Denial of Services & Counter Measurements

Janneth Malibago <jannethm@student.ethz.ch>
Seminar of Distributed Computing WS 04/05

Papers

[1] Internet Indirection Infrastructure
I.Stoica, D. Adkins, S. Zhuang,
S. Shenker, S. Surana; SIGCOMM 2002

[2] Taming IP Packet Flooding Attacks
Daniel Adkins, Karthik Lakshminarayanan,
Adrian Perrig, Ion Stoica (UC Berkeley and
CMU); HotNets 2003

Overview

- Motivation
- Useful defenses against packet flooding
- i3-based approach / i3 summarization
- IP-based approach
- i3-based approach vs. IP-based approach
- Summary / Conclusion

Motivation (1)

- Denial-of-service (DoS) caused by IP packet floods
  - One of the major problems faced by Internet hosts
  - Hosts in the Internet are unable to stop packets addressed to them
  - IP routers respond to the overload by dropping packets arbitrarily

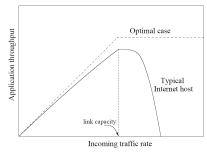


Figure 1: Application throughput as a function of incoming traffic rate for (a) typical Internet host and (b) optimal case.

Motivation (2)

- A host could respond more effectively to overload if it had control over which packets were dropped
  - reject new connections rather than accept excess load
  - give higher priority to some services than others (service differentiation)
  - provide lower quality service rather than reject requests (service degradation)

Main thesis of paper [2]

- Hosts – not the network – should be given control to respond to packet floods and overload
  - Fine-grained control over how routers process the packets addressed to the host
  - Ability to decide which packets to receive, which packets are dropped, and which packets are redirected

Why hosts should be given control?

Why not implement more sophisticated drop policies at routers instead?

- Hosts inherently have more information about the type and the importance of the traffic they receive than the network does

→hosts are in the best position to respond to IP flooding attacks

Why hosts should be given control? Example (1)

- Example:
  - Consider a host that runs two services A and B
  - Assume that the traffic to service B surges abruptly causing congestion on the incoming link
  - The best possible response to this event may depend on knowledge available only to the host

Why hosts should be given control? Example (2)

a)

- Service B has higher priority than A
- The host believes that the surge is because of a flash crowd (e.g., B is a web server that has just announced a new popular product),

→ The host may decide to stop the traffic of the less important service A

b)

- A is the more important service
- The host believes that the surge is due to a DDoS (Distributed Denial of Service) attack

→ The host may choose to stop the traffic of B

Why hosts should be given control? Example (3)

- The traffic in the two cases appears to be the same to the network

→ Impossible for the network to have an optimal response to the congestion without input from the host!

Useful defenses against packet flooding

1. Avoid receiving packets at arbitrary ports
2. Contain the traffic of an application (service) under a flooding attack to protect the traffic of other applications
3. Protect the traffic of established connections
4. Throttle the rate at which new connections are opened

1. Avoid receiving packets at arbitrary ports

- Internet hosts can receive packets they did not ask for at ports where no service runs
- Though these packets are dropped by the kernel, they consume network bandwidth and may affect other services

→A host should receive packets only at ports on which it is listening or as part of an established connection

- prevents arbitrary scanning of networks and also illegitimate packets sent to random ports

2. Contain the traffic of an application

- With the ability to decide which packets are dropped, hosts can contain the traffic of individual applications that might be under a flooding attack

→ protecting other applications that run on the same host

3. Protect the traffic of established connections

- To maximize the application throughput, hosts need to protect the traffic of established connections against arbitrary traffic

→ More difficult for an attacker to perform IP flooding attacks

- harder to establish a connection and sustain the traffic on that connection rather than send arbitrary packets
- establishing a connection requires the attacker to handle data and signaling packets (e.g., ACK, SYN ACK packets) from the victim

4. Throttle the rate at which new connections are opened (1)

- Previous defenses protect the established connections
- But: still difficult for legitimate clients to open new connections in the presence of DoS attacks

→ A host under attack should be able to reduce the fraction of connection attempts made by the attacker e.g. by using cryptographic puzzles or captchas

(captchas: tests to distinguish between humans and computers)

4. Throttle the rate at which new connections are opened (2)

- This approach will throttle the rate of connection setup of all clients
- But: much greater effect on the attacker than on legitimate clients

(a legitimate client opens, in general, far fewer connections than an attacking host does)

Two possible realizations of the above defenses

2. i3-based approach

   (Internet Indirection Infrastructure)

5. IP-based approach

i3-based approach

What is i3?

