A Practical Method to Counteract Denial of Service Attacks

Udaya Kiran Tupakula   **Vijay Varadharajan**

Information and Networked System Security Research
Division of Information and Communication Sciences
Macquarie University
Sydney, Australia

{udaya, vijay}@ics.mq.edu.au

Abstract

Today distributed denial of service (DDoS) attacks are causing major problems to conduct online business over the Internet. Recently several schemes have been proposed on how to prevent some of these attacks, but they suffer from a range of problems, some of them being impractical and others not being effective against these attacks. In this paper, we propose a Controller-Agent model that would greatly minimize DDoS attacks on Internet. With a new packet marking technique and agent design our scheme is able to identify the approximate source of attack (nearest router) with a single packet even in case of attack with spoofed source addresses. Our scheme is invoked only during attack times, is able to process the victims traffic separately without disturbing other traffic, is able to establish different attack signatures for different attacking sources, can prevent the attack traffic at the nearest router to the attacking system, has fast response time, is simple in its implementation and can be incrementally deployed. Hence we believe that the scheme proposed in this paper seems to be a promising approach to prevent distributed denial of service attacks

Keywords: Denial of Service, DoS, Controller-Agent Model, Packet Marking, Broad Attack Signatures.

1 Introduction

Denial of service attacks on the Internet has become a pressing issue following a series of attacks during recent times. Recently, several papers [CER00, MVS01, How98] have reported on the prevalence of such attacks on commercial servers and ISP's and the disruption they cause to services in today's Internet. A "denial-of-service" (DoS) [CER00, CER99, DD99, DO99] is an attempt by attackers to prevent access to resources by legitimate users for which they have authorization. In case of DDoS, an attacker compromises several hosts on the Internet. Often, these are weakly secured machines and they are broken into using well-known defects in standard network service programs and common operating systems. These compromised computers are referred to as zombies. The attacker then uses these compromised computers to launch coordinated attacks on victim machines. Further these attacks make use of the inherent weaknesses in the network protocol [Bel89]. Technologies, such as cable modems to home users, have further increased the threat of DDoS attacks. This is because with the cable modems the home users are always connected to the Internet and it is easier for an attacker to compromise these systems, which often have weak security. It would be even more difficult to prevent such attacks if the compromised systems attack with spoofed source address and constantly change the attack traffic pattern.

Over the recent years, several schemes have been proposed to deal with the distributed denial of service problem. For instance, [FS98, SAN00] describe some techniques for preventing traffic with spoofed address from crossing the border, while some others [Bel00, DFS01, SP00, SWKA00, BC00] have proposed techniques to trace back the approximate source of the attacking system even in the case of spoofed source address. Another paper [MBFIPS01] proposes technique to automatically identify the attack at the congested router and treats the traffic by punishing only the identified aggregates. In this paper, we propose a model that identifies the attack at the victim, prevents the attack near to the attacking system (rather than the attacked system) and quickly responds to the changes in attack traffic. In other words, our model offers several of the required features to minimize the DDoS attacks on the Internet. We propose a new packet marking technique and agent design to identify the approximate source (nearest router) of the attack with a single packet. In this paper, we will give a detailed description of our model, how it achieves its aims in minimizing the DDoS attacks and how the model is implemented in practice. We will also give a detailed comparison of our model with the previously proposed techniques.

This paper is organized as follows. Section 2 describes the characteristics of a robust DDoS solution. Section 3 gives the current state of the art in DDoS attacks and describes some of the most important work done to date in this area. Section 4 discusses our approach, implementation of different aspects in our model, system operation and an example comparing our scheme with previously proposed ones. Section 5 discusses the additional characteristics of our proposed scheme and finally section 6 gives the concluding remarks.

2 Characteristics of a Robust DDoS

Before we can propose any solution to thwart DDoS attacks, it is first useful to identify and examine the essential characteristics of a robust DDoS solution architecture. Consider Fig: 1 below, where the attacker has already done a substantial amount of work in creating the zombies. So it is only a matter of few keystrokes for the attacker to launch a severe DDoS attack [DD99, DO99]. If the victim is not equipped to fight this back upstream, s/he can only prevent the attack at the boundary of his network if s/he has some firewall or intrusion detection system. But the regular benign traffic to the victim's network is not protected and more over the victim cannot have access to other networks (e.g. Internet).

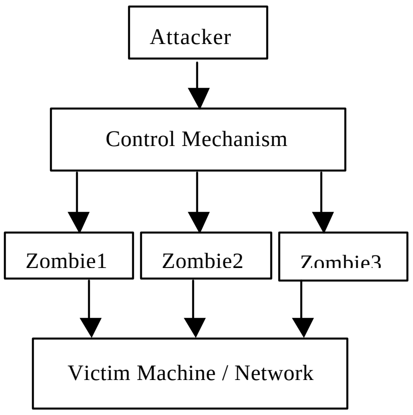


Fig 1: Attack Mechanism

An effective approach against these attacks would be to trace the attacker and prevent him/her from commanding the zombies to attack the particular host or network. However this is not possible with the present available technology because often the zombies are controlled by an attack control mechanism, which is remotely controlled by the attacker. Moreover the communication between the zombies, control mechanism and the attacker can be encrypted [CER99, DD99]. Alternatively the victim should have a mechanism such as the one shown in Fig: 2 in order to prevent the attack effectively. In this case, the victim identifies the attack at the point where it occurs but prevents the attack nearest to the attacking source.

