Volume 3, Issue 4, April 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Various Techniques Involved in Detection and Controlling IP Spoofing

Tanmay A. Abhang[#1],
[#]*M. Tech. Student, CSE Dept., SGGS IE&T Nanded-431606, India.*

Dr. U. V. Kulkarni[*2]
[*]*Professor & HOD, CSE Dept., SGGS IE&T Nanded-431606, India.*

Abstract— The Internet protocol suite is commonly known as TCP/IP protocol suite, because of its two important protocols, A) Transmission Control Protocol (TCP) and B) Internet Protocol (IP). IP spoofing, also known as IP address forgery is a technique in which an attacker attacks on a host by masquerading as a trusted host. Attacker does so by modifying the IP packet header with a forged or spoofed source IP address field. Main purpose behind IP spoofing is to conceal true identity of the sender by impersonating another computing system. By employing IP spoofing, attackers remain hidden from detection and put a considerable limitation on the destination network or victim for policing attack packets. In response to such attack the victim machine or network is misled to innocent network or host and drops subsequent packets from the same. IP spoofing has become a popular tool in various attacks like Distributed Denial of Service (DDoS) attack, Internet Control Message Protocol Magnification attack (ICMP smurf attack), TCP SYN (synchronize) flood attack, Ping flood attack and man in the middle attack. Various countermeasures for limiting IP spoofing are discussed in this paper.

Keywords— IP spoofing, DDoS, TCP SYN flood, ICMP smurf attack, Packet filtering.

1. INTRODUCTION

DDoS attack presents a great threat to the legitimate use of internet. In a DDoS attack, the incoming traffic flooding the victim originates from many different sources, potentially hundreds of thousands or more. This effectively makes it impossible to stop the attack simply by blocking a single IP address; plus, it is very difficult to distinguish legitimate user traffic from attack traffic when spread across so many points of origin.

Main purpose of such attack is to exhaust victim servers or saturating networks link. Based on victim, DDoS attack can be classified into two category a) Network resource attack and b) Server resource attack. In Network resource attack an attacker sends large number of useless packet in order to deplete the bandwidth of the link connecting network and the internet. In server resource attack attacker sends large number of packets to the server with intention to overload the victim server so that it can't process any packet further or consume all memory of the server. DDoS attack remains popular because of involvement of IP spoofing. As true location of attacker remains hidden it becomes very difficult to overcome such attack. To prevent a network or host from being a victim of DDoS or other attack involving IP spoofing, some preventive measures have been suggested by different authors. In this paper, we are surveying some of the techniques that help to solve or limit the IP spoofing problem.

2. VARIOUS TECHNIQUES TO CONTROL IP SPOOFING

2.1 Unicast Revaerse Path Forwarding

The Unicast Reverse Path Forwarding (uRPF) feature helps to mitigate problems that are caused by spoofed IP source addresses into a network by discarding IP packets that lack a verifiable IP source. When uRPF is used, the source address of IP packets is checked to ensure that the route back to the source uses the same interface that the packet arrived on. If the input interface is not a feasible path to the source network, the packet will be dropped. uRPF deflects IP spoofing attacks by only forwarding packets having source addresses that are valid and consistent with the IP routing table. This action protects the network of the ISP (Internet Service Provider), its customer and the rest of the Internet.

When uRPF is employed on an interface, the router examines all packets received as input on that interface. Router checks that each packets source address and source interface appears in the routing table and matches with the interface on which the packet was received. In other words uRPF checks to see if any packet received at a router interface arrives on one of the best return paths to the source of the packet. uRPF does this by doing a reverse lookup in the IP table. If the packet was received from one of the best reverse path routes, the packet is forwarded as normal. If there is no reverse path route on the same interface from which the packet was received, it might mean that the source address was modified or forged. If uRPF does not find a reverse path for the packet, the packet is dropped.

With uRPF, all equal-cost "best" return paths are considered valid. Meaning that uRPF also works in scenario where multiple return path exists. But the condition is that, each path must be equal to each other in terms of the routing cost (number of hops, weights etc.) as long as the route is in the FIB (forwarding information base) [2]. Upon receiving a packet at the interface where uRPF and access control lists (ACLs) have been configured, the steps are as follows:

© 2013, IJARCSSE All Rights Reserved

- input ACLs configured on the inbound interface are checked,
- uRPF checks to see if the packet has arrived on one of the best return paths to the source, which it does by doing a reverse lookup in the FIB table,
- FIB table lookup is carried out for packet forwarding,
- output ACLs are checked on the outbound interface,
- the packet is forwarded.

