Title:	Document Version:
Deliverable D2.5 **Requirements on Security in 6QM Project**	3.1

Project Number:	Project Acronym:	Project Title:
IST-2001-37611	6QM	IPv6 QoS Measurement

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type* - Security**:
30/04/2003	20/06/2003	R – PU

* Type: P - Prototype, R - Report, D - Demonstrator, O - Other
** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Responsible and Editor/Author:	Organization:	Contributing WP:
Jean-Michel Combes	FT	WP2

Authors (organizations):

Miguel Angel Díaz (Consulintel), Jordi Palet (Consulintel), Yann Adam (FT), Alexandre Dubus (FT), Zainab Khallouf (FT), Emile Stephan (FT), Lidia Yamamoto (HEL).

Abstract:

This document specifies the needs for the security of the architecture, the data and the detection of DoS attack.

Keywords:

Denial of Service (DoS), Distributed DoS (DDoS), Integrity, Privacy, Security.

Revision History

The following table describes the main changes done in the document since his creation.

Revision	Date	Description	Author (Organization)
v0.1	05/11/2002	Document creation	Yann Adam (FT)
v0.2	24/01/2003	Integration	Emile Stephan (FT)
v0.3	03/03/2003	DDoS	Jean Michel Combes (FT)
v0.4	28/04/2003	Architecture and data	Jean Michel Combes (FT)
v0.5	28/04/2003	Deliverable preparation	Emile Stephan (FT)
v0.6	21/05/2003	Deliverable style correction	Alexandre Dubus (FT)
v0.7	21/05/2003	Completion of the multicast section	Zainab Khallouf (FT)
v1.1	23/05/2003	Clarification	Jean Michel Combes (FT)
v1.2	27/05/2003	Clarification	Jean Michel Combes (FT)
v1.5	28/05/2003	Clarification	Jean Michel Combes (FT)
v2.0	28/05/2003	Finale version	Emile Stephan (FT)
v2.1	30/05/2003	Review comments	Lidia Yamamoto (HEL)
v2.2	02/06/2003	Clarification	Jean-Michel Combes (FT)
v2.3	03/06/2003	Clarification	Jean-Michel Combes (FT)
v2.4	05/06/2003	Review	Lidia Yamamoto (HEL)
v2.5	05/06/2003	Review	Jean-Michel Combes (FT)
v2.6	10/06/2003	Review	Jean-Michel Combes (FT)
v2.7	10/06/2003	Review (Format)	Jean-Michel Combes (FT)
v2.8	18/06/2003	Review (Format)	Emile Stephan (FT)
v2.9	19/06/2003	Review	Jean-Michel Combes (FT)
v3.0	20/06/2003	Minor corrections	Miguel Angel Díaz (Consulintel)
v3.1	20/06/2003	Final Review	Jordi Palet (Consulintel)

Executive Summary

The main security issue in QoS measurement architecture is to secure signaling between all the actors permitting to collect data. Such architecture cannot work and provide concrete results to the ISP if its elements are incontrollable and give back bad data.

The second important point is to ensure that data collected are the good ones (authentication of the sender, integrity of data and anti-replay) in order to setup QoS-based services. Similarly, confidentiality of such measurements must be guaranteed in order to protect business.

Customers may disagree with having the traffic of their applications to be analyzed and so to be metered. Privacy may be important for customers so ISP have to respect it.

Denial of Service (DoS) is not the main security issue but may be a important danger for an ISP with the loss of the quality of the services provided. Customers cannot accept to have their link to Internet disturbed or their IP based services disabled due to an attack on the provider networks. Thus, the detection and the suppression of DoS is necessary for an ISP. The information collected by QoS measurement systems can be used to detect DoS attacks. In this deliverable we devote one chapter to this issue.

Measurements of the performance of IP based services require high tech systems. These systems should not be used to destroy the network they are measuring.

This document illustrates all these points.

Table of Contents

Table of Figures

1. INTRODUCTION

Measurements techniques may raise issues concerning security and privacy.

Active techniques, in which traffic is injected into the network, can be abused for denial-of-service attacks disguised as legitimate measurement activity.

Passive techniques, in which existing traffic is recorded and analyzed, can expose the contents of Internet traffic to unintended recipients.
This document firstly illustrates the concerns and the issues related to signaling and data protection and confidentiality. The next part describes a list of requirements resulting of these issues. Then it focuses on Denial of Service. It describes the states of the art, identifies measurement parameters usable to perform DoS prevention operations. Then it lists requirement for of a resilient measurement system.

2. ARCHITECTURE AND DATA

2.1 Architecture from a Security Point of View

Here are the different entities of the 6QM architecture:

- Measurement point/point of Measure.
- Measure.
- Collector.
- Management entity (Configuration and Fault).

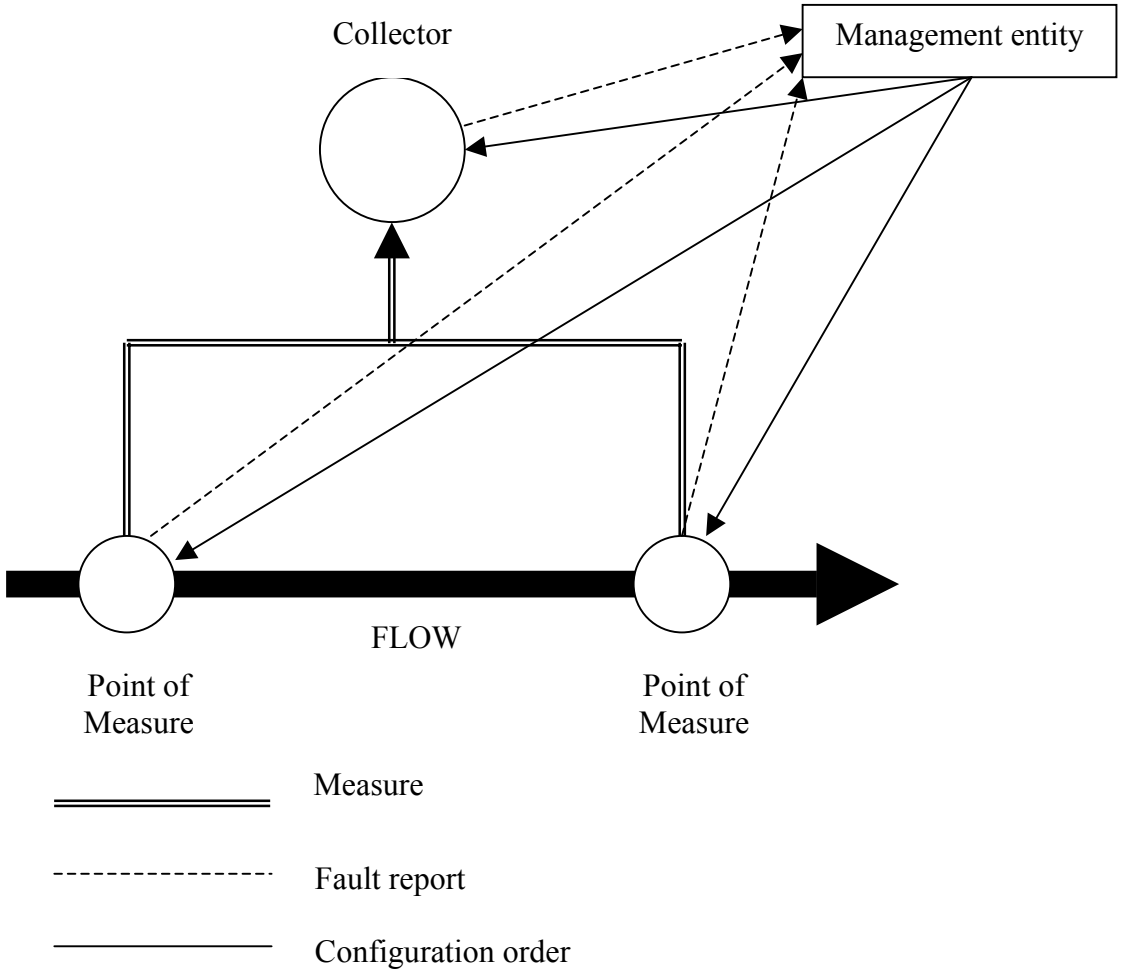

Figure 2-1: Basic Architecture

2.2 Security Analysis

2.2.1 Measurement Points/Points of Measure

As the points of measure are on the production network, some "bad guys" wanting to disturb the measurements could attack them easily by a DoS attack.

2.2.2 Measure

In the case of passive measurement, a typical threat is a "man in the middle" DoS attack. Between two probes, an attacker could modify/block the packets that are used for the measurement.

In the case of active measurement, of course, there is also the same problem concerning "the man-in-the-middle" DoS attack.

Another case of security problems concerns the transfer of the measure to the collector. There could be a "bad guy" between the point of measure and the collector that could modify/block the data sent and so could disturb the service.

At least, from the point of view of the network users, there could be some privacy problems. In the particular case of passive measurement, the packets are recorded and analyzed.

2.2.3 Collector

A collector could be the target of a DoS attack to disrupt the measurement service. But we could suppose that a collector is on a separated network.

2.2.4 Management

Like in all mechanisms, the security of signaling is vital for the services providing by these services. In a measurement context, the orders sent to the different actors (i.e. points of measure and collector) must be secured to avoid a disruption of the measurement. In particular, these actors (i.e. points of measure and collector) must be able to verify that the orders come from the management entity and that the integrity of these orders is not compromised.

In a same case, the management entity must be able to have the right data concerning the measurement architecture and so to allow the administrator to take the right decisions. Thus, the entity management must be able to verify that all data concerning the fault management are really coming from the measurement actors (i.e. points of measure and collector) and that the integrity of these data is not compromised.

2.3 Security Requirements

From Section 2.2 we can now derive the corresponding security requirements for each element of the architecture.

2.3.1 Measurement Points/Points of Measure

Requirement A1.1: Points of measure protection. These entities **MUST** be protected against DoS attacks and in particular flooding attacks.

Requirement A1.2: Security of signaling to the points of measure. The signaling to control a Measurement Point **MUST** be secured to ensure the authentication of the management entity and the integrity of its orders (e.g. in using IPsec/AH/transport mode).

2.3.2 Measure

Requirement A2.1: Measures security. The results sent to the collector **SHOULD** be protected to ensure the authentication of the sender, i.e. the Measurement Point, and the integrity of the measurement information (e.g. in separating public network from "measurement" network, a private network containing the collector and the management entity and only linked to the points of measure OR in using IPsec/AH/transport mode).

Requirement A2.2: Active measurement protection. The copied packets **SHOULD** be protected to ensure the authentication of the sender, i.e. the Measurement Point, and the integrity of the measurement information (e.g. in using IPsec/AH/transport mode).

Requirement A2.3*: Active measurement confidentiality.* The copied packets **MAY** be secured to ensure the confidentiality of the measurement information (e.g in separating public network from "measurement" network, a private network containing the collector and the management entity and only linked to the points of measure OR in using IPsec/ESP/tunnel mode).

2.3.3 Collector

Requirement A3.1: Collector protection. This entity **MUST** be protected against DoS attacks and in particular flooding attacks.

Requirement A3.2: Signaling to collector security. The signaling to control a collector **MUST** be secured to ensure the authentication of the management entity and the integrity of its orders (e.g. in separating public network from "measurement" network, a private network containing the collector and the management entity and only linked to the points of measure OR in using IPsec/AH/transport mode).

2.3.4 Management

Requirement A4.1: Configuration management signaling security. The signaling to control a Collector **MUST** be secured to ensure the authentication of the management entity and the integrity of its orders (e.g. in separating public network from "measurement" network, a private network containing the collector and the management entity and only linked to the points of measure OR in using IPsec/AH/transport mode).

Requirement A4.2: Fault management signaling security. The signaling to inform a management entity **MUST** be secured to ensure the authentication of the Fault Management and the integrity of its information (e.g. in separating public network from "measurement" network, a private network containing the collector and the management entity and only linked to the points of measure OR in using IPsec/AH/transport mode).

