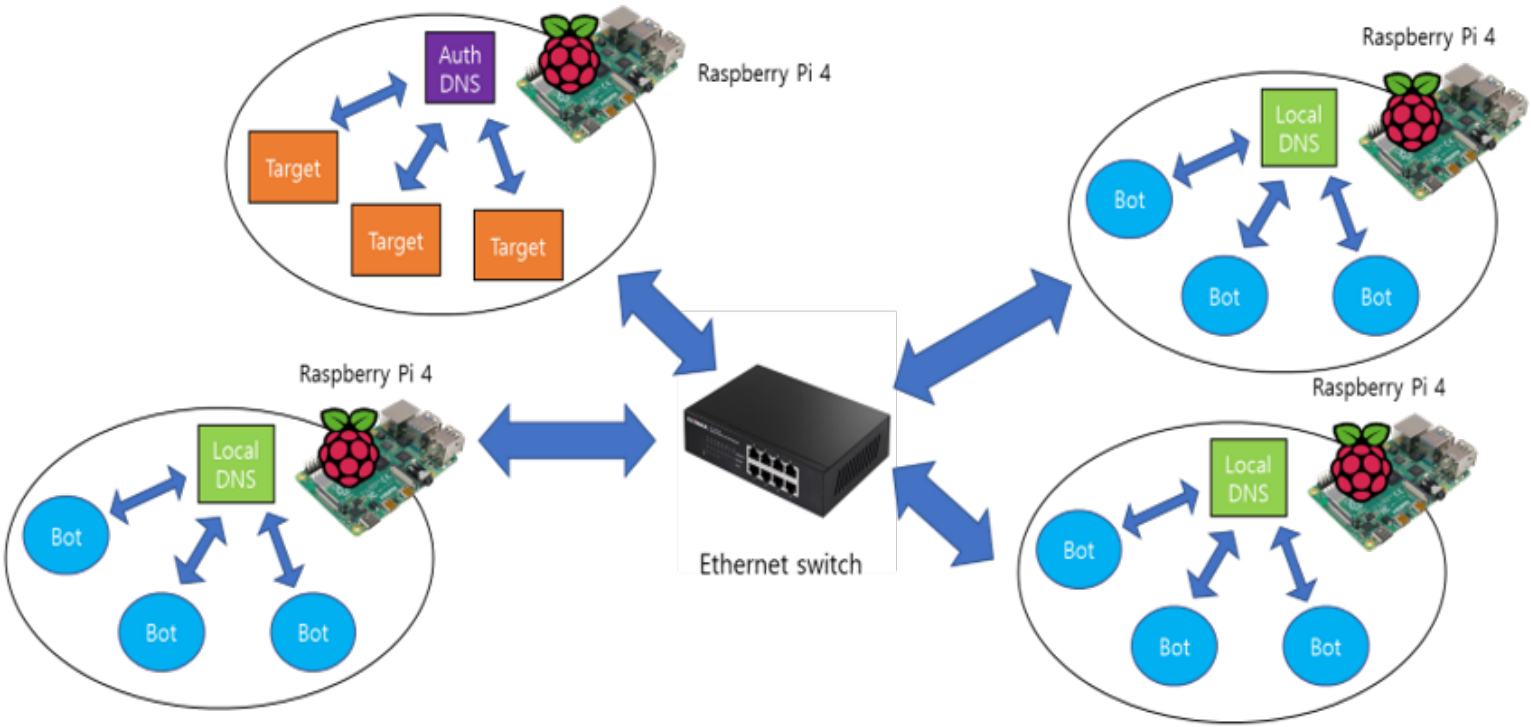

Figure 9: Testbed

Without puzzle policy

Using 1 client, packet/sec = 4249 packet/sec

Table 1: CPU resource with out puzzle

Number of client	CPU resources
1	1.235%
2	1.605%
3	2.0665%