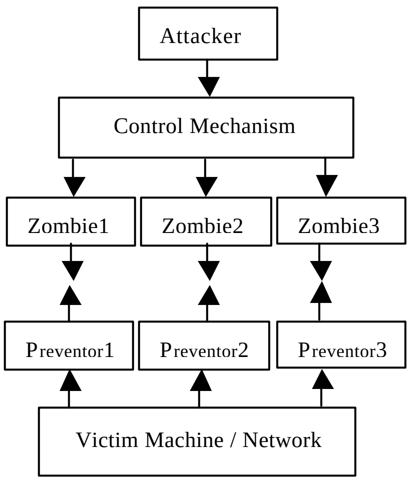


Fig 2: Robust Solution

We believe that an ideal effective model against distributed denial of service attacks should at least have the following characteristics:

- It should be invoked only during the attack times and at other times it must allow the system to work normally.
- It must provide simple, easy and effective solution to counteract the attacking sources in preventing the attack.
- It should identify the attack at the victim and prevent the attack at the attacking source.
- It should prevent only the attack traffic from reaching victim. That is, the model should be able to differentiate a malicious traffic flow from a regular benign flow by incorporating different attack signatures for different attacking sources.
- It should have fast response time and should respond quickly to any changes in attack traffic.
- It should provide mechanisms for retaining the attack evidence for any future legal proceedings.

However with the present technology, implementation of such a model will be difficult in practice due to several factors such as

- Attackers use spoofed source addresses. Hence tracing the attacking system is difficult and would consume a lot of time.
- Even if the approximate source can be traced, we cannot prevent the attack traffic at that point because the systems are randomly distributed over the Internet and may be under the control of different ISP's. Hence different ISP's have to cooperate for the solution to be effective. There could be hundreds of systems under the jurisdiction of different ISP's, where the filters need to be placed.
- Even if the above were done, as the process is not fully automated, if the attacking systems change the attack traffic pattern, reconfiguring the filters in these systems with different ISP's would be an onerous task.

3 Related Work

In this section, we discuss the most important previously proposed schemes and outline their advantages and disadvantages.

3.1 Ingress/ Egress Filtering

Perhaps the most standard approach against denial of service attacks is the use of Ingress/Egress filtering techniques [FS98, SAN00]. Ingress filtering is from the point of view of the Internet. Here an Internet Service Provider (ISP) filters out packets with illegitimate source address, based on the ingress link by which the packet enters the network. In contrast, egress filtering is from the point of view of the customer network and the filtering occurs at the exit point of a customer domain. Here a router checks whether the source addresses of packets actually belong to the customer's domain. Packets with invalid source addresses are dropped. It is similar to a postman checking all the return address on the letters at a local post office and drops the letters that have a different

PIN code to that of the local one on the return address. While this is considered to be a first step to prevent DoS attacks with spoofed source address, it is not likely to completely eliminate the ability to spoof source IP address. For instance, the attacker can still forge his source address with an address that is within the customer domain. The main disadvantages of this approach include the extra overhead and the associated decrease in performance. It is necessary to check every packet that flows in or out of the network. It may require upgrade of existing hardware as not all routers support this type of filtering. Furthermore, this technique is not effective against compromised machines with valid source addresses. Perhaps the major stumbling block is the unwillingness of many ISP's to provide this feature.

3.2 Center Track Approach

In Stones [Sto00] approach, an overlay network is created by deploying special tracking routers which links all the edge routers to a central tracking router. During an attack, the traffic to the victim is routed through the overlay network by dynamic routing. Then starting from the tracking router closer to victim, hop-by-hop tracking is used to trace back the ingress point of the attacking source. This reduces the number of hops required to trace back to the approximate source of attack. The main disadvantage of this approach is the changes required to the global routing table. Also a poor implementation of this model would make it too easy for a small error to severely disturb the total network. The attack traffic originating from the backbone cannot be prevented.

Stone [Sto00] also proposes a second approach where all the edge routers maintain the database of all the traffic passing through them. The data should at least include the source address, destination address, adjacency and approximate number of packets. When an attack is reported, the database is searched based on the attack signature and the ingress adjacency is determined. The advantage is that no tracking hops are required to trace the ingress edge and the tracing is possible even after the attack is completed. However this approach has a high overhead of storage and processing, as the entire traffic database has to be maintained.

3.3 Trace Back Approaches

One of the major difficulties associated with tracing a packet stream back to the source is that it requires cooperation of all the intervening ISP's. This is complicated by the potential use of indirection to conceal the true origin of an attack. The approach proposed in [SWKA00] probabilistically marks the packets with partial path information as they arrive at routers. It makes the assumption that there would be a considerable amount of packets generated during an attack. They argue that a victim will be able to reconstruct the entire route the packets followed by combining a modest number of marked packets, though each marked packet represents only a sample of the path it has traversed. Bellovin [Bel00] suggests a similar approach but with ICMP traceback. In this approach, when forwarding packets, routers send an ICMP packet to the destination with a low probability (say 1/20,000). The ICMP traceback includes the identity of the router, contents of the packets, and information about the adjacent routers. The approach of Burch and Cheswick [BC00] starts by creating a map of the routes from the victim to every network, using any known mapping technology. Then, starting with the closest router, a brief burst of load is applied to each link attached to it, using the UDP *chargen* service. If the loaded link is a component of the path of the attacking stream, then the induced load will change the attacking stream traffic. Then this link is considered to be along the path to the attacker and this is carried until the nearest source is reached. If there is no change in the traffic then different path is chosen. The main disadvantage of these approaches is that even if the route between the source and victim could be traced, no action is taken. Furthermore, routers in between the source and the victim may be compromised. Also the attacker may send some faulty trace back messages. Even more challenges lie in the tracing of the routes if the attack is distributed.