2.3.5 Summary of the Requirements

Type of requirement	Req. ID	Requirement	Level of requirement
Measurement points/ Points of measure	A1.1	Protection against DoS attacks and in particular flooding attacks.	Must
	A1.2	Authentication of the signaling to control a Measurement Point.	Must
Measure	A2.1	Authentication of the results sent to the collector.	Should
	A2.2	Authentication of the copied packets.	Should
	A2.3	Ciphering of the copied packets.	May
Collector	A3.1	Protection against DoS attacks and in particular flooding attacks.	Must
	A3.2	Authentication of the signaling to control a collector.	Must
Management	A4.1	Authentication of the signaling to control a collector.	Must
	A4.2	Authentication of the signaling to inform a management entity.	Must

Figure 2-2: Security Analysis Requirement

3. PRIVACY

Privacy may be an important concern because companies refuse to have their voice traffic, their email and their files to be observed by third parties. The main concern is to avoid this information to be potentially used by competitors. But their marketing services are looking intensively for customer behavior, for customer relation management.

Citizens consider privacy as a fundamental right. As Internet is now part of their lives they do not want the analysis of the behavior of their traffic to potentially track them during all the days.

From a technical point of view, customers may want to use security mechanisms to protect their privacy, such as IPsec. In that case, it is important to the network provider not to modify user's packets during passive measurement to avoid packets to be rejected by the customer's receiver side.

Thus, the requirement concerning privacy and measures must be added to the measure requirements (section 2.3.2).

- *Requirement P1.1: Privacy guarantee.* The secured packets used for the measures **MUST NOT** be modified to ensure the communication between the users.

4. DENIAL OF SERVICE

The goal of this section is to provide a bibliographical state of the art concerning methods that can be used to avoid denial of services attacks (DoS). We define denial of service attacks by attacks that are designed to reserve or use a large number of resources on a networked device by performing a large number of service requests. The reservation of resources is such that legitimate users cannot access the service or can only access the service with degraded performance.

Today, Denial of Service (DoS) attacks are mainly caused by two types of requests; TCP connection establishment requests (TCP SYN) and ICMP requests. These attacks cannot be easily stopped because they usually originate from computers where IP source addresses are modified. Because of the way routing is performed in the Internet, the modification of the source address does not prevent packets to be correctly routed to their destination. However such attacks can also occur with legitimate addresses or legitimate flow making attack detection more difficult. Moreover the amount of traffic generated by attackers can sometimes be so large that local access control measures can be either impossible or useless.

Since DoS attacks constitute a major threat to any device or service connected to the Internet, it is important to implement techniques in the network in order to limit the ability of attackers to generate such attacks. These techniques may serve one or several objectives such as:

- Avoiding Address spoofing.
- Tracking packets, flows or flow aggregates in the network when spoofed addresses are used.
- Control the bandwidth allocated to flows or flow aggregates.
- Detect resource starvation or over-reservation.

A meaningful DoS avoidance strategy combines several techniques to cover the four main aspects of DoS attacks avoidance: Prevention, detection, tracking of packets, flows or aggregates creating the DoS and suppression of the attack. These aspects can be combined.

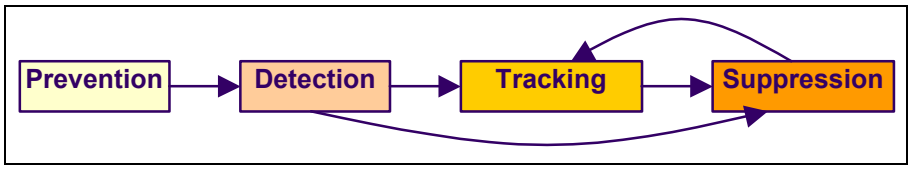

Figure 4-1: DoS Attacks Prevention Steps

This section does not focus on how DoS attacks can be detected (Detection Step) but rather on the identification of attackers and on techniques that can be used to prevent either in a preventive or reactive way these attacks (Prevention, Tracking and Suppression steps).

It is organized as follows; we first present a summary describing the denial of service problem and show several types of attacks that can be used in practice in order to generate denial of service either on network equipment or end devices. We then evaluate the importance of denial of service attacks. In the next part we describe existing proposal aimed at detecting, limiting or suppressing denial of service attacks. This section has two goals. The first one is to identify risks that may result in DoS attacks on measurement components. The second one is to define requirements that may be used to eliminate or reduce these risks.

4.1 Specific Symbols and Acronyms

ACL	Access Control List		
CPU	Control Processor Unit		
DoS	Denial Of Service		
DPF	Distributed Packet Filtering	**RPF**	Reverse Path Forwarding
FMS	Fragment Marking Scheme		
IAB	Internet Activities Board	**TCP**	Transmission Control Protocol
IETF	Internet Engineering Task Force	**TOS**	Type of Service
INF	Ingress Network Filtering	**TTL**	Time to Live
IP	Internet Protocol	**UDP**	User Datagram Protocol
		VPN	Virtual Private Network
ISO	International Standards Organization		

4.2 Description of the Problem

By denial of service attacks we designate attacks that are designed to reserve or use a large number of resources on a networked device by sending service requests. The reservation of resources is such that legitimate users cannot access the service or can only access the service with degraded performance.

Although denial of service attacks can take several forms and occur in several kinds of networks, we will focus in this document on attacks that occur in an Internet environment.

The last 10 years have seen a large increase in the number of denial of service attacks and denial of service schemes have greatly evolved in term of complexity. For example [Mo01] reports that among a group of computer security professionals, 27% had been exposed to DoS attacks in 2000.

Although we do not plan here to give a full coverage of possible denial of service techniques, we classify denial of service attacks into three main classes according to the degree of indirection between the attacker and the device under attack.

4.2.1 Direct Attacks

Direct attacks are the simplest form of denial of service attacks. In a direct attack the attacker sends service requests to the victim as fast as it can. Depending on the type of attack, it may be not so fast as the attacker wants. For example, depending on the configuration of the victim, TCP SYN flooding attacks usually require a fraction of the bandwidth to become effective ([Mo01] reports that a 500 pkt/s flow can be sufficient to overwhelm a server).

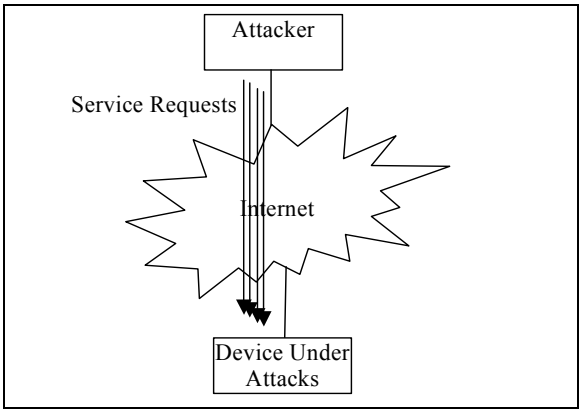

Figure 4-2: Direct Attack

Although direct attacks are very easy to perform, they are not generally very effective since:

- The attacker is limited by the amount of resources available to his computer. Consequently the attacker is usually unable to generate a large amount of traffic. As a result most attacks will result in a number of requests that can be sufficiently low to be undetected by the device under attack.
- When the flow is sufficiently large, the attacker can be easily detected and identified either because he has to use a real IP address in the case of attacks requiring several packet exchanges or because the traffic coming from his computer is relatively easy to track in the network. The traffic is usually easy to track because most of the traffic is flowing between two fixed points in the network.
- The attacker can be limited in his ability to change his IP address if ingress address spoofing check (Network Ingress Filtering – [BCP38]) and/or Unicast Reverse Path Forwarding (uRPF) [Cis00] is performed. Network Ingress Filtering and uRPF can usually be employed when symmetric routing is used. As a result these two techniques can be very effective at limiting direct attacks when implemented at the network perimeter for Internet, extranet and intranet environments or in ISP environments for customer network terminations. Network Ingress Filtering and the uRPF technique are described in Sections 4.4.1.1 and 4.4.1.2 respectively.

4.2.2 Slave Attacks

In order to avoid resource limitation problems and render attack tracking more difficult, the attacker may decide to use slaves in order to amplify the attack by creating distributed denial of service attacks. Slaves are usually made of unprotected computers belonging to corporate or private customers that have been installed with slave software. This slave software can be installed without the knowledge of the user by taking advantage of viruses, worms or regular software downloaded over the Internet.

Infected computers can at any time be connected or disconnected to the Internet. As a result the attacker must have a way to count available slaves. [Gib01] presents an example where the communication between slaves and the attacker is done using IRC channels where available slaves report when connecting to the Internet.

When the attacker decides to launch a distributed attacks against his victim he sends instructions to available slaves who would attack the victim by following a process similar to the one used in a direct attack.

In order to further hide his actions, the attacker can add an additional layer of indirection by using one or several master slaves that would be able to send instructions to simple slaves.

The distribution of the slave or master slave software may seem difficult. However recent viruses such as “I love you” or “Code red” have demonstrated that this code distribution may not be such a large problem. Using slaves brings a large number of advantages to the attacker. Not only does it give him much more power against the victim but it makes it much more difficult to detect the attack. In the case of a direct attack flows between the attacker and the victim, the attacker can usually be easily tracked because of the size of flows. In the case of Distributed DoS attacks (DDoS), an attacker with thousands of slaves can limit the attack of each slave to a very small flow of packets that is very difficult to distinguish from a legitimate one. In extreme cases the attacker can develop slaves that will not have to spoof IP addresses so that ingress filtering and uRPF techniques are made useless. However this last point makes it easier to find slaves and filter slaves requests.

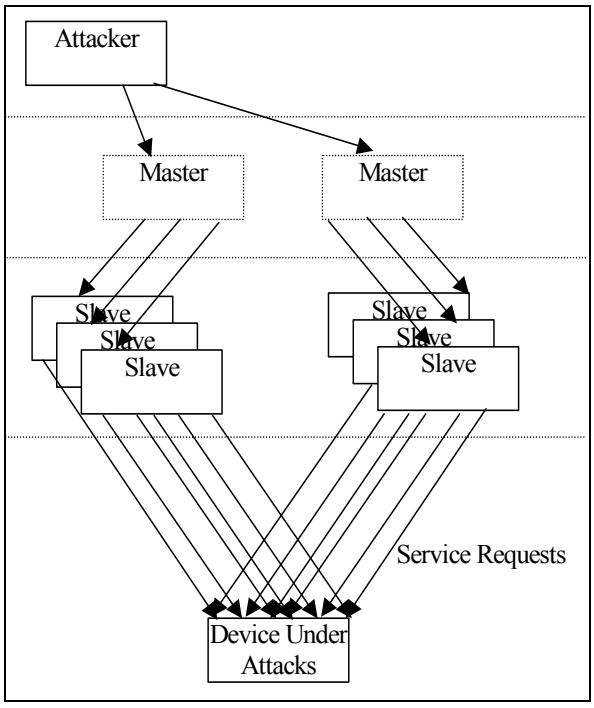

Figure 4-3: Attack Using Slaves

4.2.3 Reflector Based Attacks

As described in [Pa01], attackers can render denial of service attacks more difficult to detect by hiding the denial of service traffic using deflectors.

Instead of sending request directly to the victim, slaves can be instructed to send requests to intermediates such as web servers or web proxies with a fake return address. These intermediates are called reflectors. A reflector is basically any IP host that will return a packet if a packet is sent to him. If the fake return address is the address of the victim, the reflector will send the answer to the victim instead of answering to the slave therefore increasing the number of indirections between the attacker and the victim and therefore making the attacker more difficult to track. The number of reflectors can also render tracking more difficult by increasing the dispersion of the attack sources. This dispersion is not very difficult to achieve since any publicly accessible machine can basically be used as a reflector.