12	6.8376%

Effect of puzzle

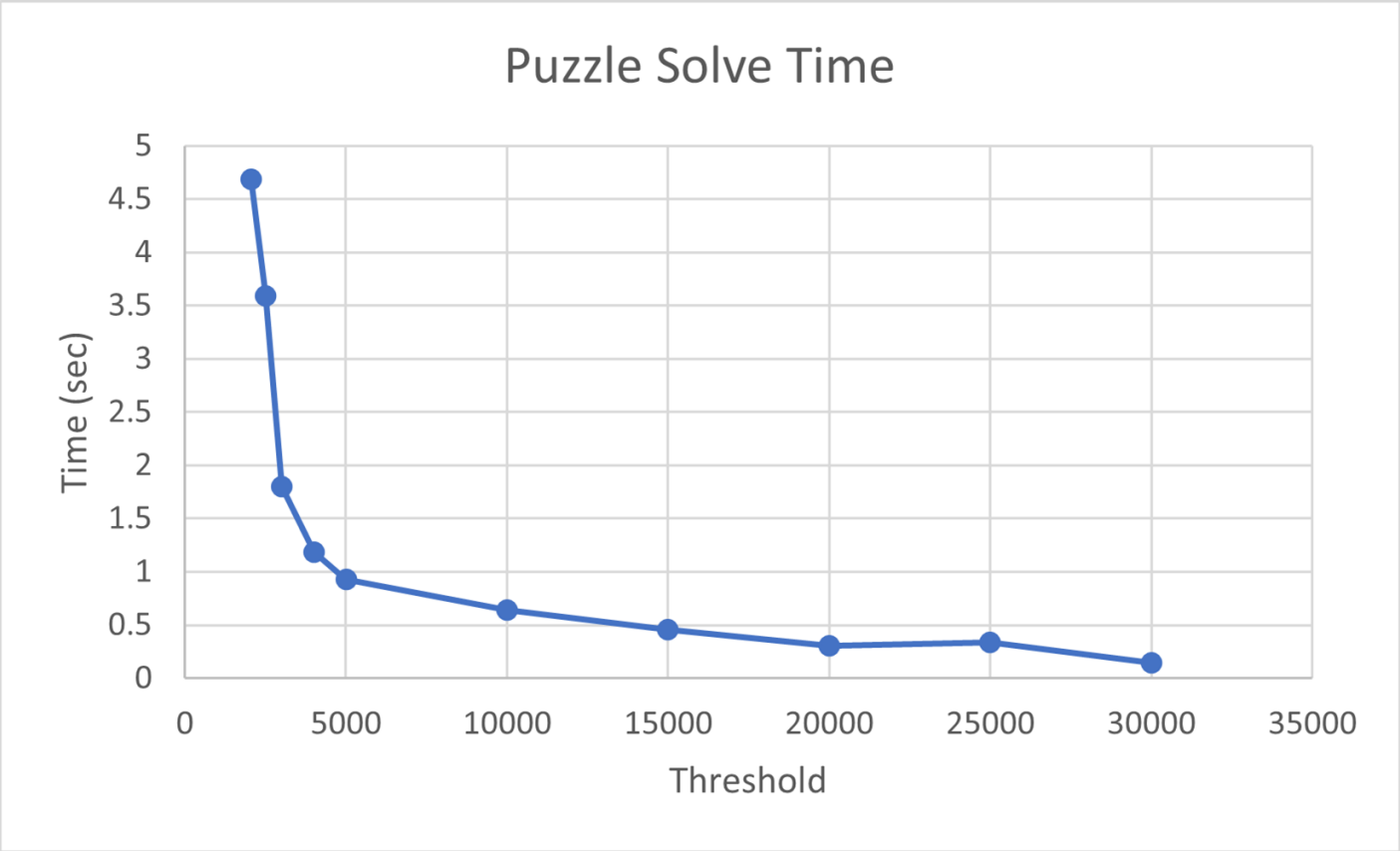

Figure 10: Puzzle solve time for each threshold

Puzzle dificulty at packet rate

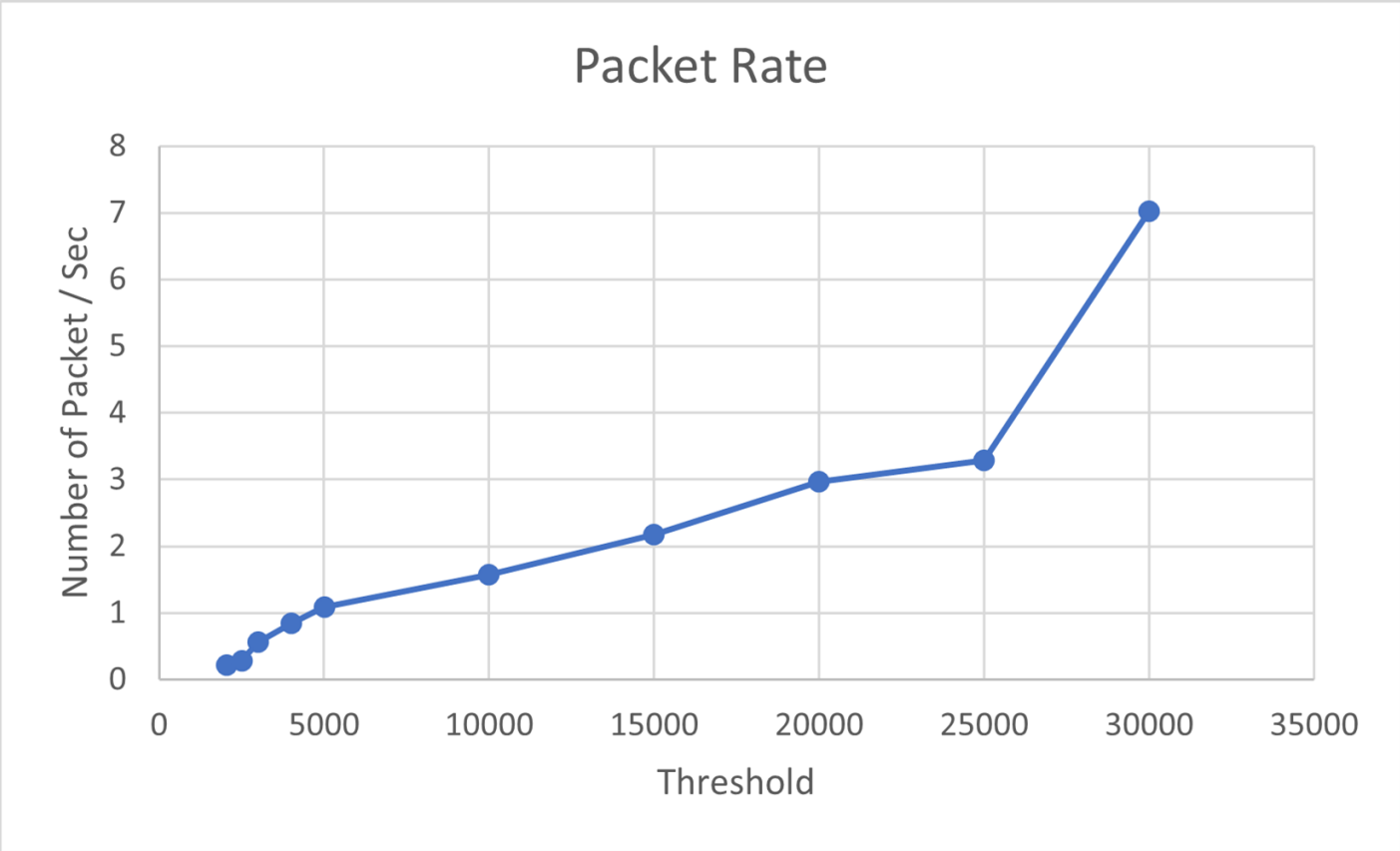