The uRPF can be used in any "single-homed" environment where there is essentially only one access point out of the network. Networks having one entry point are the best example of symmetric routing, which means that the interface where a packet enters the network is also the best return path to the source of the IP packet. uRPF works efficiently at the network perimeter for Internet, Intranet scenario and in enterprise networks with a single connection to an ISP. uRPF is simple but effective provided that symmetric routing is present but in today's internet routing is essentially asymmetric i.e. the forward and reverse paths between a pair of hosts are often quite different. In such scenario the uRPF becomes ineffective as it tends to drop such packet. uRPF loose mode tackles this problem.

The uRPF Loose Mode feature creates a new option by providing a scalable anti-spoofing mechanism suitable for use in "multi-homed network" scenarios [2]. Loose mode removes the match requirement on the specific ingress interface, allowing uRPF to loose-check packets. This packet checking allows the peer router of an ISP having multiple links to multiple ISPs to check the source IP address of ingress packets. Checking determines whether they exist in the FIB. If they exist, the packets are forwarded. If they do not exist in the FIB, the packets are dropped.

2.2. Hop Count Filtering

Idea behind Hop Count Filtering (HCF) is, though attacker can spoof arbitrary any IP address, he cannot forge or control the number of hops a packet takes to reach a network or host. Hence most of the packets with spoofed address will have a different hop count than legitimate packets hop count.

This hop count information can be obtained from time to live (TTL) field from the IP packet. Hop count information is not directly stored in IP header we can deduce it using information present in TTL field [3]. TTL is 8 bit field in an IP packet indicating lifetime of the packet in the internet. Upon received at router, value in the TTL field is decremented by one and then packet is forwarded to the next router or destination network. Therefore original TTL value minus number of hops to reach destination will give us final TTL value. But the problem is destination only knows the final TTL value. As there is no concurrency on the initial TTL value unless and until all operating systems use the same initial TTL, which is practically not the case. We cannot assume a single static initial TTL value for each IP address. However many widely used operating system selects initial value from 30, 32, 60, 64, 128 and 255 [4]. Very few internet hosts are separated by more than 30 hops, therefore we can deduce the initial TTL value of a packet by selecting next larger value from the set of initial TTL values. In the case of value 30 and 32, and 60 and 64, a hop-count value for each of the two possible initial TTL values are computed and if either hop-count matches the packet is accepted.

```
For each packet:
        extract the final TTL T and IP address S;
        infer the initial TTL To;
        compute the hop-count Hc=T -To;
        index S to get the stored hop-count Hs;
        if (Hc ≠ Hs)
        Packet is spoofed;
else
        Packet is legitimate;
```

Fig. 1. Algorithm for HCF

HCF builds an IP to hop-count mapping table. On receiving packet router extracts the source IP address and calculates hop count by the procedure shown in fig. 1 [3]. Then IP to hop-count table is searched for the extracted IP address if corresponding hop count matches with calculated hop count then packet is allowed to pass as it is legitimate one else packet is discarded. It is simple but effective technique. But if an attacker has knowledge of the hop count between a network and victim, he can spoof such addresses which are at same hop count distant from the victim as attacker itself. Then in such case the HCF will not be able to distinguish between attacker traffic and legitimate traffic. Attacker can use 'traceroute' utility to gain knowledge of hop count so, in such scenario HCF fails to control IP spoofing.

2.3. Path Identification

Path Identifier (Pi) is a deterministic packet marking mechanism in which a path identifier is attached to each packet so that victim can know the path traversed by the packet. Each packet traveling along the same path carries the same Pi. This scheme is extremely light-weight, both on the routers for marking, and on the victims for decoding and filtering [5]. By attaching an identifier to each packet based on the router path that it traverse, victim can filter packet itself based on the path information carried by that packet. Suppose a router drops a packet because of spoofed address, it remembers the path identifier of the dropped packet and discards all the subsequent packet traversing along path same as dropped packet or having same path identifier.

2.3.1 Pi Marking Scheme

In the packet marking scheme proposed by A. Yaar et al. [5], the 'identification' field of an IP packet is modified which is 16 bits in length. A router marks last '*n*' bits of its IP address in the IP identification field of the packet it forwards

© 2013, IJARCSSE All Rights Reserved

in a 'n' bit marking scheme. The identification filed is divided into $\lfloor 16/n \rfloor$ sections. For indexing section of the field mark, value of packets *TTL* modulo $\lfloor 16/n \rfloor$ is used. Fig. 2 shows the packet marking scheme used in Pi. Packet travels from source A to destination V. Between A and V routers R1, R2, R3, R4, R5 are present. On receiving packet on one of its interface a router insert marking into identification field using *TTL* value of t he packet as an index. For the sake of convenience here 1-bit marking in a 4-bit field is shown in fig. 2.