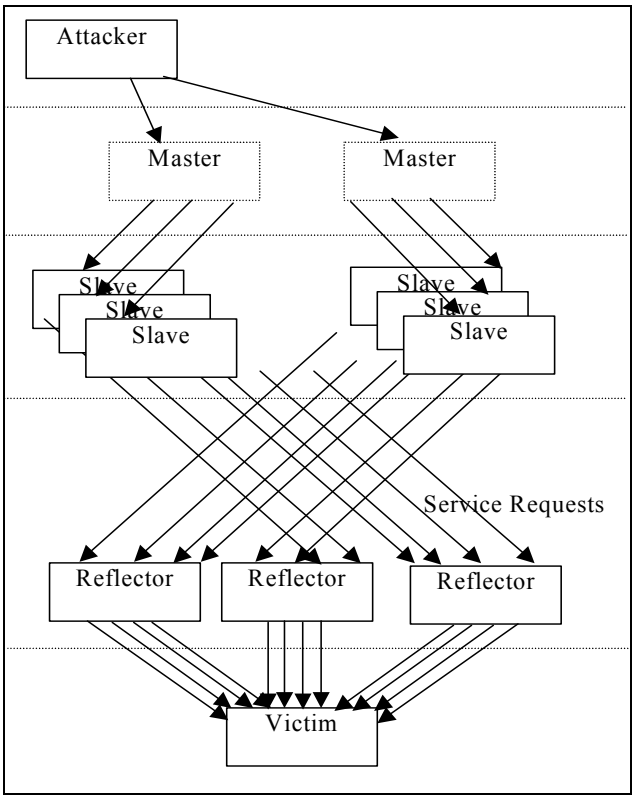

Figure 4-4: Attack Using Slaves and Reflectors

4.3 Attacks Evaluation

4.3.1 Current Trends

[Mo01] is to our knowledge the only paper to provide a general analysis of denial of service attacks by performing the analysis of backbone packet traces. A monitor is set up inside the backbone network and is expected to capture all packets flowing on a backbone link between a /8 network and the rest of the Internet. The detection of attacks is done using a behavior common to most denial of service generation tools. Actually, most DoS generation tools hide the address of the attacker by generating packets with a random source address. As a result, analyzing unsolicited packets coming back from a network node can be sufficient to understand if the source is currently experiencing an attack. An attack is identified by a flow of unsolicited packets with destination addresses randomly distributed over the IP address space and are called "backscatter". This analysis is based on the following hypothesis:

- Address uniformity: Attackers spoof source addresses at random.
- Reliable delivery: Attack traffic is delivered reliably to the victim and the monitor can reliably observe backscatter.
- Backscatter is composed of unsolicited packets observed by the monitor.
- The analysis carries on 1/256 ($2^{24}/2^{32}$) of the IP address space.

A trace including a sufficiently large number of packets allows DoS attacks to be identified and the attack rate to be estimated. Attacks identified during the course of one week were then analyzed in the paper. The main findings reported in the paper are the following:

- 12805 attacks targeting more than 5000 victims were detected. The total number of attack packets was estimated to 200 M.
- Most attacks were generated through TCP SYN packets (61% of attacks and 29% of packets) and ICMP requests (39% of attacks and 71% of packets)
- 38% of uniform random attack events had an estimated attack rate over 500 pkt/s (sufficient to overwhelm a server). The highest estimated attack rate was over 670 kpkt/s.
- 50% of the attacks lasted less than 10 minutes, 90% of the attacks lasted less then one hour. The longest attack lasted several days.
- A large part of victims were home users (13% of attacks, 18% of packets). A small part of the attacks were directed against routers (2% of attacks but 4% of packets).

4.3.2 Future Evolutions

[Hou01] provides a description of the future evolution of DoS attacks through the analysis of several different trends.

In the deployment of attacks, authors expect a change in the propagation of attack tools allowing the deployment to be completely automated. Deployment phases including scanning, exploitation, deployment and exploitation should become more and more automated allowing autonomous propagation of the tools.

The authors also expect an evolution in the nature of victims. Victims used to be mainly Unix based systems, however improvements in attacks against Microsoft Windows as well as the widespread use of this operating system makes it more attractive for attackers. Such attacks may use worms like Code-Red or Nimda in order to facilitate attack tools propagation. Propagation among Windows operated computers is also facilitated by the lack of technology and security knowledge of their owners.

Another trend in propagation techniques is the increasing use of router to generate attacks. Routers constitute an ideal attack platform for attackers because they usually provide the opportunity to the attacker to generate more powerful attacks by using routing/forwarding capabilities. Moreover routers are usually less protected through security or monitoring mechanisms making them easier targets.

In the execution of attacks, authors expect two main evolutions: The bandwidth available to attackers should continue to increase through an increase in the size of networks as well as through the migrations of users to broadband technologies. DoS traffic may also become more difficult to detect for several reasons. Attacks tend to use more and more legitimate addresses making the distinction between legitimate and illegitimate requests more difficult to make. Additionally DDoS attack tools also tend to communicate with more legitimate traffic making their detection more difficult. Finally the use of end-to-end encryption techniques renders monitoring techniques harder to implement.

Finally authors note that devices or services impacted by DoS attacks are no longer limited to a single administrative entity. The interconnection and integration of devices and services that may physically reside in various places of the network but that depend on services provided by each other makes attacks more effective and increase the impact of attacks.

4.4 Denial Service Detection and Prevention

Although several approaches have been proposed to handle distributed denial of service problems most share common constraints:

- (C1) Attackers are able to send any packet.
- (C2) Multiple attackers can act together.
- (C3) Attackers are aware of the DoS prevention scheme.
- (C4) Routes between hosts are generally stable.
- (C5) Packets can be reordered or lost.
- (C6) Routers cannot do much per-packet computation.
- (C7) Routers are not compromised.
- (C8) All routers have to participate.

These constraints will be used to evaluate existing proposals. We classify these proposals into 5 classes according to the type of technique used: Ingress Filtering, Packet header marking, Control plane based approaches, Packet logging, and other approaches. The main techniques within each class are described below.

4.4.1 Ingress Filtering

Ingress Filtering is historically the first method that has been proposed to prevent DoS attacks. The Ingress Filtering approaches are mainly preventives and can be very effective at preventing address spoofing by attackers. Although techniques used by Ingress Filtering approaches are different they share a common limitation: Ingress Filtering does not provide any direct protection to the entity implementing it. The implementation of such a technique only improves the protection of the whole network by prohibiting address spoofing in the restricted part of the network. This means that there is usually little direct incentive for an entity to implement Ingress Filtering.

4.4.1.1 Network Ingress Filtering (C1, C2, C3, C6, C7, C8)

[BCP38] presents a first approach to avoid distributed denial of service attacks. As mentioned in section 4.1 most denial of service generation tools generate packets with fake IP addresses in order to hide the real origin of the attack. As a result checking the source address at the network ingress point can prevent attackers from taking addresses of hosts located outside of their own network (network A in our example) therefore limiting their ability to hide their identity. Note that depending on the nature of network A, attackers may still spoof addresses as long as they belong to network A. As a result the efficiency of Network Ingress Filtering increases as it is implemented as close as possible to the source. Note that this is not always possible, in particular in the case of a shared physical support (e.g. Local Area Networks, Cable networks). Filtering Ingress Network is usually quite easy to implement on routers through access control lists.

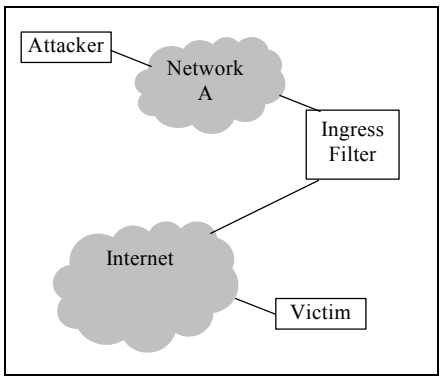

Figure 4-5: Network Ingress Filtering Location

However this measure has several limitations:

- Some Internet usages such as mobile IP require routers to route packets that do not originate from hosts with home addresses. A proposal has been made to avoid this problem by tunneling packets from visiting hosts to their home network, however this proposal comes in contradiction to other proposals aimed at optimizing packet paths for mobile nodes in the network and is not widely deployed.
- Depending on the position of the filter, ingress filtering may cause problems with DHCP or BOOTP.
- Network Ingress Filtering becomes efficient only when everybody implements it on the Internet.
- Network Ingress Filtering is usually useless against encapsulated packets.
- Attackers may generate attacks by using legitimate addresses and legitimate flows.

4.4.1.2 Unicast Reverse Path Forwarding (C1, C2, C3, C6, C7, C8)

Unicast Reverse Path Forwarding (uRPF) [Cis00] is proposal to provide a service similar to Ingress Network Filtering. When uRPF is enabled the router uses its forwarding table to check that each incoming packet is coming from the "right" interface. Three notions of a "right" interface for a given packet exist today.

In the case of *strict* uRPF this interface is defined by the interface to which the packet would have been sent if it had been sent from the destination to the source. When a packet is received by the inbound line-card forwarding engine, input ACLs are first checked, the engine then performs a reverse lookup in the FIB (Forwarding Information Base) in order to see if the packet has arrived on one of the best return path to the source. The engine then performs a regular address lookup for packet forwarding. The packet is then sent to the outbound interface.

In the case of *loose* uRPF this interface is defined by any interface that is included in the FIB for the source address carried by the packet.

In the case of *feasible* uRPF the right interface is an interface including one of the best route to the source. The best route set includes a limited (2 or 3) number of routes for which the packet may have come from the source using less hops. For example if we suppose that DoS packets are sent from A to B and that fives routes R1, R2, R3, R4, R5 with route lengths of respectively 1,3,8,2,4 hops are available on router C for interface C1, the packet will only be forwarded if the inbound interface FIB includes R1, R4 or R2. Alternative routes are stored in the FIB similarly to the best route during the FIB configuration process.

The main advantage of uRPF over Network Ingress Filtering is to prevent a lengthy access control lists configuration process of each edge router by implementing an automated filter configuration process that derives configurations from routing tables. uRPF is usually implemented in hardware thus preventing performance issues.

Similarly to INF the efficiency of uRPF depends on the position of the uRPF filter in the network. Moreover strict uRPF cannot be used with asymmetric routes or multi-homed networks since best paths may differ from one router to the other. On the other hand loose uRPF does not bear this limitation but provides a limited protection. As a result Cisco recommends using strict uRPF in enterprise networks with a single connection to an ISP or in Network Access Servers and loose uRPF in the other cases. Similarly to INF, uRPF is not able to handle encapsulated IP packets. Finally feasible uRPF represents a tradeoff between the two other techniques that may work better in some situations.

Note that juniper offers strict and feasible uRPF on its routers, while Cisco offers loose and strict uRPF.

4.4.1.3 Park et al. (C1, C2, C3, C4, C5, C6, C7, C8)

[Park01] presents a mechanism called DPF (Distributed Packet Filtering) aimed at preventing and detecting DoS attacks. Although the authors present their contribution as independent of uRPF, the two proposals seem to share several common aspects. The general idea is to avoid a per-router mechanism but to focus instead on a per peering-point function. By cleverly choosing peering-points it becomes feasible to prevent most address spoofing attacks with a limited subset of routers.

They show that Ingress Packet filtering is not effective even when a very large proportion (95%) of ingress routers implement it. Consequently combining INF and trace back or packet marking with ingress filtering does not lead to a large improvement in the ability to identify attackers correctly. As a result they suggest moving filtering functions in peering points in order to prevent AS addresses spoofing between ASes. They show that given a map of AS interconnections, it is possible to compute a set of filters that can prevent AS spoofing. They also show empirically that the placement of filters on peering-point should follow a vertex cover in order to get a maximum improvement. They then demonstrate through examples that the coverage (proportion of peering-points) needed to provide a significant protection (88% of AS cannot be used to generate DoS attacks) against AS spoofing can be as low as 19% with 1997 to 1999 Internet Maps. They also show that the availability of several routes between AS decrease the efficiency of a peering-point based filtering approach. However even with a complete loose routing the protection remains over 70% in the same conditions (Vertex Cover, Real Internet Map). Finally, they suggest combining trace back packet marking and AS based packet filtering in order to increase the probability to identify attackers correctly.