3.4 Controlling High Bandwidth Aggregates

The idea behind the approach proposed by [MBFIPS01] is to identify the particular set of packets that are causing overload. These are referred to as aggregates. An aggregate is defined as a collection of packets from one or more flows that have some property in common. For instance, "packets to destination X", "TCP SYN packets" and "IP packets with a bad checksum" are examples of aggregates. The aim is to identify aggregates responsible for congestion and prevent them getting to the victim. Prevention can be done at the victim or at the router where there is congestion. Prevention is implemented using two mechanisms: one using a local aggregate based congestion control (ACC) scheme in which a router deals with sustained overload by itself, and other referred to as a pushback scheme which extends the local ACC to other routers upstream. The ACC itself can be broken into two phases namely detection and control. In the detection phase, the ACC agent is responsible for identifying the aggregate and calculating the rate limit for them. The rate limit in the control phase determines whether a packet is to be discarded or forwarded. The advantage is that the aggregates can be completely stopped or limited to a particular bandwidth. In some sense, pushback is similar to traceroute. Once the aggregates are identified the router can ask the upstream routers to prevent it and hence reach the source of the attacker. When pushback is applied upstream, there is more bandwidth for legitimate traffic in downstream routers. The disadvantages associated with this approach are strong authentication is required to perform a pushback; otherwise this in itself could turn out to be a DoS attack. Furthermore, even if strong authentication is implemented, if the attacker can compromise a single ACC implemented router, then s/he can cause severe damage.

4 Our Approach

We consider an architecture similar to [Sto00] where the network is divided into ISP domains and customers connected to these ISP domains. In general, there are two types of routers in an ISP domain: internal routers and external routers. Internal routers belong to the ISP domain and external routers belong to the customers or another ISP. Intemal routers themselves can be of two types, namely edge routers and transit routers. Edge routers are internal routers that are adjacent to one or more external routers. Transit routers are internal routers that are only adjacent to other internal routers. In Fig: 9 (R0, R1, R3, R6, R8, R9) are edge routers and (R2, R4, R5, R7) are transit routers. While there is a chance for attack traffic to originate from the internal routers within the ISP's network domain, often most attacks originate outside the ISP's domain and target the victim which is also outside the ISP's domain. Hence all the malicious traffic mostly passes through at least two edge routers of the ISP's domain. The malicious traffic passes through only one edge router of the ISP if both the victim and attacking source are connected to the same edge router. The attack traffic enters the ISP's domain at the Ingress edge and exits from the ISP's network through the Egress edge. The edge refers to adjacency between an edge router and an external router. In this paper, we will focus on these types of external attacks. We consider DDoS attacks with spoofed source addresses, broad attack signatures, attacking systems changing the type of attack traffic. Attack signature in our scheme refers to congestion signature in [MBFIPS01]. One of our main aims in our scheme is to identify and eliminate the traffic from bad sources and provide more bandwidth to traffic from the poor sources to the victim. Bad sources send malicious (attack) traffic to the victim and poor sources send good traffic to the victim. Our aim is to prevent the attack at nearest point to the source of attack (that is, at the Ingress edge). In this section, we consider a single ISP domain. Currently we are extending our model to multiple ISP domains.

Our architecture involves a Controller-Agent model. In each ISP domain, we envisage that there exists a controller, which is a trusted entity (within the domain) and is involved in the management of denial of service attacks. In principle, the controller can be implemented on any internal (transit or edge) router or at a dedicated host. The agents are implemented on all the edge routers. We believe that this is appropriate as the transit routers contribute to only a small part of the attack traffic, which can be identified and prevented by the agent at the egress edge router of the ISP that is near to the victim. On the other hand, if the transit routers were known to contribute a large amount of attack traffic, then the agents can be deployed on the transit routers as well and this requires no modifications to our scheme. We will see below that using our packet marking technique and agent design, both edge and transit routers can identify the packets marked by other agents and the attacker. Only the controller has information about all the agents that are present in the domain. An agent has the information of its domain controller only. Both the controller and agents are designed to handle the attacks on multiple victims simultaneously. To simplify the presentation of our model, we consider a basic scenario and make the following assumptions. The controller[1] is always available and any host is able to contact the controller at any time. The communication between the controller and its agents is protected and no internal routers are compromised. Before considering the overall system operation, we will introduce our new packet marking technique and describe the controller and agent operation mechanism in detail.

4.1 Packet Marking Schemes: A Survey

Several packet-marking techniques have been proposed over the recent years such as the ones in [SWKA00, DFS01, SP00]. The proposed schemes mark the 32-bit IP address of the router with their own techniques by overloading the 16-bit Fragment ID field of an IP packet [Pos81]. Since packets are probabilistically marked, the victim would need large number of packets to calculate the total path traversed by the attack traffic. Even if we assume that each packet is marked, it could take a considerable amount of time for the victim to calculate the path, as computation is needed to retrieve the 32-bit IP address of the marked router. In general, the time taken to reconstruct the path (RT) is dependent on the number of attacking systems, number of parallel paths and length of the path [SP00]. Fig: 3 depicts the characteristics of the traceback approaches and Fig: 4 shows the characteristics that are required for our scheme.