Figure 11: Packet rate for each threshold using 1 client

Puzzle dificulty at CPU resource

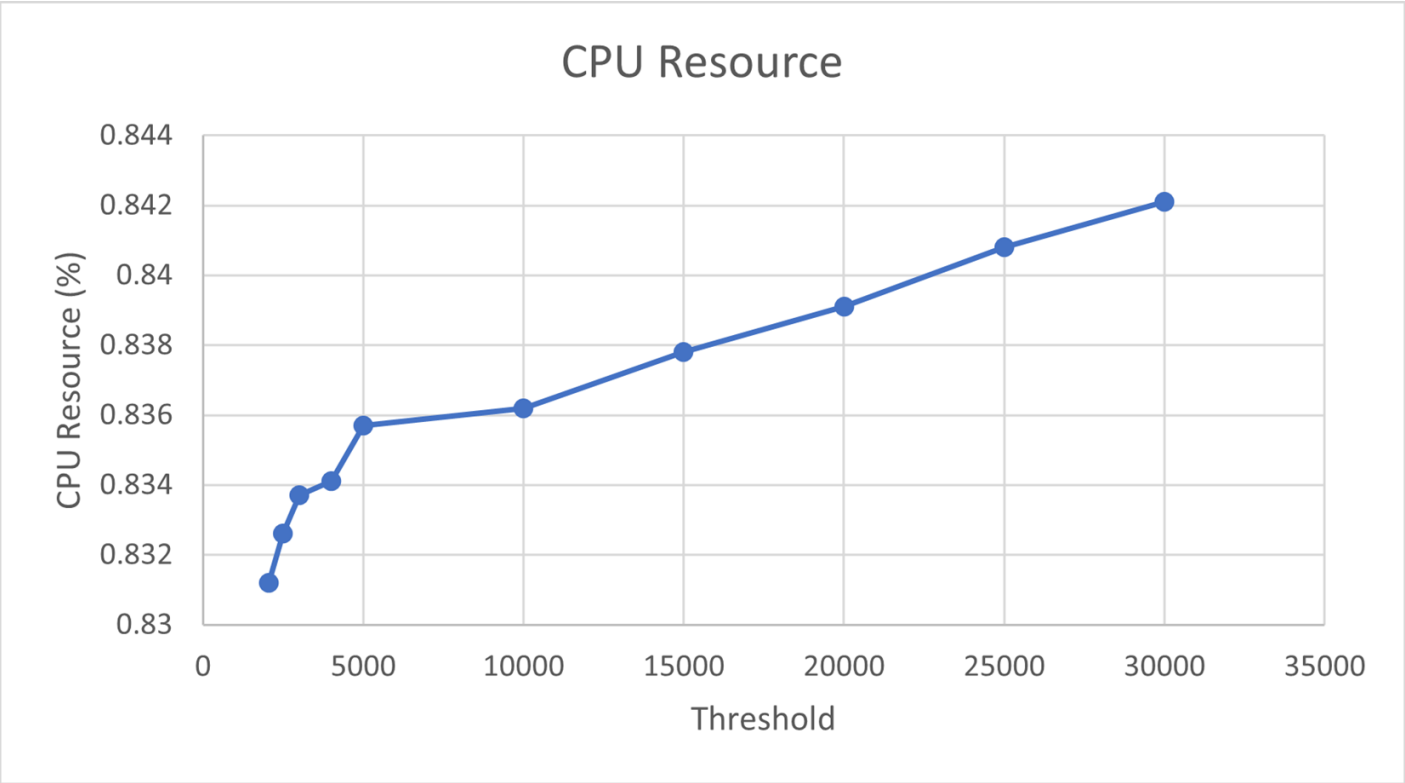

Figure 12: CPU resource for each threshold using 12 client

Comparison

Without puzzle policy: 6.8376% CPU