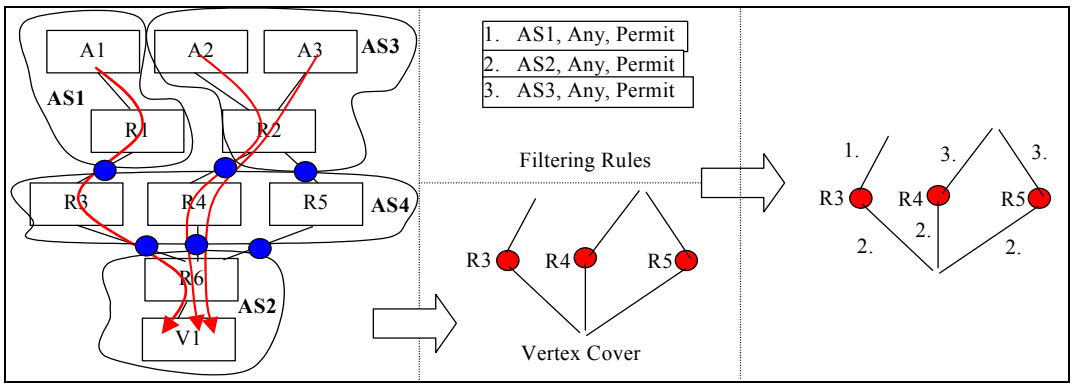

Figure 4-6: AS Filtering Example

A few limitations should however be noted:

- The identification of attackers is based on AS identities. This is somehow different from route-based identification where the complete route is expected to be found.
- They assume that every AS is willing to implement filtering. This is unlikely to happen in a real network. The distribution of filtering AS peering point is more likely to be random. Under random conditions results are far from being so good, even when a large proportion (50%) of AS peering points take part to the filtering process.

The implementation supposes that the whole internet-topology is available in order to compute filters and the vertex cover. This is usually false since many routing protocols do not include source reachability.

4.4.2 Packet Header Marking Approaches

The packet overwrite approach consists in overwriting one or several fields in the IP packet in order to store information about the path taken by each packet from its source to its destination. The destination is expected to retrieve this information to reconstruct the path and identify the attacker. The efficiency of existing packet header overwrite approaches can be evaluated according to several parameters such as the number of marked packets required to build the path to the attacker, the accuracy of the path (probability of the path being correct), the number of paths that can be detected simultaneously, the resources (spatial and temporal complexity) used to compute the path at the destination, the resources used to compute the path code in each router and the need for additional information such as the network topology.

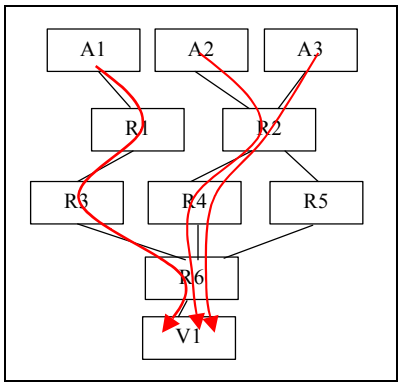

Figure 4-7: Topology Example

4.4.2.1 Doeppner et al. (C1, C2, C3, C4, C5, C6, C7, C8)

[Doe00] suggest extending the IP header with a field where routers could code their address as well as the source interface for each packet. In order to prevent a variable length field that would extend at each router, the authors suggest using a fixed size field. In order to prevent attackers to fill the field in advance, the authors also suggest filling the fields in a non-deterministic manner so that an attacker cannot fill the fields in advance. The number of slot in this field is called s.

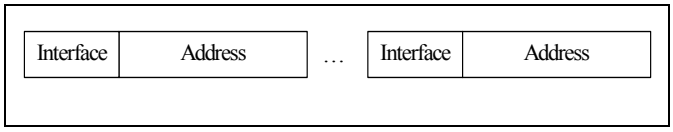

Figure 4-8: Marking Fields Doeppner et al

In order to do so, a probability p = 1/r is chosen by all routers where r is the maximum number of routers between two hosts in the network. When the router receives a packet another probability (x) is chosen. When x < sp the slot [x/p] is used to store the timestamp identifying the router. Since x is chosen probabilistically several routers on a path can overwrite slots. However a probability of a slot not being overwritten can be computed thus giving the number of packets to be received in order to know each router on the path. In order to reconstruct the path from a source to a victim, the authors suggest using a map of the network. Finally the authors suggest changing the stamps periodically and probabilistically in order to prevent attackers from faking stamps to create false attack paths.

The proposal bears several limitations. The biggest one is that attackers can easily bypass the last operation by retrieving packets marked by routers in the network. This can be done generating communications that would pass through these routers.

Another problem is the implementation in existing routers. Packets including such a field would be very similar to IP packets using the RR option. As a result it is very likely that these would have to be treated similarly to IP packets with options. IP packets with options are usually treated by a general-purpose processor, instead of being treaded by a specialized circuit. As a result the performance of such approach may be low.

Finally it is not clear how such approach would behave in the case of multiple attackers.

4.4.2.2 Savage et al. (C1, C2, C3, C4, C5, C6, C7, C8)

[Sav00] suggest coding edges in the ID IP packet field. In this scheme called FMS (Fragment Marking Scheme) an edge is made of two adjacent network nodes identifiers and the distance between the device monitoring the packet and the more distant node. For example in the network example provided above an edge may be constituted with (R1, R3, 2) to describe the link between R1 and R3. Since several network nodes may be willing to code edges in the packet, a scheme is given that allows:

- Nodes to determine if a start edge has already been defined.
- Nodes to fill edge information in a probabilistic way.

As a result a monitor located between R6 and V1 receives packets with edge information and can reconstruct the path between attackers and V1 by performing edges concatenation. However since the ID field is only 16 bits large, edges identifiers have to be compressed. In order to perform this compression, the authors suggest two different techniques:

- XORing edge start (s) and end (e) addresses in order to divide the space required by addresses by two. The monitor is still able to find the two addresses by computing s xor e xor s = e and s xor e xor e = s.
- Dividing the address field into k non-overlapping fragments that are sent separately with the position of the fragment. This scheme divides the size required for address coding by k but requires the storage of the position of the fragment (cost is log(k)). Moreover this scheme increases the number of packets necessary to reconstruct the path by a factor k.

In order to avoid collision between edge-id (decrease the probability to have several edges producing similar edge fragments – this can happen because of the IP address structure) they also suggest appending an error detection code to each address. The monitor uses this error detection code to check if an address slice can be associated to an existing edge. This last scheme is called CEFS (Compressed Edge Fragment Sampling).

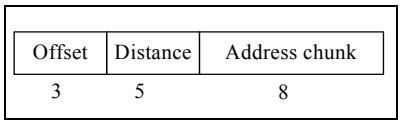

Figure 4-9: Marking Fields for Savage et al

The general idea is very interesting, however this scheme suffers from several problems. [Son01] shows that this scheme usually requires a lot of packets to perform the identification of the path correctly. It also demonstrates that the FSM approach is not able to handle a large number of simultaneous attackers. For example tracking 25 simultaneous attackers may require several days of computation because of the temporal complexity of the path reconstruction algorithm. Moreover it can also result in a large number of false positive results (legitimate users are identified as attackers).

Finally [Wal02] presents a new way to generate collisions between marking fields that can be used to fool CEFS. These collisions are relatively easy to generate because the error detection code generated by routers is based on a field that the attacker can modify. As a result the attacker does not need to generate a collision (two addresses resulting in similar marking field) but only a *near collision* (two addresses resulting in two valid marking fields given their previous values) by the modification of the edge and the error code simultaneously. Such combinations can be easily found using brute force attacks.

4.4.2.3 Song et al. (C1, C2, C3, C4, C5, C6)

[Son01] while keeping the same approach, suggests several improvements over the FMS approach:

- Propose to code the edges using a hash code (instead of creating address chunks) to reduce the address size from 32 bits to 8 bits and the XOR function to combine start and end edge addresses.
- Since the hash function can result in collisions (ie several addresses being coded with the same hash code), the authors suggest decreasing the probability of collisions by using several hash functions on each node, the hash function identifier being coded as an additional field (flag_id). Decreasing the probability of collisions is expected to decrease the number of false positive.
- In order to improve the temporal complexity of the path reconstruction algorithm, the authors propose to use a network map to pre-compute possible edges.

- Finally, they suggest combining address hashing with a secret key to provide network nodes authentication. The authors also provide a way to handle the key distribution problem.

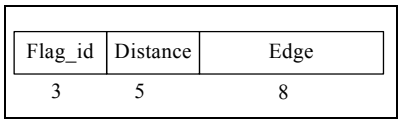

Figure 4-10: Marking Fields for Song et al

Simulation results given in the paper show that this approach would be able to handle attacks by thousands of attackers in seconds, with a reasonable number of packets when the path between the attacker and the victim, even when the path between the attacker and victim is as long as 30 nodes.

However the improvements in the reconstruction algorithm is mainly based on the ability to get a map of the whole network, which may be a problem.

4.4.2.4 Dean et al. (C1, C2, C3, C4, C5, C6, C7)

[Dea01] also keeps the same approach but suggest encoding the path between the attacker (A1) and the victim (V1) as a polynomial expression. If we suppose that each router i is able to code its address as R_i, then each router on the path can compute a polynomial expression $P_{i,j} = P_{i-1,j}.x_j + R_i$, where x_j is a random value chosen for packet j and $P_{i-1,j}$ is the polynomial expression transmitted from router R_{i-1}. If the degree (i.e. the number of routers on the path) of the polynomial expression is d then the victim only need d packets to obtain a full rank matrix equation that can be solved in $O(d^2)$. Solving the matrix equation provides the value of each Ri on the path.

Example:

Let's consider the path between A1 and V1 in our example: We suppose $P_{0,x} = 0$ and that $R_i = i$,

R1 computes $P_{1,1} = P_{0,1}.x_1+R_1 = R_1$.

R3 computes $P_{2,1} = P_{1,1}.x_1+R_3 = R_1.x_1 + R_3$

R6 computes $P_{3,1} = P_{2,1}.x_1+R_6 = R_1.x_1^2+ R_3.x_1 + R_6$.

After 3 packets we end up with three equations at V1:

$P_{3,1} = P_{2,1}.x_1+R_6 = R_1.x_1^2+ R_3.x_1 + R_6$

$P_{3,2} = P_{2,2}.x_2+R_6 = R_1.x_2^2+ R_3.x_2 + R_6$

$P_{3,3} = P_{2,3}.x_3+R_6 = R_1.x_3^2+ R_3.x_3 + R_6$

If we assume that $x_i = i$ we end up with the following equation:

$R_1.1 + R_3.1 + R_6 = 10$

$R_1.4 + R_3.2 + R_6 = 16$

$R_1.9 + R_3.3 + R_6 = 24$

Which fortunately results in $R_1 = 1$, $R_3 = 3$ and $R_6 = 6$.

Authors then present several improvements. In order to avoid the $P_{0,x} = 0$ hypothesis, routers can decide probabilistically that they are the first hop in the path which result in a partial encoding of the path. They also propose that routers encode only edges (just like [Sav00]) to reduce the size needed to store the polynomial expression.

Finally the authors present existing algorithms that may be used solve the path recovery problem when the above conditions are met. They also consider the case where faked information is sent to the victim in order to fool path recovery schemes. However it is not clear if proposed algorithms perform better than [Sav00] in the general case since algorithms presented in the paper that perform better than [Sav00] require a field larger than the ID IP packet header field.

4.4.2.5 Adler (C1, C2, C3, C4, C5, C6, C7)

[Adl02] presents several additional proposals to mark packets. The article is interesting for several reasons. First it provides lower bounds (complexities of the best case scenario) for packet marking algorithms in the case of single-path and multi-paths attacks. These results are particularly interesting since they are based on a set of conditions that is very easily met. It also proposes three protocols that improve existing coding schemes in some particular situations. On the other hand, the article is rather theoretical and finally shows that the coding scheme is not so interesting when all conditions used to compute upper bound complexities (complexities of a real case scenario) are relaxed. This scenario unfortunately is the one met in the real life.

Previous approaches used to encode the path in a set of bits in the IP packet header. The basic idea behind the article is somehow different since it suggests that the victim could compute the path(s) to the attacker(s) not by using the value of such a field but by using the frequency at which the value appears. This frequency can be used to compute a probability distribution allowing the value of each router identifier to be computed with a certain probability.

For the upper bound problem, the whole article is based on a basic model of the network where the victim is connected to one or several attackers by a binary tree. The victim is supposed to have a complete knowledge of the network topology and each node of the tree (router) is supposed to have a complete knowledge of its (3) neighbors. The path between the victim and one attacker can be described as a binary string where each bit designate the link to follow from one node to its son in the tree.