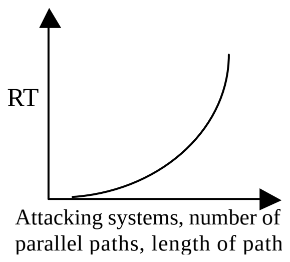


Fig: 3

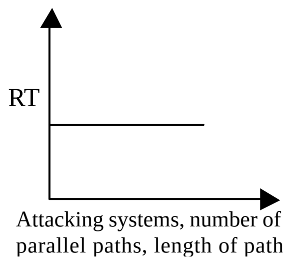


Fig: 4

4.1.1 A New Packet Marking Technique

We require a new packet marking technique that has the following characteristics:

- Only the Ingress agent should mark the packet. In case of incremental deployment, only the first agent that sees the traffic to the victim should mark the packet. If the agent receives a packet that is marked, it should easily determine whether the packet was marked by an authorized agent or by an attacker.
- The packet should be marked in such a way that the first (Ingress) agent that marked the packet can be identified with a minimum number of packets. In an ideal situation, we should identify this first (Ingress) agent by observing a single packet.

[1] There can be a group of controllers within an ISP to avoid DoS on the controller itself. In the case of a single controller, there can be a backup controller to avoid failure.

- Packets should be marked in such a way that it involves minimal computation overhead to retrieve the complete address of the first (Ingress) agent. In an ideal situation, a packet should carry the complete identity of the first (Ingress) agent. Otherwise, some additional computation would be needed at the destination.

Let us now propose a new packet marking approach that satisfies the above requirements. A system on the Internet is identified using a 32-bit IP address. An efficient packet marking technique cannot be developed if the 32-bit IP address is to be identified by overloading a 16-bit fragment ID field. Since our approach is based on the Controller-Agent model and is implemented within the ISP, there is no need to identify each router with its 32-bit IP address as long as the controller can uniquely identify each agent with a different identifier. For example, let us assume that there are 200 agents, the controller can easily identify each agent uniquely by assigning an agent ID with a maximum of say 8 bits (as 2^8-1 = 255). Since there are 16 bits in the fragment ID field, the controller can assign different ID to each agent and it can identify 2^{16} -1 (=65535) agents uniquely. So with this technique if the agent can mark each packet only at the first (ingress) agent, the controller can identify the ingress agent by observing a single packet. The controller just needs to maintain the database of the 32-bit IP address and the unique ID assigned for each agent. The 32-bit IP address of the agent can be easily retrieved from the agent ID marked in the fragment ID field. Thus we have achieved our first objective of identifying the approximate source address with a single packet. Next we need to determine whether the packet is marked by the agent or by the attacker or if it is a fragment. Since fragments are known to contribute to a variety of DoS attacks, our objective is to drop all the fragments *during* the time of attack. Moreover if fragments are allowed in our model, the attacker can flood the victim with malicious fragmented packets. Let us consider two possible cases in detail.

In the first case, the controller assigns a unique ID for each agent and informs the agents about all other valid agent IDs. Now if an agent receives a packet that is not marked, then it marks the packet with its agent Id in the fragment field. If the agent receives a packet that is already marked, then there are three options for this packet to be marked. (a) The packet is marked by some other agent. (b) The packet is a fragment. (c) The packet is marked by an attacker. Since our objective is to drop all the fragments and packets that are marked by the attacker during the time of attack, there is no need to differentiate the packet whether it is a fragment or marked by an attacker. As the agent has the details of all other valid agent IDs, it can check the marked packet for a valid agent ID. If valid, then it can determine that the packet is marked by another agent. If not, the packet could be a fragment or marked by an attacker and the agent drops the packet. The main disadvantage of this approach is that there would be a large number of packets generated during an attack and all these packets have to be searched for a range of valid agent ID's at every agent. This could cause considerable delay. Also the probability of an attacker sending a packet with a valid agent ID would be greater.

In the second case, both the controller and the agents identify each other by their ID's. That is, the controller assigns a controller ID for itself and assigns a unique agent ID for each agent. Now if the agent receives a packet that is not marked, it marks the packet with controller ID and its unique agent ID in the fragment ID field. If it receives a packet that is already marked, it only needs to check for the valid controller ID. If valid, the agent can determine that another agent has marked the packet and hence can pass the packet. Otherwise, the packet would be dropped as it could be a fragment or marked by an attacker. Note that in this case, the agent need not know the other valid agent IDs and hence the search process is more efficient. Another advantage of this approach is that the probability of an attacker marking the packet with controller ID is very less since there is only one valid controller ID. In the earlier approach a marked packet is treated to be valid if it matches with any of the valid agent IDs. The main disadvantage of this approach is that as the number of bits in the controller ID increases, the number of agents that can be identified uniquely decreases.

4.2 Controller Mechanism

The controller can be implemented on a dedicated host or on a router with Aggregate-based Congestion Control (ACC) [MBFIPS01]. It responds to the victim's request and sends commands to its agents. However in our case, there is no need for the controller to identify the attack (congestion) signature unlike the ACC router. Hence the implementation of the controller is much simpler than in [MBFIPS01]. Another difference between the ACC router and our controller is that the ACC router sends pushback messages only to the upstream routers irrespective of the administrative boundaries but the controller's commands (pushback messages) are to its agents that are located within the ISP domain. The controller can identify its agents using the unique agents' IDs and all the agents can identify its controller with the controller ID. When the controller receives a request from the victim of a DDoS attack, it dynamically generates and assigns these ID's and maintains the database of the 32-bit IP address and the agent ID assigned to each agent. These ID's vary each time whenever our scheme is invoked and remains constant for that particular attack. It is a must that when messages carrying these identities are transferred over the network, they must be protected for confidentiality and integrity and must facilitate data origin authentication. But these are standard network security problems and there is a range of security protocols that currently exists which can be used to achieve these. So we will not discuss how these are achieved in our model. The controller can operate either manually or automatically. This is because, in case of severe DDoS the victim may not be in a position to send a request to its controller. In this case the victim has to contact the controller out of band and the process starts manually. There are two important commands from the controller to its agents. The first command is issued to the agents when the Controller receives a request from the victim to prevent the attack. This includes the 32-bit IP address of the victim (multiple addresses for attack on multiple victims or a common prefix of the 32-bit destination IP address for attack on a group of addresses), controller ID (which is same for all the agents) and a unique agent ID for each agent. Fig: 5 shows the format of this command. The second command is issued when the victim identifies the attack signature based on the agent ID and requests the controller to prevent the attack at the identified agent. The controller retrieves the 32-bit IP address of the agent based on the agent ID and commands that particular agent to prevent the traffic that is matching the attack signature. Figure 6 shows the format of this command.