Use threshold as 30000: 0.844% CPU
$6.8376/0.844 \approx 8.10$!

With puzzle policy

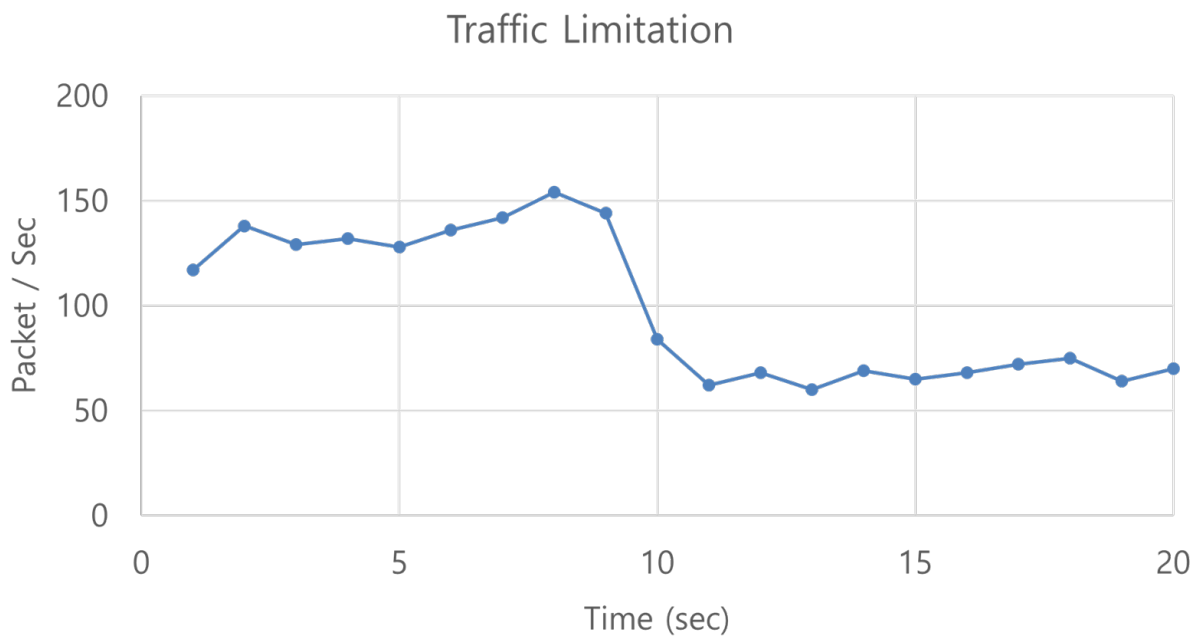

Figure 13: Traffic limitation using system call

We can prevent host die and limit traffic using this system!