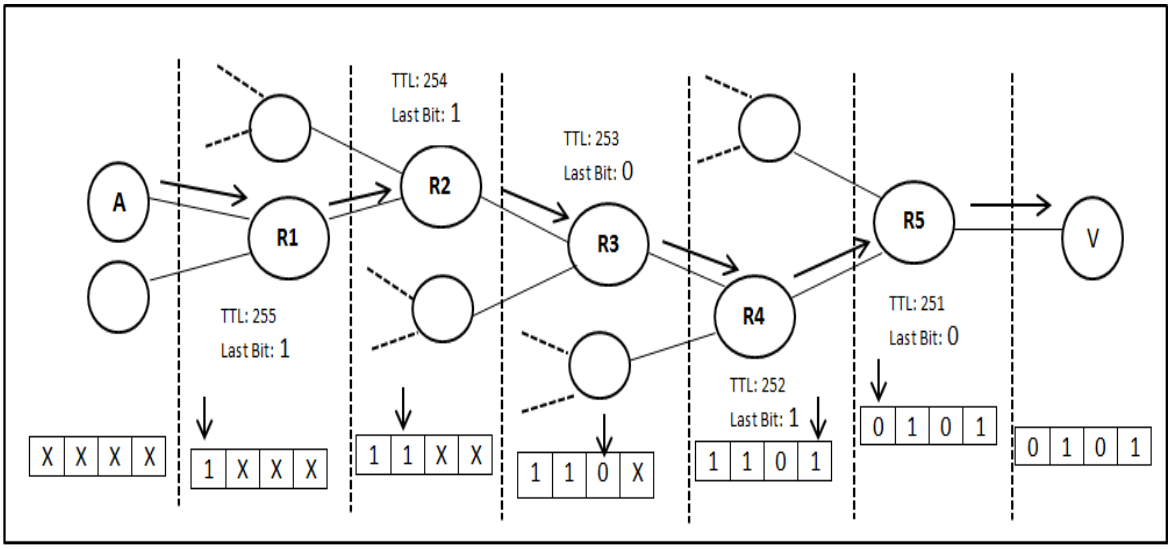

Fig. 2. Packet marking scheme.

Fig. 3 shows the algorithm for Pi [5]. In case of attack the victim can filter packets based on Pi markings. Vitim has to classify a single packet as an attack packet; victim then records the marking from same packet and further drops all packets carrying same marking. But unfortunately as there are only 16 bit identification field, there are 2^{16} number of Pi marks. As number of attacker increases it is more possible that any given Pi mark will receive some attack packets, which will cause all legitimate traffic to be dropped [5].

```
P = Pi mark of the packet
n = number of bits each router marks
Pimark(P, TTL,CurrIP, n)
{
M  = 2^n - 1;
B = markingbits(Curr_IP)&m;
bitpos = (TTL mod ⌊16/n⌋ ) n;
b << bitpos;
m <<bitpos;
return( (P& ~m)|b);
}
```

Fig. 3. Algorithm for Pi

2.4 Source Address Validity Enforcement protocol

This protocol when employed enforces all IP packets to carry correct source address. Source Address Validity Enforcement protocol (SAVE) is based on the building an incoming table that consists of association of each incoming interface of the router with different valid source address block. If such tables are deployed at many routers, choices of spoofing addresses reduced to great extent. If such tables are deployed at many routers, choices of spoofing addresses reduced to great extent. Every router has a forwarding table that indicates the outgoing interface for a given destination. SAVE suggests that there must be an incoming interface for a source address. Suggesting all packets from specified address space can be reach to destination indicated in incoming table of the router.

In this way SAVE works very much similar to forwarding table, but in reverse fashion. Routing updates are propagated in between routers so that each router knows the network reachability information of other router, similarly SAVE updates must be propagated allowing routers on the path to destination to gain knowledge of valid incoming interface for a source address. A SAVE router thus periodically generates SAVE update message propagated toward each entry in its forwarding table so that a valid incoming interface is set up along the route.