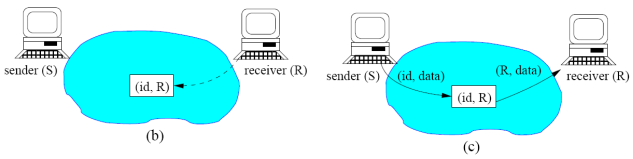
i3 summarization (1)

- Overlay-based Internet Indirection Infrastructure (i3) that offers a rendezvous-based communication abstraction
  - Purpose of i3: provide indirection
    → decouples the act of sending from the act of receiving
  - Applications can easily implement a variety of communication services on top of this communication abstraction
    - multicast, anycast, mobility, ...

i3 summarization (2)

- Service Model:
  - Sources send packets to a logical *identifier*
  - Receivers express interest in packets by inserting a *trigger* into the network
  - Packets are of the form (id,data)
  - Triggers are of the form (id,address), where addr is either an identifier or an IP address.
  - Given a packet (id,data), i3 will search for a trigger (id,addr) and forward data to addr
    → logical rendezvous
  - If a host wants to stop receiving packets from a particular trigger, it can simply remove that trigger

i3 summarization (3)



(b) The receiver R inserts trigger (id;R)
(c) The sender sends packet (id;data)

i3 summarization (4)

- Client-server communication*:*
  - Servers that expect connections from arbitrary clients must have triggers whose identifiers are well-known → *public* triggers
  - Once a client contacts a server through its public trigger, they exchange a pair of identifiers which they use for the remainder of the communication → *private* triggers
- Variety of communication services
  - Mobility, Multicast, Anycast, Service Composition, …
  - On top of the communication abstraction

i3 summarization (5)

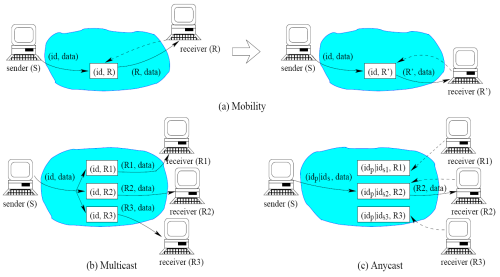


Figure 2: Communication abstractions provided by i3. (a) Mobility: The change of the receiver's address from R to R' is transparent to the sender. (b) Multicast: Every packet $(id, data)$ is forwarded to each receiver R_i that inserts the trigger (id, R_i). (c) Anycast: The packet matches the trigger of receiver $R2$. $id_p|id_s$ denotes an identifier of size m, where id_p represents the prefix of the k most significant bits, and id_s represents the suffix of the $m-k$ least significant bits.

i3 summarization (6)

Figure 4: (a) Service composition: The sender (S) specifies that packets should be transcoded at server T before being delivered to the destination (R). (b) Heterogeneous multicast: Receiver $R1$ specifies that wants to receive H.263 data, while $R2$ specifies that wants to receive MPEG data. The sender sends MPEG data.

i3 summarization (7)

- Summary:
  - i3 provides a general-purpose indirection service through a single overlay infrastructure
  - Prototype based on the Chord lookup protocol
  - Promising simulation results, but the details of the design are still preliminary!
  - → Need to gain more experiences with using/deploying new applications on top of i3!
- Too many details to cover here!
  - → see paper [1] or http://i3.cs.berkeley.edu
- Here:
  - only discussion of i3 as a solution to solve Denial-of-service (DoS) problems caused by IP packet floods

Two possible realizations of the above defenses

2. i3-based approach

   (Internet Indirection Infrastructure)

5. IP-based approach

Useful defenses against packet flooding – i3 based approach

1. Avoid receiving packets at arbitrary ports.
2. Contain the traffic of an application (service) under a flooding attack to protect the traffic of other applications.
3. Protect the traffic of established connections.
4. Throttle the rate at which new connections are opened.

1. Avoid receiving packets at arbitrary ports

- Clients in i3 can hide their IP addresses and publish the identifiers of only their public triggers
  - We assume here that it is very hard for the attacker to find the IP address by other means

2. Contain the traffic of an application

- The traffic of different applications can be distinguished from one another by the triggers used for communication
- Example:
  - Each application could have a different public trigger, or each client contacting a server could do so with a different private trigger
  - To contain the traffic of an application, we could associate a drop probability with each trigger
  - If an application is attacked or becomes overloaded, we could raise the drop probability of its triggers to reduce its traffic
  - If necessary, we could disconnect the application entirely by setting the drop probability to one or by removing its triggers

3. Protect the traffic of established connections

- In i3 a client can send packets to a host only using the host's public triggers or the private triggers corresponding to the client's connections
- The host can protect the traffic of its established connections by dropping some of the packets destined to the public triggers
  - → a host while maintaining m of its n public triggers can choose an appropriate value of m such that the traffic of its established connections is not affected

4. Throttle the rate at which new connections are opened (1)

- Consider a server S that is under a flooding attack. S can use indirection to redirect traffic to a gatekeeper (gatekeeper: powerful third party server, which shields the server S from the attack)
- The gatekeeper gives cryptographic puzzles to the client which have to be solved in order to contact the server
  - This will considerably slow down attacking hosts that attempts to open a large number of connections
  - In contrast, the impact this has on a typical client which opens very few connections will be small

4. Throttle the rate at which new connections are opened (2)

- Implementation:

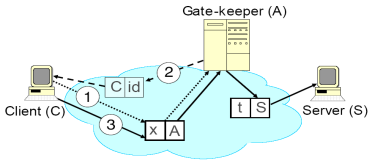


Figure 3: Slowing down a DoS attack on public triggers. When a client C wants to contact a server S, it must first solve a puzzle before the gatekeeper A will forward its packet to S.