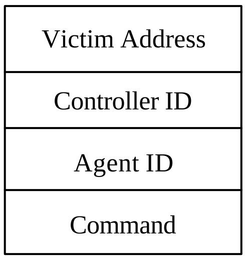


Fig: 5

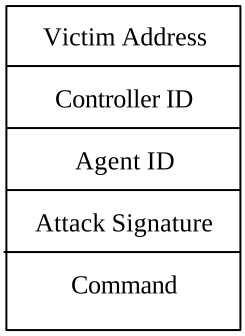
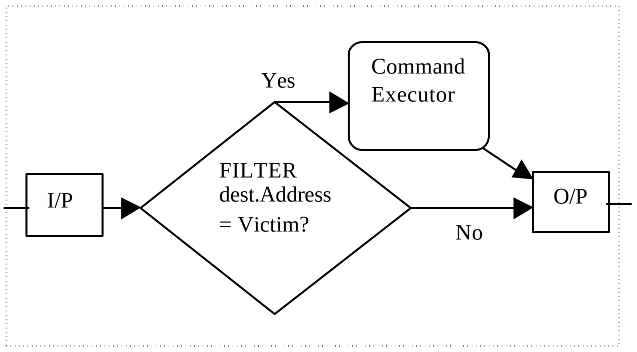


Fig: 6

4.3 Agent Mechanism

The agent can be implemented with ACC router [MBFIPS01] with minor modifications. The agent functions as an ordinary router and the special agent mechanism will be invoked only during times of attack, when it receives a command from its controller. Typically the agent can be assumed to be an upstream ACC router [MBFIPS01] that responds to pushback message from its downstream ACC router. Fig: 7 shows the implementation of the agent mechanism. When the agent receives the command from its controller, only the victim's traffic is filtered and fed to the Command Executor. The agent will place multiple filters for attack on multiple victims or a single filter with the common prefix of the 32-bit destination addresses for attack on a group of addresses.

There are two important stages in the implementation of agent mechanism as shown in Fig: 7. The first stage is invoked when the agent receives a command from its controller to mark the traffic to the victim. The command from the controller gives the information of the victim address, controller ID and the unique agent ID. Now the agent dynamically applies a filter with the destination address that of the victim and marks the packet with the controller ID and agent ID in the fragment ID field. Packets are marked only by the Ingress agent or by the first agent (in case of incremental deployment) that sees the traffic to the victim. All the fragments and packets that are marked by the attacker are identified and eliminated at this stage (see section 4.1.1). The second stage is invoked when the agent receives a command from its controller to prevent the attack traffic to the victim. In this stage, the agent drops all the packets that match the attack signature. All the packets that do not match the attack signature are again marked with the controller ID and the agent ID and are destined to the victim. The main reason for marking the packets even if it does not match the attack signature is to enable the victim to identify if there are any changes in the attack traffic pattern for it and generate a quick response.



Parameter **pkt** = packet

MarkPacket(pkt)
1. Check if pkt marked.
2. If pkt marked, drop and log pkt if controller ID not present.
3. If pkt not marked, mark packet with controller ID and agent ID.

BlockAttackTraffic(pkt)
1. If Pkt matches Attack Signature drop and log Pkt.
2. else **MarkPacket(pkt)**.

Fig: 7. Agent Mechanism

4.4 System Operation

Now we are in a position to describe the complete system operation. The operation begins when a system recognizes that it is under attack. If the victim is already equipped with some standard security tools such as firewalls and intrusion detection systems, then it can detect and prevent the DDoS attack at its boundary. If the victim decides to handle the attack upstream, then it contacts the controller in its domain. A session is established between the controller and the victim after proper authentication of the victim. As mentioned before, we will not discuss about authentication in this paper. Depending on the number of agents present within its domain, the controller will generate and issue the controller ID and unique agent Id to each agent and commands its agents to mark the traffic to the victim. Now the controller updates the victim with the details of the controller ID and the valid agent ID's. The agents filter the traffic that are destined to the victim and mark the traffic with controller ID and its unique agent ID in the fragment ID field. Packets will be marked in such a way that only the first agent that sees the traffic will mark the packet. All the fragments and packets that are marked by an attacker will be dropped at this stage (see section 4.1.1). Since agents are deployed on all the edge routers, all the traffic to the victim is marked with the controller ID and the ingress agent ID in the fragment ID field. As we have assumed that agents are deployed only on the edge routers, all the traffic originating in the backbone will be marked by the egress agent of the ISP that is connected to victim's network. Since the victim knows the controller ID and valid agent ID's, it can easily identify different attack signatures for each agent that are to be prevented at different agents. Now the victim updates the controller with attack signatures based on the agent ID's. The controller retrieves the 32-bit IP address of the agent from its database, based on agent ID and commands that particular agent to prevent the attack traffic from reaching the victim. As attack signatures are identified based on the agent ID, only the agent through which the attack traffic is passing will receive this command. Now all the agents that receive this command will start preventing the traffic that matches the attack signature from reaching the victim. Only the traffic that is matching with the attack signature will be dropped and logged at the agent. The traffic that does not match the attack signature will be marked with the controller ID and agent ID and destined to the victim. This is to enable the victim to easily track the changes in attack traffic. The agent will update the controller on how much attack traffic the agent is still seeing. Prevention will be done until the agent receives a reset signal from its controller. However the packets will be marked for an excess amount of time. This is very useful for intermittent type of attacks where attacking systems do not flood the victim continuously but send attack traffic at regular intervals. A diagrammatic representation of our scheme is shown in Fig: 8. In the figure dotted arrows represents the communication between the victim-controller and controller-agents. The thick lines between agents-zombies shows that the attack traffic is stopped at the ISP boundary. The controller is represented in dotted line because it is not necessary for the controller to be in between the path of agents and victim.