© 2013, IJARCSSE All Rights Reserved

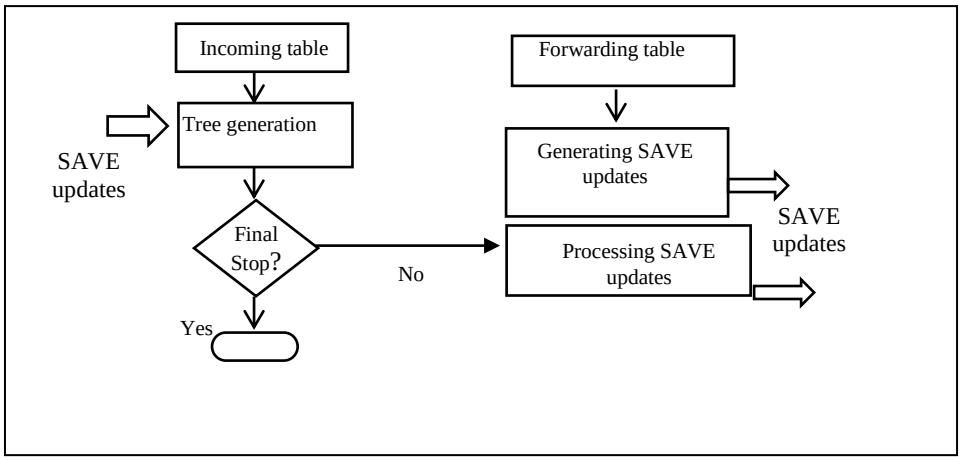

Fig. 4. SAVE protocol architecture

There are certain properties desired as SAVE runs concurrently along with routing protocol [7].

- In order to run smoothly on top of another routing protocol SAVE must be independent of that routing infrastructure and it must be modular in nature.
- It should update incoming table in accordance with the routing changes.
- It should produce minimum overhead and scale properly.
- SAVE should be secured so that attacker can't attack directly on SAVE.

The format of the SAVE update is as follows:

<destination address space = D, ASV = {S_R*}, appendable = true>*. Here *ASV* stand for Address Space Vector which records source address spaces on the path that this update has traversed [7]. This updates are sent within IP datagram having destination selected from D. On receiving this updates a router processes the updates in order to adjust its incoming table with the received update and forwards this updated information to downstream router as SAVE update. Fig. 4 [7] shows the architecture of SAVE protocol. After exchanging SAVE updates each router is equipped with the address space with their correct interface information, now it is able to filter packets. Disadvantage of this method is it introduces overhead in form of SAVE updates which must be exchanged frequently resulting in bandwidth consumption.

2.5. Packet Passport System

Packet passport uses a light weight message authentication code (MAC) such as hash-based message authentication code (HMAC) or message authentication code based on universal hashing (UMAC). A passport is nothing but a sequence of autonomous system (AS) numbers and their MAC's. MAC's are computed using a secret key known only to the source and passport checking domain between source and destination. Therefore passports cannot be forged by cryptographic methods [8]. As packet travels from source to destination, routers between them validate the passport.

Packet passport can be implemented as an IP option or as a shim layer [8]. Working of packet passport system is explained with the help of fig. 5. Host A sends a packet to destination B, both are situated at AS1 and AS4 respectively. In between them AS2 and AS3 are present. Whenever inter domain routing comes into picture, BGP also present as it is only protocol currently used for this purpose. At time packet reached at router R2, which is border router for AS1 is includes passport within IP packet. The passport included in IP packet has 3 MAC's for 3 pair of ASes i.e {(AS1, AS2), (AS1, AS3), (AS1, AS4)}. Using a secret key $K(AS1, AS_i)$ which is known only to two involving ASes, MAC is calculated. The secret key is known and distributed prior to communication. Methods of key distribution are explained detailed in paper [8].

Border router R3 on receiving packet uses the secret key *K*(AS1, AS2) to verify whether the packet has come from AS1 or not. So at this stage a router is able to filter packets on the basis of received MAC. If MAC matches with the source from which packet has arrived, packet is forwarded. But the packet is discarded if MAC doesn't matches with the source of the packet. Packet Passport system suffers from overhead involved in key distribution.

2.6 Network Ingress Filtering

Network ingress filtering is a packet filtering technique used by many Internet service providers to try to prevent source address spoofing of Internet traffic. Network ingress filtering is a "good neighbor" policy explained with the help of fig. 6. In the example, the attacker belongs to network 115.12.16.112/8, and tries to start an attack with spoofing address other than its network address. Network 115.12.16.112/8, has an Internet Service Provider D. An input traffic filter on the ingress or input link of "router 1", which provides connectivity to the attacker's network restricts traffic to only allow traffic originating from source addresses within the 115.12.16.112/8 prefix. It restricts an attacker from using "invalid" source addresses which reside outside of this prefix range.