For the lower bound, the whole article supposes that the model is made of two entities: The network and victim. The first one tries to send the second one the attack path(s) on a limited number of bits.

For each model, the number of bits that can be stored in a packet is noted b and the number of bits required coding the path between the most distant attacker and the victim is noted n.

The article considers four different scenarios:

- Where b equals 1 and a single attack is performed.
- Where b>1 and a single attack is performed.
- Where b=1 and a several attacks are performed simultaneously.
- Where b>1 and a several attacks are performed simultaneously.

Although the paper mentions a single attack scenario, the scenario that is actually treated in this case is a single flow scenario. That is to say: A single flow (from the attacker to the victim) is sent in the network. The term single attack is therefore a little bit misleading.

The main idea of scenario 1) is that each node N can code the value of the link L from which the packet has been received (1 or 0) in the packet depending on the value of the bit B and a given probability (r) so that N sets B to 1 with a probability 1 if B=1 and L=1, with a probability 0 if B=0 and L=0 and with a probability r in the other cases. Given this coding scheme, the victim can determine the path by comparing the frequency of the 1 and 0 bit values with a set of possible frequencies.

For example, let's consider the diagram below where a victim is connected to six nodes and r=1/2. The probability that a packet from A1(11), A2(10), A3(01), A4(00) to reach V1 with B=1 is respectively 3/4, 1/2, 1/4 and 0 (if we suppose that B is set by the attacker to 0). As a result is the proportion of B=1 packets reaching V1 is 1/2, V1 can deduce that the path is 10.

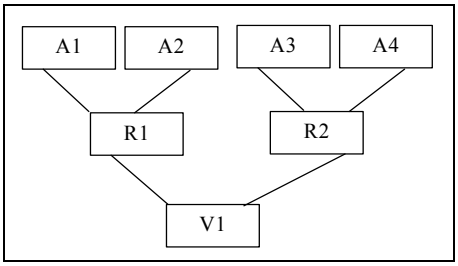

Figure 4-11: Network Example (one bit code)

The author also shows that the scheme can be modified to accommodate the case when the attacker modifies the B bit. The scheme requires $O(2^{2n})$ packets to allow a full path recovery.

The article shows that the lower-bound in this case is slightly better with $O(2^{n})$ packets.

The main idea of scenario 2) is to partition the network in d sets of nodes and to apply the one bit scenario in each set. In order to differentiate operations made in each set, the set identifier is coded on b-1 bits and the last bit is used to perform the one bit protocol within a given set. The number of packets required for this protocol is much better with $O(bn^2 2^b 2^{4n/2\hat{}b})$ packets.

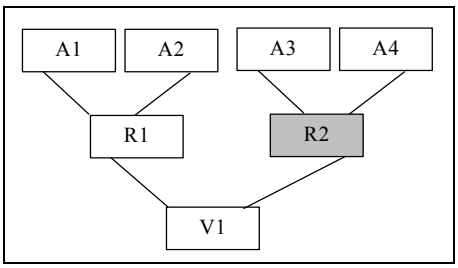

Figure 4-12: Network Example (2 bits code)

For example if we consider the case where b=2. If we code the zone as a color (B[1]=1 means gray, B[1]=0 means white) the probability that a gray packet from A1, A2, A3, A4 will reach V1 is respectively 0,0,1 and 1. The probability of a packet from A1, A2, A3, A4 to be received with B[0]=1 is respectively 1/2, 0,1/2 and 0. As a result if the proportion of gray packets with B[0]=1 reaching V1 is 1/2 the victim knows that these packets are coming from A3. Note that the number of packets needed to recover the correct path in this case is only two against four in the previous case.

The authors also shows that the lower-bound in this case is $O(2^b 2^{n/2^b})$ packets.

The author shows that scenario 3) cannot be solved regardless of the number of packets that may be sent to the victim whether using the upper bound or lower bound scenario. More generally the author shows that in the case of the lower-bound scenario the multi-path problem cannot be solved if b<log(2k-3)+1 where k is the number of path of attacks.

Finally in scenario 4) with upper-bound restrictions the author suggests that each path may be encoded in b=log(2k+1) bits of the header. The author proposes a way to encode information into B such that the receiver can compute for each input header he receives an output header with a certain probability. By choosing carefully header transitions and providing to the victim the knowledge of the probability of each transition in the network, the victim is able to compute the distribution of each possible path by resolving a 2k full rank Vandermonde matrix after receiving $O((k^2 2^{(2k^2+k)(n+2)})^2 \ln(2k))$ packets. Resolving a 2k full rank Vandermonde matrix can be done in $O(4k^2)$.

4.4.3 Control Plane Based Approaches

Historically control path approaches have been the first to be proposed. Control path approaches bring several improvements over packet header rewrite. First, control path approaches do not require changes in the semantic of existing packet header fields. This is important since rewrite may change the semantic of packets (for example [Dea01] suggests to use the field TOS in addition to the field ID which may generate a problem with diffserv). Even in proposals that only use the ID field, problems may occur when fragmentation is performed in the network or when the IPsec AH is used. It is also not very clear if packet rewrite approaches would handle IPv6 packets since a) larger addresses would result in a much larger collision rate or complex path recovery algorithms, b) IPv6 does not include an ID field. Another point is that packet rewrite approaches require ultra-fast in-routers path computation functions since each packet has to be "signed". It is not clear if proposed functions can be sufficiently fast for existing and upcoming terabit routers. Finally packet rewrite approaches are relatively useless against attacks using deflectors since the marking information is lost at the reflector.

Oppositely control path approaches suggest transmitting information about DoS in additional packets. These packets should be sent at a much lower rate than packets handled routers fast forwarding paths. We classify existing control plane proposals in two classes.

4.4.3.1 ICMP Based Approaches

4.4.3.1.1 ICMP Traceback (C1, C2, C3, C4, C5, C6)

[Bel01] suggest that routers on the path between the attacker and the victim should help the victim tracing packets by sending with a low probability a trace back message that is sent along with the original packet to the destination. By using the information sent in the trace back message, the victim is able to associate a DoS packet with trace back messages and to recover the path to the attacker by looking at routers addresses.

From a protocol point of view, the trace back packet can include one or two edges. Each edge can be described through a MAC, IPv4 or IPv6 address pair or an interfaces/link identifier. Trace back packets also include a timestamp, a part of the content of the traced packet, the probability and the router identifier.

[Bel01] also propose to use authentication to prevent attackers from generating faked trace back messages. Authentication is performed through a specific authentication field in the trace back

packet including a key identifier a timestamp and the authentication data. Authentication keys can be distributed by revealing keys after a pre-defined time period in the trace back packets. Consequently, trace back messages can only be authenticated afterward, once the authentication key has been revealed.

I-Trace suffers from several problems. First, [Man01] shows that I-trace performs poorly against largely distributed DoS when the number of packets generated by each attacker is small. If we come back to our example and suppose that A1, A2 and A3 generate each 1 pkts/s toward V1, R1 is likely to generate a trace back packets 3 times slower than R6 and 2 times slower than R2. As a result V1 is likely to learn about R1 a long time after learning about R6. Moreover if the probability to generate a packet in each router is low, small flows can take a lot of time to generate a packet. For example, with a probability of 1/20000 a 1 pkts/s flow in R1 will generate a trace back packet after an average of 20000 packets or more than 5 hours.

4.4.3.1.2 Intention based Traceback (C1, C2, C3, C4, C5, C6)

In order to cope with the long paths recovery problem, [Man01] and [Ma01] suggest modifying the probability to generate packets according to the content of the packet (c0), the distance between the node and the victim (c1), the need expressed by the victim for trace back packets (c2), the number of trace back messages already sent to the victim (c3) and the global number of trace back messages generated by the whole router (c4). The general idea is to separate packets in two classes:

- Class A: Packets targeted at addresses requiring trace back messages.
- Class B: Packets targeted at other addresses.

Although the general probability to generate a trace back packet would still be fixed and low (1/20000 as proposed in [Bel01]), the probability (Pa) to generate packets for Class A addresses would be increased and depend on c0, c1, c3 and c4. At the same time the probability (Pb) to generate packets for Class B addresses would decrease so that Pa + Pb = 1/20000.

From an architectural point of view, the decision to generate a trace back packet is taken by a software component called decision module. The generation of the trace back packet itself is done by an external component called itrace generation module. The architecture of each router is modified so that the decision module is able to indicate to the router line-card which packet has to be copied to the itrace generation module. This is done by modifying a bit in the forwarding table.

Although the decision module can obtain the c0, c2, c3 and c4 parameters locally, c1 has to be obtained from the victim. [Man01] suggest distributing the intention victims through BGP routing information by setting the BGP “community attribute” to “true” when the victim desires to receive trace back messages and “false” in the other case.

4.4.3.1.3 Reverse Traceback (C1, C2, C3, C4, C5, C6)

[Bar01] is a proposal to extend the first two trace back proposals in order to deal with attacks using reflectors. The main reason why the two first proposals are not able to handle reflectors is that trace back messages are sent from a router on the path between the reflector and the victim to this last one therefore allowing the attacker to hide his location by hiding the path between the reflector and himself.

In order to solve this problem [Bar01] suggest sending reverse trace back messages. These messages would of course be generated by routers on the path between the reflector and the victim (R2, R4, R6 in our example) but also between the attacker and the reflector (D1). Since

the attacker has to modify his address to make the reflector believe his request is coming from the victim, reverse trace back packets would be sent on routers on the path between the attacker and the reflector to the victim therefore allowing him to locate the attacker.

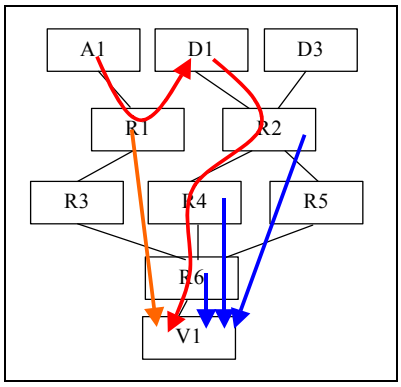

Figure 4-13: Reverse Traceback

4.4.3.1.4 Active Traceback (C1, C2, C3, C4, C5, C6)

[Yam02] presents a protocol to control the behaviour of traces generators. The protocol also allows trace-back clients to "proxy" their trace-back requests to routers outside of their domain. This is useful when clients are located behind a NAT device and authentication between generators and clients is requested. Is it not obvious what advantage would be brought by such architecture compared to other trace-back approaches. In particular the proposal suggest choosing the set of trace generators that should be controlled which may limit the usefulness of trace-back messages. It is also not clear how these generators should be identified.

4.4.3.1.5 Pushback (C1, C2, C3, C4, C5, C6)

Finally [Mah01] suggests another method to detect and limit the effect of DoS attacks. This proposal is not specifically targeted at DoS attacks but instead aims at controlling high bandwidth aggregates in the network. A high bandwidth aggregate is a set of packets that share similar properties and use a significant proportion of the physical or logical bandwidth available at a network node interface. [Mah01] distinguishes two main sources of high bandwidth aggregates: Flash crowds and denial of service attacks.

Oppositely to previous proposals [Mah01] does not only propose a solution to identify sources of packets but also suggest a method to identify DoS attacks and a way to control these aggregates in the network therefore allowing the effect of DoS attacks to be controlled. The basic idea is to extend existing network nodes with the ability to analyze output queues drop packet rates in order to detect congestions.

Note that having a description of high bandwidth aggregates is not sufficient since some DoS attacks may be generate high packet loss rates with relatively low bandwidth flows. This is the reason why packet drop rates and natures are preferred over a simple definition of high bandwidth aggregates.

Once congestion is detected, the node (R) is expected to identify the highest bandwidth aggregate (A). This bandwidth aggregate is then rate-limited before the output queue. Once rate-limited, future evaluations of A will therefore include the bandwidth A at the output queue but also the bandwidth dropped by the rate-limiter. The process to identify new high bandwidth aggregates responsible for packets drop continues as long as the output queue is experiencing a significant packet drop rate.