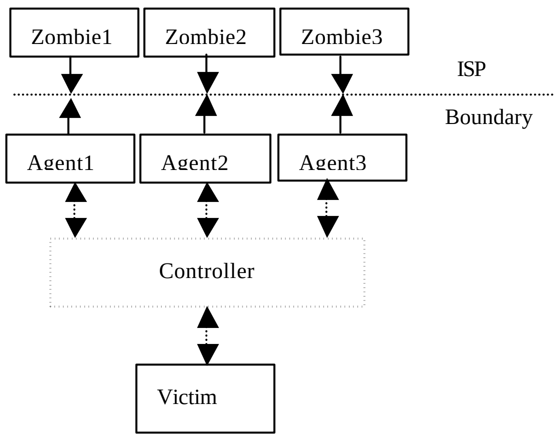


Fig 8: Representation of our Model

4.5 Example

Let us now illustrate our technique using a simple ISP backbone network shown in Fig: 9. Here R2, R4, R5, and R7 are transit routers and R0, R1, R3, R6, R8 and R9 are edge routers. The dotted lines in Figure 9 shows the direction of flow of the traffic to the victim V. Ati is the attack traffic from bad sources and Pi is the good traffic from good sources to the victim. Let us now examine how the various approaches proposed so far address the attacks in this scenario. We will show that our approach has many advantages when compared to other proposed schemes.

Center Track Approach

Let us consider the two schemes described in the Center Track paper [Sto00]. In the first scheme, all traffic to the victim is diverted through an overlay network passing through the Center Track router. The ingress edge router of the attack traffic is determined by the process of input debugging starting from the victim. Advantages of this scheme include the reduction in the number of hops required to trace the ingress edge and only the victim's traffic is diverted through the overlay network without disturbing other traffic. However there are several disadvantages in using this scheme: The first one is that this approach requires changes to the global routing table. This increases network complexity and causes additional administrative tasks. Even if the traffic is diverted through the overlay network, trace back is done through the process of input debugging. Attacks originating from backbone cannot be identified.

The second scheme maintains a database of all the traffic at every edge router (R0, R1, R3, R6, R8, R9). When an attack is reported the data is then searched based on the attack signature, to determine the Ingress edge adjacency of the attack. Advantages of this scheme include the following: Evidence of attack is retained at the first router near to the attacking source. No hops are needed for tracing the ingress edge of the attack. The main disadvantage of this scheme is that it requires a very high computation and storage overhead, as the entire traffic database has to be maintained, even when there is no attack. Difficult to determine where the data is to be stored. For example if data is be stored at edge routers, then all routers have to be queried. If data is to be stored in a single location, this will increase the network traffic since all the data is to be transmitted to the single location.

IP Traceback

In these approaches [SWKA00, DFS01, SP00], packets are probabilistically marked and only with partial router IP address. Hence the victim should receive a large number of packets to traceback the total path. Advantages of this approach include the ability to trace route the approximate source of an attack even in case of heterogeneous networks without cooperation of ISP's and the retention of evidence for legal proceedings. In general the proposed schemes have atleast some of the following disadvantages:

- Packets are marked on some probability and moreover each marked packet represents only a part of the IP address of the marked router. So it consumes lot of time to calculate the path traversed by the attack traffic.

- The reconstruction time is dependent on number of attacking systems, length of the path and number of parallel paths.
- Packets are marked even if there is no attack.
- Victim should have map of all the upstream routers.

Advantages of this approach are:

- It can automatically identify the attack at the congested router or with the request from the downstream server (victim) and can prevent the attack upstream.
- Filters are dynamically placed.
- It can prevent an attack that crosses different administrative boundaries.