Limitation

1. Due to limitations of TCP, this is not a complete kernel operation
2. UDP is done at the application level
3. Limited number of bots (raspberry pi)
4. Etc.

Table of Contents

Current status

- Finished CID2!
- Modifying the IP layer of the Linux kernel

Expected plan

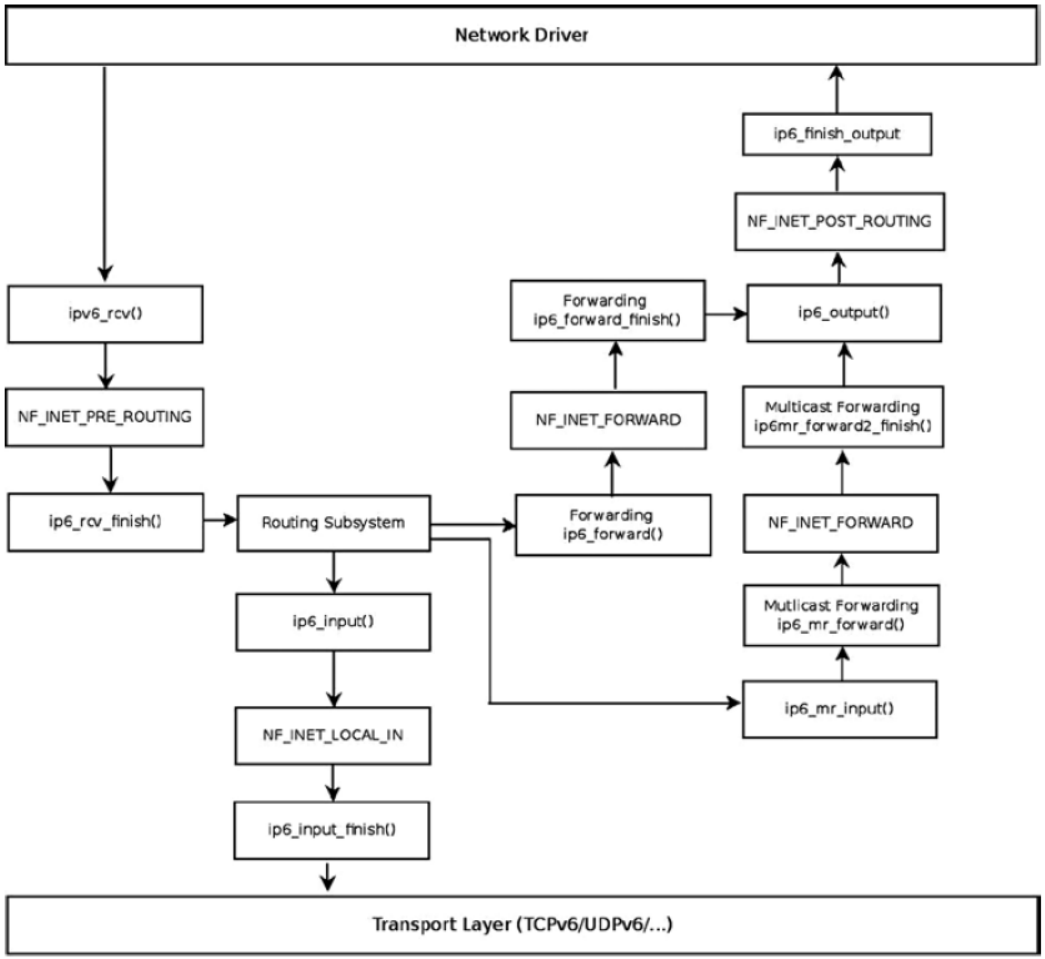

Figure 14: Linux kernel ipv4 packet flow

Expected issues

- Kernel panic and debugging
- Testbed (limited bots)
- Calculate effect of DNS
- Etc.

Future Plan

Quetions

- Correctness of IP layer?
- How to control UDP while TCP?
- How to calculate effect of DNS?
- Redesign testbed (limits?)
- Etc.

Helpful links

- https://www.youtube.com/watch?v=ia6vt7F9ONk
- https://www.youtube.com/watch?v=dnjCd09DRFc
- https://github.com/Sagit25/DDoS-DNS-Puzzle-CID2
- https://github.com/minjun0305/linux_for_practice

The end

Thank you for listening my presentation!