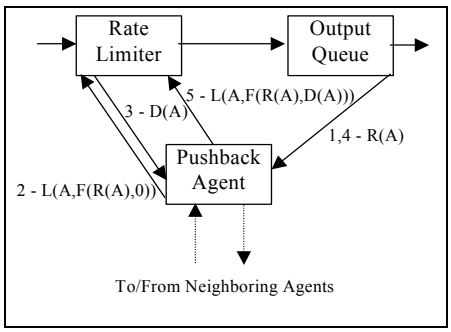

Figure 4-14: Pushback Agent Process

The node is then expected to identify neighbor upstream routers (NRi) that forwarded the largest part of A and identify each neighbor share to the total traffic (SRi). Once identified, the node sends a message to each NRi including a description of the flow (F) to be controlled and the rate limit to be enforced. This message is called pushback message. Neighbor nodes use this information to rate limit the aggregate. Neighbors also send periodically a report including the number of bytes carried by packets by F to R so that R can evaluate the amount of traffic that would have reached him if no rate-limit had been in place. This information allows him to send updated pushback requests. NRi nodes are also expected to use a similar scheme in order to notify their upstream neighbors so that pushback messages can flow upstream in the direction of attackers as described in the figure bellow. Finally NRi nodes stop the rate limitation process for the flow F if they do not receive an update from R after a predefined amount of time.

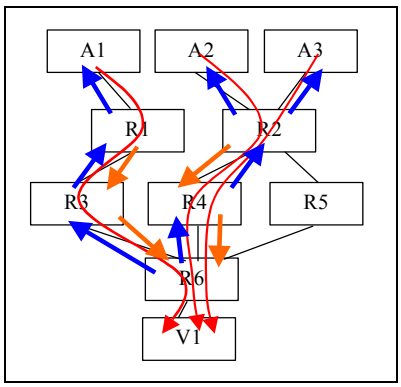

Figure 4-15: Pushback Messages Propagation

Simulations presented in the paper show that the proposal brings interesting results. However scenarios presented in the paper only include a small number of attackers (64). It is not clear if the same scheme could be used with a larger number of attackers.

[Ioa01] presents an implementation of the pushback architecture using FreeBSD and IPFW. Showing that the architecture could be easily and effectively implemented.

This solution has several advantages over previous proposals: The implementation of this proposal would usually not require any modification to existing routers since information such as the output queue congestion rate or mechanisms such as rate limit are already available in most routers. Pushback is also the only proposal that tries to combine a DoS prevention proposal with the DoS detection.

This solution also has several limitations. It is not clear if high bandwidth aggregates describe all existing DoS attacks. For example [Mo01] explains that DoS can be performed using a packet

rate as low as 500 pkts/s. These aggregates would not be considered as high bandwidth aggregates in [Mah01] and would therefore not be controllable. It is also likely that this solution will not be able to identify attacks paths when attacks are composed of a large number of highly distributed very small flows for the same reason.

As a result it would be interesting to understand how approaches like intention-based trace back and pushback could me merged to provide attacks prevention in the case of thousands of attackers.

Finally pushback may represent a good tool to protect the network itself. Routers usually lack means to detect sophisticated DoS attacks but still represent a valuable target for attackers as shown in [Mo01]. Pushback may be an adequate mean to avoid router control path congestion.

4.4.3.2 Routing Based Approaches

Existing routing based approaches focus on using routing protocols to propagate tracking request or DoS suppression information. Due to the propagation of routing information a common limitation to all routing based approaches is that the reactivity of these approaches is usually lower than other existing proposals. Moreover the amount of resources used to prevent attacks in these cases is usually quite large. As a result we envision the use of such approaches mainly in the case of large and long attacks.

4.4.3.2.1 CenterTrack (C1, C2, C3, C4, C5, C6, C7, C8)

[Sto00] suggests creating an overlay network in order to redirect DoS attacks to a router where the attack can be analyzed and the attack origin can be located. The basic idea is to build tunnels between each edge router in the network and one or several central routers. These routers are called Tracking Routers.

When an attack is detected a signature of the attack is constructed by the victim and sent to the network operator. The traffic directed to the victim is then redirected through a modification of the routing topology from edges routers to central tracking routers using existing tunnels. Input debugging is then performed on the tracking router the closest to the victim in order to know from which ingress edge router the attack is coming from. In the case of a single level topology (each edge router would be directly connected to a single tracking router) the operation is quite simple. However in the case where several tracking routers have to be used, the operation has to be repeated hop by hop until the edge ingress router is found.

The author also provides a very complete description of the configuration of the tunnels and the routing topology to be implemented making the paper very valuable from a practical point of view.

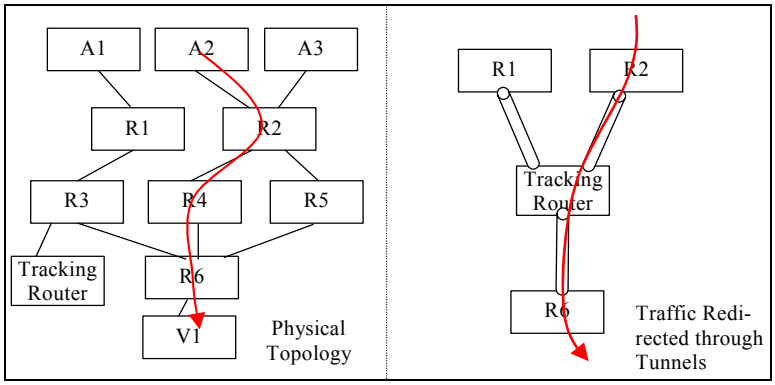

Figure 4-16: Centertrack Architecture

The main advantage of this proposal is that it does not require any hardware or software modification in existing routers and as a result can be implement as is. Moreover it appears that the approach has been widely tested by UUNET, allowing the efficiency of the method to be assessed. On the other hand this approach also bears several limitations. As mentioned by the author, it is unlikely that CenterTrack will be able to track widely distributed attacks. This approach is also unable to deal with attacks targeting backbone routers. Finally the centralized approach used in CenterTrack makes it difficult to use in the case of a large number of simultaneous attacks.

4.4.3.2.2 Blackhole routing (C1, C2, C3, C4, C5, C6, C7, C8)

The main goal of blackhole routing is to divert traffic from a victim to a router located in the network operator network by updating the routing information in the network. When an attack is detected, the traffic is redirected to a tracking router where the traffic can be analyzed.

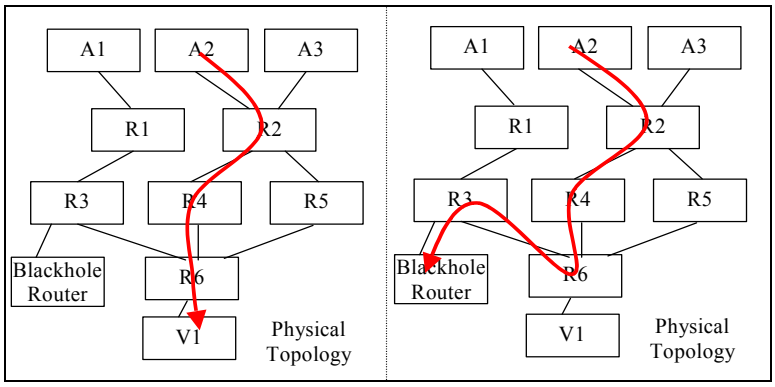

Figure 4-17: Blackhole Routing

When attacks are sufficiently long, the originating edge router can be tracked by using one of the techniques presented in the following section.

The redirection of the traffic can be done either by adding a static route on each edge router, however this technique is obviously not very scalable. As a result a more scalable solution can be achieved by using iBGP to distribute a new route to the Blackhole router.

The previous technique can be improved ([Beh02]) by dropping the traffic at edge routers directly in order to avoid DoS traffic in the network. This can be done by setting up a local static route to an unused address (X) on each edge router so that the traffic will be rejected (it can be the null0 interface on Cisco routers) and to announce with iBGP a next hop X for the victim address. The result is that each edge router, when receiving BGP announces rejects the traffic directed to the victim.

4.4.3.2.3 Blackhole routing with ICMP backscatter (C1, C2, C4, C5, C6, C7, C8)

Blackhole routing with ICMP backscatter ([UU02]) is an extension to the previous blackhole routing technique allowing an easier way to track edge routers through which DoS traffic is flowing. As its name implies ICMP backscatter uses a technique similar to the one presented in [Mo01] by supposing that source addresses are spoofed randomly. Note that this technique does *not* work with legitimate source addresses or when the attacker does not select completely random spoofed addresses.

In order to identify ingress edge routers an unused part of the address space (we take the case of a /16 network and call it w/16 in our diagram) has to be routed to the blackhole router. Filters

have to be installed on edge routers to drop the traffic to the victim. As in the previous case, this can be done either using static routes on the edges or by distributing a route to an unused address through BGP. When the traffic is dropped, ICMP unreachable messages will be sent to the sources. As a result DoS traffic with a source address within the w/16 address space will generate an ICMP unreachable message with the edge router address as source address that will be sent to the tracking router. The tracking router only needs to log incoming ICMP unreachable packets in order to find edge routers addresses.

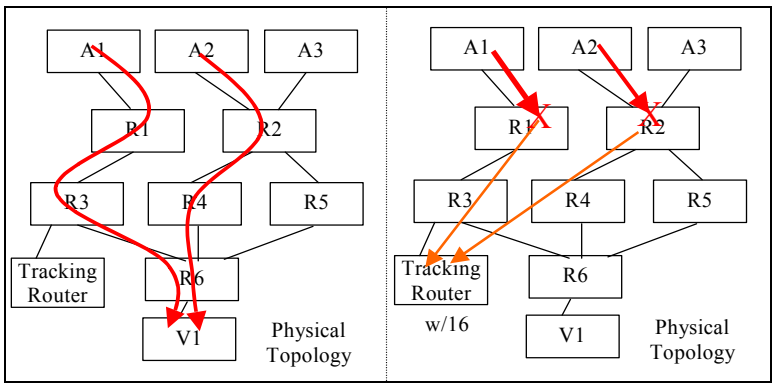

Figure 4-18: Blackhole Routing with ICMP Backscatter

4.4.3.2.4 Destination Class Usage (DCU) with BGP (C1, C2, C3, C4, C5, C6, C7, C8)

Destination class usage ([DCU00], [DCU02]) is an accounting mechanism developed for Juniper routers since JunOS 4.3. DCU allows the amount of traffic flowing to a specific AS or a set of ASes to be measured. In order to do so a specific AS or a set of ASes is identified with a community string. Each router has the ability to log the amount of traffic for 16 different sets of ASes by attributing a counter to each of them on the line card processor. This information can be retrieved remotely using SNMP and the JUNIPER-DCU-MIB. In the case of DoS this mechanism can be used by starting the logging mechanism through BGP by announcing the address of the victim with the pre-configured community string. SNMP counter have then to be retrieved. Traffic amounts have then to be sorted to extract the biggest contributors the traffic flowing to the ASes set. These edge routers may sometime be considered as the DoS entry points in the network.

Note that this technique does not automatically provide a relevant answer to identify the ingress edge routers in the case of DoS attacks since large traffic amounts at the customer level may be hidden at peering point levels.

Moreover this technique does not allow a precise definition of the attacks pattern since in only relies on the number of packet/bytes. Additional techniques have to be used at the edges to reach such a precise definition. Such a definition is essential to suppress specific attacks.

4.4.3.3 QoS Policy Propagation with BGP (QPPB) (C1, C2, C3, C4, C5, C6, C7, C8)

QPPB is a functionality proposed by Cisco ([Cis98]) since IOS 11.1(CC) allowing traffics belonging to an AS or set of ASes to be treated differently from a QoS point of view. Each set of ASes is either identified by a community list, an AS path or an access list. When announced with BGP with the relevant identifier, the traffic directed to the set of ASes can be controlled with CAR or WRED functionalities in order to implement a QoS policy.

This technique can be used in the case of DoS in order to easily limit the traffic directed to a specific destination by first configuring a CAR policy on each edge router. An unused QoS

group has then to be created on each edge router to which the CAR policy is attached. Finally the victim network is announced with the QoS group identifier (community list, AS path, …) when a DoS attacks occurs. When receiving BGP announces the traffic directed to the destination becomes rate limited.