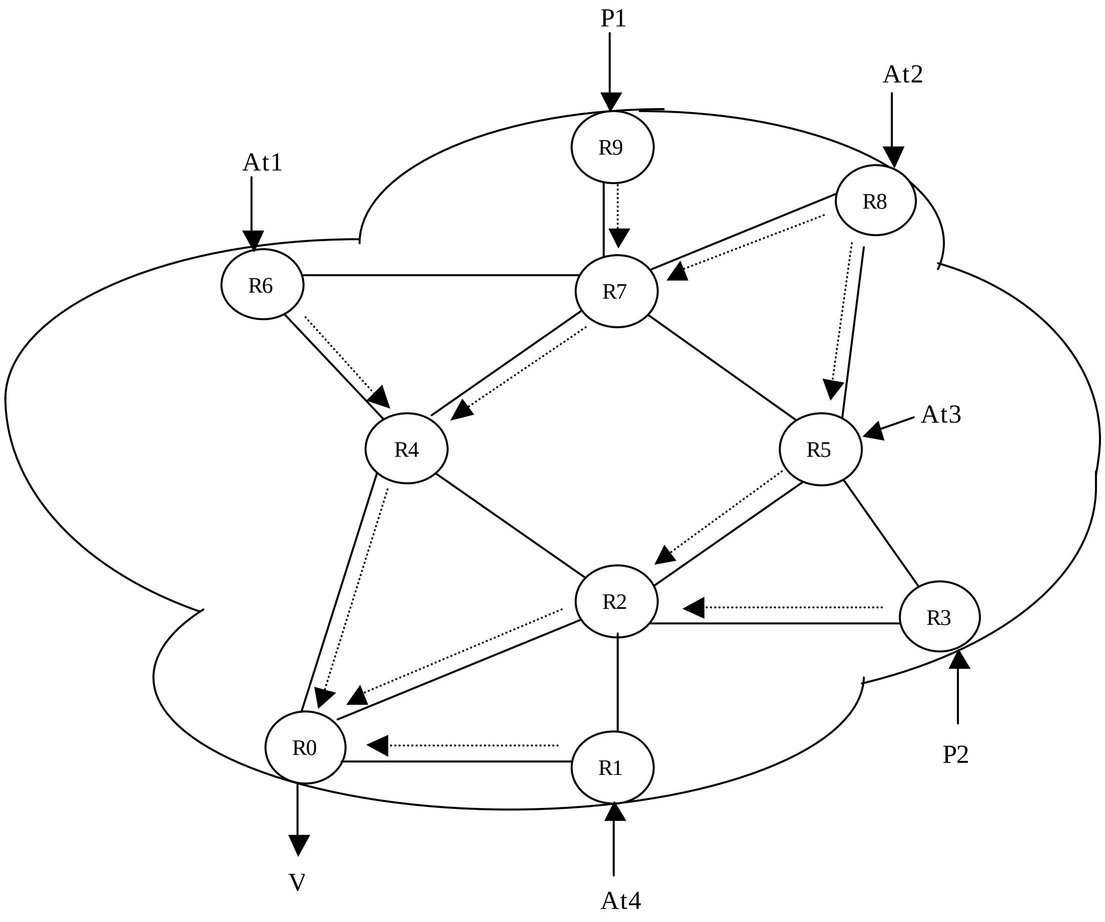


Fig: 9. ISP Backbone Network

Controlling High Bandwidth Aggregates

In this case, let us assume that all routers are replaced with ACC [MBFIPS01] routers and that there is no congestion at any of the backbone routers. So this process is invoked upon a request from the victim. Since the source addresses are spoofed, it is difficult to identify a congestion signature. If the attack is to be prevented effectively, the congestion signature should be broad enough to identify all the attack traffic (At1, At2, At3, At4). Further all the traffic is to be checked for a broad attack signature, which can cause considerable delay. If the congestion signature is narrow, then the attack cannot be prevented effectively. If congestion is detected at any of the backbone router, the ACC router has to consider different policies, past history and many other factors to make a decision on how to identify a congestion signature. So implementing this scheme can incur heavy cost.

- It can trace the approximate source through pushback technique if the attack is in progress.

Disadvantages of this approach are:

- Poor traffic such as from source P1 is not completely protected until pushback passes upstream from R0-R4-R7-R8 and P2 is not completely protected until pushback passes upstream from R0-R2- R5.
- Malicious source can be approximately identified through pushback technique *only* if the attack is in progress. Else no evidence is retained. So attacking source cannot be traced if the attacker doesn't flood the victim continuously. Intermittent attacks are difficult to trace.
- Push back occurs hop by hop upstream and the cost of the implementation of the overall scheme can be quite high.

Our Approach

Let us assume that the controller is implemented on any of the router and agents are implemented on all the edge routers R0, R1, R3, R6, R8, R9. Even if source addresses are spoofed, for example, all the attack traffic At1 will have a common ID of R6 (which is R6's unique agent ID) in the fragment ID field. So the victim can easily identify different attack signatures for different attacking systems based on agent ID (At1 for R6, At2 for R8, At3 for R0 and At4 for R1) and request the controller to prevent the attack at the identified agent. In this example, as the agents are implemented only at edge routers, the attack traffic At3 originating from R5 is identified and prevented at router R0. This is because there are no agents in between and R0 is the first agent to see the traffic to the Victim. If the agent were deployed on the router R5, then this would be identified and prevented at that point itself. Since the victim identifies that P1 and P2 is poor (good) traffic the agents at R9 and R3 will not receive any command to prevent traffic. So poor traffic is completely protected

The advantages associated with our approach are:

- Controller and agents function as ordinary routers when there is no attack.
- Victim can easily identify the different attack signatures for different attacking systems.
- Once victim is authenticated, identifying preventing and tracking any changes in the attack traffic is a totally automated process. Once invoked, the response time to identify and prevent the attack is fast.
- Approximate source can be traced with a single packet and since all the packets are marked at ingress agent (nearest point to the attacking source), it is made easy to initiate legal proceedings.
- Filters are dynamically placed.
- Once attack signatures are identified, prevention of attack traffic is directly near to the attacking source (ingress agent) instead of hop by hop.
- Each agent needs to check and prevent only the attack signature passing through it. The attack signature is very narrow and hence the delay experienced is very less.
- There is no need for the agents and controller to identify the attack signatures. Hence implementation costs for controller and agents is less when compared to ACC routers.
- Fragmented packets destined to a potential victim are eliminated only during the times of attack.

Disadvantages of our approach are:

- Good fragmented packets are completely eliminated *during* the times of an attack.
- At this stage, prevention is limited to within the domain of single ISP. Currently we are extending this approach to multiple ISP domains.
- Efficiency decreases as the infrastructure of the ISP increases, because more bits are required to accommodate all the agents. This reduces the number of bits available for the controller ID.