- Note:
  these schemes adopted by servers only when under attack → under normal operation, clients will not have the burden of either solving cryptographic puzzles or trying multiple times to reach a server

Two possible realizations of the above defenses

2. i3-based approach

   (Internet Indirection Infrastructure)

5. IP-based approach

IP-based approach (1)

- Basic Idea:
  - Provide a configurable white list of allowed ports at the edge-router directly connected to the hosts.
  - Configuration settings include which ports to open, the rate at which bandwidth needs to be shared across different ports, etc.
  - Edge-routers that are directly connected to hosts need to maintain per-flow state on behalf of the hosts i.e., if R is the edge-router through which all packets destined for S must pass, then R maintains per-flow state for S

IP-based approach (2)

- Assumptions:
  1. The edge ISP (Internet Service Provider) is better provisioned than the host so that it may sustain attack traffic
  2. The ISP is willing to install filters on the host's behalf
  3. ISP filters must be modified to enable the port that the server runs on to allow incoming traffic
  4. Unmodified clients would be able to connect to the servers in the normal case, but may need to do special work (like extra computation of cryptographic puzzles) when the server they contact is experiencing a flooding attack

Useful defenses against packet flooding – IP based approach

1. Avoid receiving packets at arbitrary ports.
2. Contain the traffic of an application (service) under a flooding attack to protect the traffic of other applications.
3. Protect the traffic of established connections.
4. Throttle the rate at which new connections are opened.

1. Avoid receiving packets at arbitrary ports

- S instructs R to allow traffic on certain public ports only
- Once a client C establishes a connection to S (by using a port that is white listed by S in R), R will maintain state to allow C's packets through to S
- When the connection is terminated, R removes the associated state
- S also has the power to stop malicious clients by terminating their connections

2. Contain the traffic of an application

- S can specify how exactly to split the bandwidth among its various applications
- This functionality is similar to traffic shaping that many routers already implement

3. Protect the traffic of established connections

- S can ask R to reserve a fraction of S's bandwidth for established connections
- Under congestion, R will shape traffic according to the rules that S has specified
- R will limit the rate of packets to S's public ports in order to ensure that S's ongoing connections will receive their reserved bandwidth

4. Throttle the rate at which new connections are opened (1)

- Consider a server S that is under a flooding attack. S can use indirection to redirect traffic to a gatekeeper
- The gatekeeper gives cryptographic puzzles to the client which have to be solved in order to contact the server

→ same principle as in i3-based solution

4. Throttle the rate at which new connections are opened (2)

- Implementation:
  - For redirecting traffic to gatekeepers, one can use DNS to send the traffic to the gatekeeper
  - In fact, the edge routers, if modified further, can themselves act as gatekeepers

i3-based approach vs. IP-based approach

- Generality
  - i3-based solution:
    - General and architecturally clean solution
    - The indirection primitive gives an elegant way of redirecting traffic seamlessly to a third party which would require DNS hacks for implementing in IP
  - IP-based solution:
    - the use of IP addresses combined with port numbers to identify services running on a host is not general enough

i3-based approach vs. IP-based approach

- Deployability
  - i3-based solution:
    - assumes the existence of an infrastructure such as i3
  - IP-based solution:
    - requires changing the edge router of the ISP that provides service to the host
    - assumes that the ISP is willing to cooperate with the hosts by allowing them to install filters into the ISP network
    - incrementally deployable in the Internet. Whether ISPs would allow this is a separate issue

Summary

- Benefits when hosts are given control:
  - Victims of DoS attacks can start defending themselves
  - Hosts without server functionalities cannot be attacked by arbitrary attackers any more
    (host would enable packets only those connections that it has established)
  - Only servers that can be contacted by arbitrary hosts need a rendezvous mechanism

→ the solutions provided in paper [2] constrain attackers to attack through that narrow interface, thus protecting the ongoing connections

Conclusion

- However, there are still some questions that remain open:
  - How much control is necessary for hosts and at what cost will this control come?
  - The two proposed approaches help hosts cope with packet floods directed at them, but do not protect the network itself
  - Ultimately, we need to identify the source of DDoS attacks and stop them at the entry points in the network

→It is a challenge to design an indirection layer which is itself robust to DoS attacks

Questions