Note that this technique is only useful to limit the effects of a DoS attack.

4.4.4 Packet Logging

Packet marking and control path approaches can be fooled by attackers since the victim requires the attack to reach a certain size in order to allow the path between the attackers and the victim to be recovered. By increasing the number of paths between the attackers and the victim, the rate at which packets are being sent, the content of the packets, the number of victims being attacked at the same time and the amount of legitimate traffic being generated by attackers, the attacker is able to decrease the probability for the victim to discover his real location by reducing the difference between legitimate and malicious packets ([Park00]). Packet logging approaches take a different path supposing that each packet is potentially dangerous and has to be "recorded". However keeping track of all packets is a very complex and therefore demanding operation. As a result several approaches have been proposed to "summarize" packet traffics.

4.4.4.1 Router Packet Logging Approaches (C1, C2, C3, C4, C5, C6, C7, C8)

Cisco proposes several approaches [Cis99] in order to provide a manual way for network operators to trace attacks in their network by looking at packet logs. Depending on the IOS version, several paths may be followed. Before IOS version 11.2 four options are available:

- Counting packets through access lists commands without logging. In order to reduce the overhead generated by packet logging it has been suggested to avoid packet logging but instead to approximate the attack signature by using access-lists with counters. The goal is to start with a large definition of the attack and then reduce the possible set of signatures by specifying values for the different fields that be used to identify the attack. Once identified the access lists can be applied to several interfaces in order locate the originating interface. Looking at the progression of counters can do this. The main drawback of this approach is that it requires a lot of time and efforts to find the origin of packets on a single router.
- Logging packets through access lists logging commands. The log provides the source and destination addresses, the access list matched by the packet and the number of packets of that type received. By looking at incoming and outgoing line-card logs the security officer can determine from which interface the packet is coming from. This approach bears several drawbacks: It has a quite large performance overhead and it does not allow all packets to be logged making it difficult to identify the source interface of attacks. Moreover similarly to the previous approach the attack signature has to be identified first in order to reduce the flow of log messages.
- Using debug commands. These commands provide a level of information similar to the one obtained with access-list log commands. However the overhead is usually higher because amount of information provided is not rate-limited and therefore usually higher.
- Using Netflow ([Beh02]). Netflow can also be used to find the incoming interface of each flow to a specific destination. The main advantage of using netflow over other tracking means is that the performance overhead of netflow is usually lower ([Net02]) when hardware mechanisms are used (engine 4/5 for GSR). However the information is usually less synthetic than the one provided by other means.

After IOS version 11.2 a new extension to access lists called log-input is introduced. This extension provides the same amount of information as the log command with in addition the incoming interface in the case of unicast traffic and the MAC address in the case of a multicast traffic. According to Cisco such command would moreover generate less overhead than the log command.

Juniper [Jun00] suggests a similar approach in order to generate an attack signature and trace the source of the attack within a network.

The main drawback of all these approaches is to require en lengthy attack signature generation process, a hop-by-hop analysis process that has to be performed by hand. Finally the overhead of such approaches is usually quite high.

4.4.4.2 Cisco IP Source Tracking (C1, C2, C3, C4, C5, C6, C7, C8)

[Cis02] presents a mechanism allowing DoS attacks to be tracked. Actually the mechanism is mostly local but would allow DoS attacks to be tracked hop by hop by retrieving information from adjacent routers by hand. The improvement over existing local approaches (input ACLs, debugging) is to provide through a single configurable function the ability to provide for each flow the input line-card from which the flow originates as well as some statistics about the flow. Flows can be selected according to their destination address. The mechanism works as follows. When a line-card receives a packet matching the destination address specified by the user, this packet is forwarded to the line-card CPU where the packet is classified according to the flow definitions. Information such as the incoming interface and the size of the packet are then kept and statistics about the flow are updated. The packet is then processed by the line-card CPU like other packets. The information stored by the line-card CPU is gathered periodically by the router CPU, who generates the relevant interface and statistical information.

This approach is relatively easier than the one relying on logged input ACLs but the DoS source tracking process still has to be performed by hand. The approach also has several drawbacks; first sending packets to the line-card CPU can overflow the CPU and result in a denial of service on the router itself. Due to the large difference in term of performance between the line-card CPU and ASICS based CEF forwarding mechanisms this approach is limited to relatively small flows (<40kpkt/s). The second problem is that this mechanism is unable to trace packets sent to the router itself or to multicast destinations. Finally this router does only work on Cisco 12000 family routers using engine 0,1,2,4 with IOS version earlier than 12.0(21)S.

4.4.4.3 Trajectory Sampling (C1, C2, C3, C4, C5, C6, C7)

Although not specifically designed to provide DoS prevention, [Du00] presents a method to track attacks. The goal is to provide tools to reconstruct the trajectory followed by a packet during a network traversal. In order to do so, the authors suggest computing a hash function on a static part of an IP packet (IP and transport information without Checksum, TTL and TOS since they are likely to change during network transport and version, header length and protocol since they are usually the same for each packet). This first hash function is used to select the *same* packet at several locations in a network. By collecting the locations where such a packet has passed it is possible to reconstruct the trajectory followed by one packet.

A second hash function is then used to send a description of the packet to a collector, in charge of trajectory calculations. The authors examine which parameters have to taken into account in order to optimize the hash functions so that the reconstruction of the trajectory of a given packet takes the smallest hash messages.

The authors also suggest that trajectory sampling may be used to find the source of spoofed packets in the case of DoS attacks by collecting labels corresponding a specific packets and reconstruct trajectory from the victim to the attacker source.

[Du02a] presents an architecture where this concept is used to allow trajectory computation as well as more elaborate request to be performed. This architecture is based on two main components: Routers are expected to compute hash functions on packets to select them. Selected packets are then sent to a collection point. However in order to minimize the traffic between routers and the collection point another hash function can be computed on the resulting packet. The result of this hash function is called label. Such architecture supposes several problems to be solved. In particular:

- How collisions between labels attributed to packets can be limited for a given size of label, number of packets to be labeled and volume of information;
- How timing requirements to accept labels have to be defined in order to limit label collisions for a given network where delays between nodes are known.

[Du02b] is an IETF proposal to standardize this architecture.

The main inconvenient of the approach is that every router in a domain has to implement the trajectory-sampling scheme in order to limit the number of collisions among labels. However even in the case where such a scheme would be implemented on every router in a network, it is not clear how a trajectory would be reconstructed when adjacent routers receive the same packet within a short period of time. This technique would also have to be combined with other filtering techniques such as regular packet filtering in order to make increase the probability of small flows to be captured.

4.4.4.4 Mirkovic & al. (C1, C3, C4, C5, C7, C8)

[Mir02] suggest classifying traffic within edge routers by looking at potential attacks by comparing traffic patterns with normal or expected behaviors. This architecture called D-WARD is made of two components:

- The router that feeds the detection architecture with flow statistics and implement filtering or rate limiting rules as directed by the measurement architecture when attacks are detected.
- The D-WARD components that collect flow statistics, perform flow assessment, and configure the router with adequate filtering rules.

The D-WARD component assessment method is based on three main rules depending on the type of traffic:

- TCP flows are treated as suspect when ratio of packet acknowledgments per packet request goes under a pre-defined threshold.
- ICMP flows are treated as suspect when the ratio of responses per request for the “timestamp”, “information request” and “echo” messages goes under a pre-defined threshold or when the rate of other ICMP packets goes over a pre-defined threshold.
- UDP flows are considered as suspect when one of the following parameters is exceeded:
    - Maximum number of “connection” per destination.
    - Minimum number of packets per “connection”.
    - Maximum sending rate per “connection”.

This proposal bears several problems. The first one is that the operations performed to assess flows conformity are simple by themselves but may be too complicated when they are

implemented in an edge router since they could potentially apply to millions of flows. This problem (ability to keep the state and perform measurement operations of a large number of flows) does not currently have a nice answer even though a lot of work is currently being performed on this topic ([Ken01], [Gil01], [Est02]). In order to solve this problem the authors suggest only keeping track of most active flows. However this technique suggests that flows can be identified and measured *before* being stored in the table. Consequently it is difficult to understand how their proposal solves the problem at all. Another problem is that the proposal requires a ubiquitous deployment to become effective. Such a deployment is currently unlikely since network access provider have little incentive in such an implementation. One can also note that attackers may be able to drive the D-WARD component to perform DoS attacks on existing flows since spoofing packets on an existing connection may transform a "good" connection into a suspect one thus resulting in the connection being filtered. Finally this approach only considers traffic violation at a single point of the network without consideration to what could happen in the rest of the network. As a result this approach may prove highly ineffective against highly distributed denial of service attacks.

4.4.4.5 Gil & al. (C1, C3, C4, C5, C7, C8)

MULTOPS [Gil01] is an approach for classifying packets to detect denial of service attacks. It collects statistics about incoming and outgoing packets for a victim. In order to do so MULTOPS maintains a classification structure based on the destination address. In order to limit the size of the classification structure, the accounting operations are usually performed by aggregates. However to increase the accuracy of the victim identity the structure is dynamically modified to either focus on a specific prefix or address or to widen the scope of the aggregate monitored. The following figure provides an example of such a structure when a special attention is paid to IP address w.x.y.z.

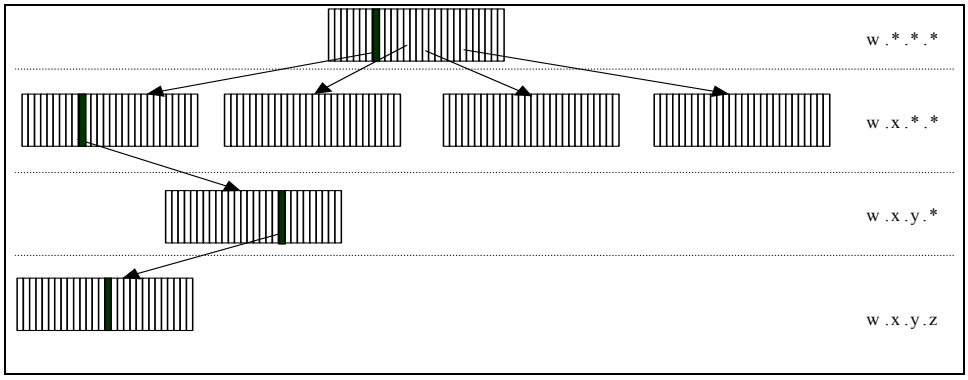

Figure 4-19: MULTOPS Classification Structure

The focusing operation is performed on a specific prefix w.z when the ratio: Number of incoming packets, number of outgoing packets for prefix w.z crosses a given threshold *t*. The heuristic is based on the assumption that the ratio incoming packets, outgoing packets in usually fixed in normal communication conditions. On the other hand the scope of the packet classification is widened from the prefix w.z to w when the ratio goes below *t*.

In addition to performing attack detection, MULTOPS also performs rate limitation when an attack is detected.

The proposal shares most advantages and drawbacks of the D-WARD approach. However the approach may also be much easier to implement since it does not rely on the ability to keep a state per flow but rely instead on packet statistics.

4.4.4.6 Cabrera & al. (C2, C4, C5, C6)

[Cab01] suggest analyzing packet traffic using Management Information Bases (MIBs). The authors suggest that attacks can be detected preemptively by finding causality relationships between MIB variables. In order to detect these causalities the authors separate attacks in several time phases (T1-slaves installation, T2-slaves attack command, T3-slaves attack and T4-attack received by the victim) and simulate specific attacks against a network node using the tools TFN2K and Trin00. Trin00 is a distributed tool used to launch coordinated UDP flood denial of service attacks from many sources. TFN2K, much like Trin00, is a distributed tool used to launch coordinated denial of service attacks from many sources against one or more targets. In additional to being able to generate UDP flood attacks, a TFN2K network can also generate TCP SYN flood, ICMP echo request flood, and ICMP directed broadcast denial of service attacks. Moreover, TFN2K has the capability to generate packets with spoofed source IP addresses.

During various attack phases, [Cab01] monitor MIB variables variations in various points of the network and try to identify causalities between these variations. Once identified, causalities can be used in the real world to detect attacks preemptively by performing variables monitoring from a network management station (NMS). For instance if we suppose that the command sent by an attacker to a set of slaves causes a specific variation in the value of one or several MIB variables it becomes possible by monitoring such variables to predict the attack.