5 Additional Remarks

Incremental Deployment
The scheme proposed in this paper is relatively simple and is suitable for incremental deployment. The Controller-Agent design does not require extra intelligence to identify the attack signatures. Hence the cost of implementation is not high. Since the packet marking technique and the agents are designed to easily identify the marked traffic, ISP can start up with a controller and a single agent and incrementally deploy the other agents.

Limitations
Our scheme is not limited by the number of attackers, but by the number of agents. That is, it is dependent on the infrastructure of the ISP. For instance, greater the number of bits available for the controller ID, lower the probability for the attacker to send a packet with valid controller ID. The agents cannot identify the packets that are marked by the attacker and which match the controller ID and invalid agent ID. These are called false negatives can be easily identified and eliminated by victim since the victim knows what the valid agent IDs are. In some cases the packet will match with both the controller ID and agent ID. These are called false positives and they cannot be identified in our scheme. False positives will be identified as an attack signature for the agent with that particular valid agent ID. Extra security is needed to secure each agent and the controller must be secure.

Response time
Our scheme will be invoked as soon as the controller authenticates the victim. The response time for attack identification and prevention is mainly dependent on 1) How quickly the victim can identify that it is under attack and 2) How quickly the victim can identify the changes in attack traffic patterns.

Compatibility
Our scheme is compatible with congestion control technique, but not with the traceback technique. Our scheme will in fact make the congestion control technique more efficient as the ACC router can identify the aggregates more easily as the packets carry the router ID of only the ingress router. If traceback technique is deployed there is a greater probability for every packet to be marked by some router. So if our scheme is invoked during the times of attack, the agent will drop the packet if it detects that the packet is marked and if no controller ID is detected. If any of the trace back techniques are deployed, the agents in our scheme should be modified to replace the marked fragment ID field with the controller ID and agent ID instead of dropping the packet.

Multiple attack Signatures for a single router

We can even identify multiple attack signatures within a single router. In this case the controller assigns a unique agent ID for each input interface of its agent. For example, if we assume that an agent has three input interfaces and one output interface, then the controller assigns three agent ID's to this particular agent. Now the agent applies a filter with the destination address of the victim at each input interfaces and marks the packet with controller ID and unique agent ID for each input interface.

6 Concluding Remarks

In this paper, we have proposed a new scheme for detecting and preventing distributed denial of service attacks in networks. Our approach involves a Controller-Agent model in each ISP domain. The controller can be implemented on any internal router or at a dedicated host. The agents are implemented on all the edge routers. Our approach enables an automated process that can identify the approximate source of attack with a single packet even in the case of spoofed source address and prevents the attack at that point. It can identify different attack signatures for each source (agent) and prevents only the attack traffic. Our new packet marking technique enables to retain evidence at the nearest source of attack. Furthermore, the scheme is suitable for incremental deployment and has fast response time. Our future work will address the prevention of attack upstream in multiple ISP's without loosing the simplicity and effectiveness of this proposed scheme and the authentication between the victim, controller and agents.

References

[BC00] Hal Burch and Bill Cheswick: Tracing Anonymous Packets to Their Approximate Source. In proceedings of Usenix LISA, December 2000.

[Bel00] Steve Bellovin: The ICMP Traceback Message. http://www.research.att.com/~smb, 2000.

[Bel89] S.M.Bellovin: Security Problems in the TCP/IP Protocol Suit. ACM Computer Communications Review, 19(2): 32-48, Apr.1989.

[CER00] Computer Emergency Response Team. CERT Advisory CA-2000-01 Denial-of-Service developments. http://www.cert.org/advisories/CA-2000-01.html, Jan.2000.

[CER99] Computer Emergency Response Team. CERT Advisory CA-1999-17 Denial-of-Service Tools. http://www.cert.org/advisories/CA-1999-17.html.

[DFS01] Drew Dean, Matt Franklin and Adam Stubblefield: An Algebraic Approach to IP Traceback. In proceedings of NDSS'01, February 2001.

[DD99] D.Dittrich: The "stacheldraht" distributed denial of service attack tool. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt, Dec.1999.

[DO99] D.Dittrich: The "Tribe Flood Network" distributed denial of service attack tool. http://staff.washington.edu/dittrich/misc/tfn.analysis, Oct.1999.

[FS98] P.Ferguson and D.Senie: Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing. RFC 2267, January 1998.

[How98] John Howard: An Analysis of Security Incidents on the Internet. Ph.D.thesis, Carnegie Mellon University, Aug.1998.

[MBFIPS01] Ratul Mahajan, Steven M.Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker: Controlling High Bandwidth Aggregates in the Network. Draft, February 2001.

[MVS01] David Moore, Geoffrey M. Voelker and Stefan Savage, "Inferring Internet Denial -of-Service Activity," In proceedings of the 10th USENIX Security Symposium, August 2001.

[Pos81] J.Postel : Internet Protocol. RFC 791,Sept.1981.

[SAN00] SANS Institute Resources. Egress Filtering. February 2000. http://www.sans.org/y2k/egress.htm.

[SWKA00] Stefan Savage, David Wetherall, Anna Karlin and Tom Anderson: Practical Network Support for IP Traceback. In Proceedings of the 2000 ACM SIGCOMM Conference, pages 295-306, August 2000.

[SP00] D. Song and A.Perrig: Advanced and Authenticated Marking Schemes for IP Traceback. Technical Report UCB/CSD-00-1107, University of California, Berkeley, June 2000.

[Sto00] Robert Stone: "CenterTrack: An IP Overlay Network for Tracking DoS Floods," In proceedings of 9th Usenix Security Symposium, August 2000.