© 2013, IJARCSSE All Rights Reserved

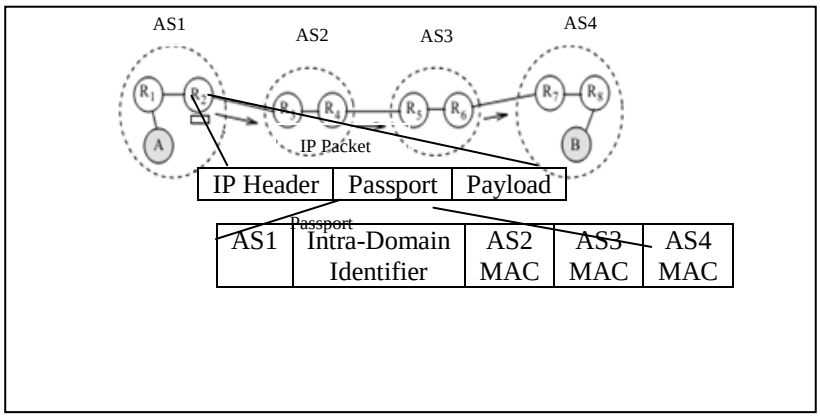

Fig. 5. Packet Passport System

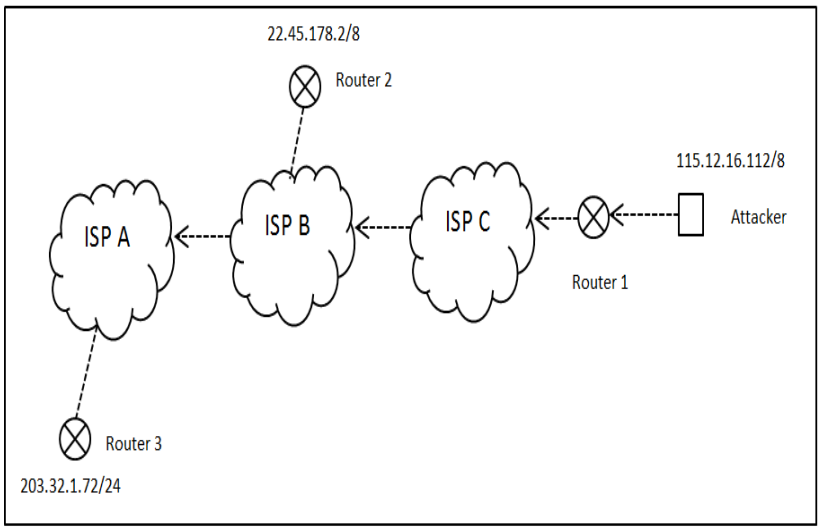

Fig. 6. Network ingress filtering.

If more internet providers and corporate network administrators implement ingress filtering, the opportunity for an attacker to use forged source addresses as an attack methodology will lessen. Filtering of this nature has the potential to break some types of special services e.g. mobile IP is affected by this method as traffic to the mobile node is tunneled, but traffic from the mobile node is not tunneled [9].

2.7 Inter Domain Packet Filter

Inter domain packet filter (IDPF) is an autonomous system (AS) level packet filtering technique based on the concept of 'Route based packet filtering'. In a network although an attacker can spoof any IP address, he cannot decide the path a packet takes to reach the destination. So, if a router has knowledge of routing paths for source to destination it can check whether it is present on that path if yes then forward the packet, if not discard the packet.

But problem with above approach is the router needs knowledge of global routing, which still is a challenge in current internet scenario, because every AS has its own policies which dictated by some local parameter. IDPF is an attempt to overcome the problem by making use of locally exchanged Border Gateway Protocol (BGP) update messages to gain knowledge of global routing assuming that all ASes employ a set of routing policies that are commonly used today. BGP is an exterior routing protocol which also is the de facto protocol for inter domain routing i.e. routing in between AS. Every AS has one or more border router that exchanges information by importing and exporting best route information to a node. It also advertises its own network and other networks that it can reach. BGP is basically used for this information exchange purpose.

Routing policy at an AS mainly depends upon the commercial relation between AS. A pair of AS can have provider to customer, peer to peer and sibling to sibling relationship [11]. IDPF assumes each AS employs its import and export policies according to the rules shown in fig. 7. It is shown by Z. Duan et al. [10], that when a routing system is stable i.e. after exchanging several BGP update messages, all neighboring border router has updated information of each other's network reachability information and further exchange is halted till a router advertises update message. At this point the export policies employed by any router u is, **export** (u→v) [{ **bestR** (u, s)}] ≠{ }. It means that at stable routing each router sends its best route for reaching a node 's' to its each neighboring router. Provided that all ASs follow the import and export policies described in fig. 7.

© 2013, IJARCSSE All Rights Reserved

If ((u1 if $((u1 \in customer(v) \cup sibling(v))$
and $(u2 \in peer(v) \cup provider(v)))$ $then$
r1.local_pref > r2.local_pref

Fig. A. Import policies

Export rules		**r1**	**r2**	**r3**	**r4**
Export to		**provider**	**customer**	**peer**	**sibling**
Learned from	**Provider**	No	yes	No	Yes
	customer	Yes	Yes	Yes	Yes
	peer	No	Yes	No	Yes
	sibling	Yes	Yes	Yes	Yes
Own routes		Yes	yes	Yes	Yes

Fig. B. Export policies

Fig. 7. Import & Export policies at an AS

Meaning of the above equation is at stable routing state each border router 'u' has notified or sent its best route to reach a destination 's' to its downstream neighbor 'v'. This lemma is pivotal as filtering decision for a packet, depends upon it. A router 'v' will accept packet M(s, d) that is forwarded by neighbor 'u' if **export** (u→v) [{ **bestR** (u, s)}] ≠ { } i.e. v will accept a packet with source address 's' and forwarded by 'u' if and only if prior to reception of the same packet, 'u' must have informed 'v about its best route to reach 's'. Otherwise source address of the packet is forged and must be discarded by 'u'. IDPF is employed at AS level, hence IP spoofing attack within AS is still possible.

3. CONCLUSIONS

IP spoofing is a threat that can cause great damage in a network as it is being used as a tool in most of the popular attacks like DDoS, TCP SYN flood, SMURF attack etc. In this paper we have discussed several techniques to mitigate the problem of IP spoofing. Depending upon situation and requirement, a network or an ISP can employ one of the above techniques. Wide acceptance and use of these techniques is highly recommended as it will certainly increase the strength of network to in order to fight against IP spoofing.

REFERENCES

[1] F. Baker, "Requirements for IP Version 4 Routers," RFC 1812, June 1995.

[2] (2007) Cisco Systems homepage on unicast reverse path forwarding. [Online]. Available: http://www.cisco.com/univrcd/cc/td/doc/product-/software/ios122/122newf%t/122t/122t13/ft_urpf.pdf.

[3] C. Jin, H. Wang, and K. Shin, "Hop-Count Filtering: An Effective Defense against Spoofed DDoS Traffic," *Proc. 10th ACM Conf. Computer and Comm. Security*, Oct. 2003.

[4] (2002) The Swiss Education and Research Network website. [Online]. Available: http://secfr.nerim.net/docs/fingerprint/en/ttl default.html.

[5] A. Yaar, A. Perrig, and D. Song, "Pi: A Path Identification Mechanism to Defend against DDoS Attacks*," Proc. IEEE Symp. Security and Privacy*, May 2003.

[6] A. Yaar, A. Perrig, and D. Song, "StackPi: New Packet Marking and Filtering Mechanisms for DDoS and IP Spoofing Defense," *IEEE J. Selected Areas in Comm*., vol. 24, no. 10, Oct. 2006.

[7] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang, "Save: Source Address Validity Enforcement Protocol," *Proc. IEEE INFOCOM*, June 2002.

[8] X. Liu, X. Yang, D. Wetherall, and T. Anderson, "Efficient and Secure Source Authentication with Packet Passport," *Proc. Second Usenix Workshop Steps to Reducing Unwanted Traffic on the Internet (SRUTI- '06)*, July 2006.

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

[10] Z. Duan, X. Yuan, and C. Jaideep, "Controlling IP Spoofing through Interdomain PacketFilters," *IEEE Transactions on dependable and secure computing*, vol. 5, NO. 1, January-march 2008.

[11] L. Gao, "On Inferring Autonomous System Relationships in the Internet*," IEEE/ACM Trans. Networking*, vol. 9, no. 6, Dec. 2001.

© 2013, IJARCSSE All Rights Reserved