The authors identify 6 relevant MIB variables (icmpInEchoReps, tcpInErrs, tcpInSegs, udpInErrors, udpInDatagrams, ipOutRequests) in the case of TFN2K and 3 variables (udpInDatagrams, udpOutDatagrams, ipOutRequests) for Trin00. These variables are considered as relevant either for phases T2 or T3.

Several limitations may be noted:

Causalities are not only specific to a specific attack but also to a specific form of the attack. By changing the communication protocol the attacker may bypass the attack monitoring system.

The architecture supposes that MIB variables are available not only at the victim but also at slaves. This is likely to be false in most of the cases.

It is not clear how the knowledge of an attack can be used in practice. One approach would be to stop the attack as close as possible to the slaves or attacker. However in order to do so the information has first to be retrieved by the NMS which would have to configure some network elements accordingly. However the time spent to do the whole operation in this case may be such that the configuration of network elements only occurs once the attack is over.

The way MIB variable may be monitored is not clear. On one hand polling (get) operations may be too slow to capture variations sufficiently fast. On the other hand, trap operations may be used by slaves or attackers to generate DoS on the NMS itself.

Finally false positive alerts generated by the system are quite high (up to 8%) even though the tests were performed in a local area network were the only traffic generated was the one generated by attackers and slave. It is not clear how such architecture would perform in a wide area network with real world traffic.

4.4.4.7 Snoeren & al. (C1, C2, C3, C5, C6)

In order to keep pace with the size of data a complete packet logging system can generate, [Sno01] suggests to use several methods to reduce the size necessary to know if a packet has passed through a network node or not:

- Only keeping the invariant part of IP packets. Logged information about IP packets should not include fields that can be modified during the transit of the packet; as a result TOS, TTL, checksum and optional fields should not be logged.
- Using a hash function. Using a hash function on the invariant part can reduce the size stored. However collisions (two different packets generating the same hash code) can occur. As a result it may be interesting to store several hash codes of the same packet generated using k independent hash functions. As demonstrated in [Son01] this scheme can successfully reduce the probability of collision by multiplying the size to be stored by k.
- Using bloom filters. The k independent hash functions Fk() can be used to build a bloom filter B[] where given a packet P, B[Fi(P)] is set to 1 for each i from 1 to k. If the size of Fi(P) is n then the size needed for B is 2n.

If we suppose that a router R keeps a bloom filter it is therefore possible to know if the packet P has been received by R by looking if $B[F_i(P)]=1$ for each i in the interval [1,k]. In order to avoid collisions in the bloom filter one can store a snapshot and reset the bloom filter every t seconds. It is possible to control the collision rate by modifying the value of t.

The packet logging architecture is called SPIE. The SPIE architecture is basically made of two main components. The DGA (Data Generation Agent) is expected to generate bloom filters for each packet passing through the router. Depending on the router architecture, the DGA may be implemented as a hardware part of the line-cards or as a software agent. Bloom filters are periodically recovered by a component called SCAR (SPIE collection and reduction agent) for longer-term storage. SCARs usually store bloom filters for a set of routers. The last component called STM (SPIE traceback manager) is expected to receive trace back requests (including a packet to be traced) and to question SCARs in order to understand which SCAR would have a bloom filter matching the packet. Depending on the answers provided by SCARs components, the STM is able to build a graph describing the path taken by the packet and use a description of the network topology to find neighboring SCARs that would potentially hold a matching bloom filter.

Since packets can undergo transformation (fragmentation, tunneling, address translation) in network nodes, [Sno01] suggests to store for each packet being transformed a set of information allowing the original and new packets to be matched in case of a request.

[San01] presents an FPGA hardware implementation of the SPIE architecture where hash functions are made using salted CRCs. For a 50 Mpkts/s line-card, the authors suggest to use a two level memory scheme where a first 16 Mb SRAM memory would be used to store a bloom filter updated every 5ms. This bloom filter would be transported every 5ms to a pool of 375 256 Mb DRAMs. The circular buffer organization of the DRAM memory area would allow 30 seconds for packet matching requests to reach the SCAR.

Although this packet logging approach appears very attractive since it allows any packet to be traced, it is unlikely that existing routers will be able to implement such a scheme because of the hardware cost associated to the implementation.

4.4.5 Other Approaches

4.4.5.1 Burch et al. (C1, C2, C3, C4, C5, C6, C7, C8)

[Bur00] suggests recovering the path between an attacker and a victim by generating small DoS attacks on network elements between both parties. By detecting variations in the attack rate in response to DoS attacks it is possible to find which network elements are traversed by the attack. In order to find possible paths leading to the victim the authors and to reduce the number of possible device, the authors suggest using a map of the network. This map should be constructed by using traceroute from the victim. The authors also describe how DoS attacks can be conducted against network elements without requiring important resources.

Although the proposal is interesting, it suffers from several limitations: It is unlikely to work against a largely distributed attack since variations in this case would be small. It is also possible for attackers to hide their attacks by modulating the attack flow. Another problem is the way to retrieve the map of the network since traceroute will only provide the path between the victim and other hosts on the Internet. This path can be different from the path between the attacker and the victim because of asymmetric routing. Finally the main problem with this approach is that it generates DoS itself which makes legitimate path recovery actions difficult to distinguish from illegitimate ones.

4.4.5.2 Overlay Networks (C1, C2, C3, C4, C5, C6, C7, C8)

[Ker02] suggest that DoS may be prevented using a secure overlay network architecture. This architecture is made of five components (Source, Destination, Filter, SOAP, Beacon and Servlet). The source and destination have a classical role.

- The filter is expected to control the access to the destination so that only a specific number of nodes called servlets can access the destination. The filter can be implemented using a router with access control lists.
- The servlets for each destination served will compute a subset of the nodes. These nodes will be used as beacons. In order to prevent attackers from spoofing servlet and attack the destination directly, servlet identities have to remain secret.
- Beacon nodes advertise destinations in an overlay network architecture. This overlay network architecture includes beacons as well as SOAPs (Secure Overlay Access Points). The property of the overlay network is to allow traffic sharing and node failure recovery to improve the reliability of communications between network participants.
- SOAPs (Secure Overlay Access Points) are responsible for external node acceptance in the network. They perform external nodes authentication and only forward traffic from authenticated nodes to relevant beacons inside the overlay network.

The main drawback of this approach is that it supposes that nodes authenticate themselves to the overlay network. More simple schemes could probably be used to solve DoS attacks when the identity of users can be assessed. Another problem is that it supposes that authenticated users do not generate DoS attacks using the overlay network itself. Finally in the case where the topology of the overlay network remains fixed, it would be possible for an attacker using a strategy similar to the one described in [Bur00] to find the identity of the secret servlet nodes by measuring the modification of its traffic when attacking nodes in the regular network.

4.5 Additional Issues

Several additional issues that have not yet been taken into account by DoS prevention techniques may be mentioned:

4.5.1 Mobile IPv6

IPv6 nodes must be able to process Routing Header and Home Address options. [Sav01] shows that these two options can be used to generate reflector attacks, that cannot be detected by most existing DoS detection schemes. The main reason is that these options allow a node in the network to modify the IP packet header (source and destination addresses) after the packet has passed through a firewall. These options cannot be filtered because the information provided in the optional fields is, in the general case, independent from the domain controlling the packet. As a result a border controller is usually unable to decide, by only looking at the content of these fields, if a packet should be forwarded or not. A proposal such as [Bar01] which is usually able to detect reflector based attacks ([Pax01]) does not work in this case since the attacker is able to spoof the source and the destination address by combining both options. The only efficient DoS detection scheme against these attacks would be [Sno01] since each packet whether it is modified in the network or not is logged. The author also suggests two ways to handle the problem:

- Restrict the treatment performed by hosts.
- Link the generation of filtering rules to registration operations.

4.5.2 Multicast

Multicast traffic can render the DoS situation even worse since resources in term of memory, processing power or bandwidth used to limit the ability of attackers to generate a large number of packets to a large number of destinations. Multicast sources do not suffer from this limitation because the processing power and bandwidth required generating large-scale attacks could be found on network routers outside of the computer of the attacker. In order to avoid this problem it would be interesting to stop the propagation of malicious packets in the network as early as possible. A possibility to solve this problem would be to use hop-by-hop authentication and access control such as in [Xu02]. However the cost of hop-by-hop, per packet authentication can be prohibitive. It is also not very clear how existing DoS detection proposals, mainly designed to solve DoS problems for unicast flows would handle multicast traffic.

Multicast is an efficient service for delivering data to a group of receivers. However security considerations must be addressed before the multicast communication service be deployed and used largely by the Internet service operators. More precisely, two complementary levels of security must be considered: The application-level security needs, which are the concern of the end users and content providers, and the multicast routing infrastructure security, which only considers routing aspects which is not necessarily the concern of the end-users. In the following we focus on the second aspect, the routing infrastructure security. Attacks on the Internet infrastructure can lead to enormous destruction and cause a Denial of Service (DoS); moreover, end-to-end security can be foiled by attacks that exploit vulnerabilities in the network infrastructures.

4.5.2.1 Introduction to Secure Multicast Routing Infrastructure

Many existing protocols focus on providing multicast security services for data packets at the application layer, While these protocols are very effective for theirs purpose, they use existing

insecure multicast routing for data transmission; which permits many possible attacks against the protocol. A number of attacks are possible when authentication, authorization and integrity checking of control messages are not used and any router in the multicast tree can send control messages affecting the entire multicast routing tree [clay99]. Actually there is a compelling need to develop architectures, algorithms, and protocols to realize a dependable multicast infrastructure.

The attacks to multicast routing infrastructure can be classified into **edge attacks** and **core attacks**. The edge attacks exploit the open IP-multicast service model, where, no mechanism restricts the hosts from creating a multicast group, receiving data from a group, or sending data to a group, while the core attacks exploit the vulnerabilities of multicast routers and multicast routing protocols used to construct the multicast delivery tree, actually the multicast routing protocols have much vulnerability when facing a strategically placed intruder. This intruder has the capability to fabricate, reply, monitor, or delete any of this routing traffic. In the routing environment the following general classes of intruders are identified [smith97]:

- Masquerading routers: A masquerading router is a node that forges an authorized router's identity. This can be accomplished using IP spoofing or source routing attack.
- Subverted routers: A subverted router is one that is caused to violate the routing protocols or to inappropriately claim authority for network resources. This typically occurs due to bugs in the routing code, mistake the configuration information.
- Unauthorized routers: An unauthorized router is a node that is not authorized to participate in the construction of the routing tree.
- Subverted link: A subverted link is a channel controlled by access to the physical medium or via compromise the protocols underlying the routing protocol in a manner that allows control of the channel (e.g. the TCP session hijacking).

Tow types of attacks are considered: Insider and outsider attacks. Outsider attacks involve an intruder masquerading as a router who generates, distributes, delayed and eavesdrop routing information. While, Insider attacks are mounted by a subverted or compromised router. In general, the edge or the core attacks might cause a **Denial of Service (DoS)**, which is intended to compromise the availability of the multicast service and deprives legitimate users from using it.

There are two approaches to secure the routing infrastructure. The first approach is a **protocol driven**; in this solution the security requirements are build into the routing protocols. These requirements are:

- Authentication: When a router sends out a message the identity of the originator should be able to be validated.
- Integrity: The received message is consistent with the data being received.
- Non-repudiation: Provide irrefutable evidence that a certain event took place.
- Confidentiality: The information is kept secret from all except those can see it.
- Authorization: Only authorized entities can use or alter the routing information.

In general, the protocol driven approach is achieved by using the IPsec protocol.

The other approach is to secure the routing infrastructure using **intrusion detection systems (IDSs)**.

IDSs have been widely applied for the protection of computing systems and network environments by detecting anomalies or misuses patterns. A protocol specific IDS appears to be desired extension of protocol driven techniques, moreover, IDS is viewed as a second line of defense.

Actually there are significant works in secure unicast routing protocols that can be leveraged to secure multicast routing protocols. In the following we present some works that done to secure the unicast routing protocols, firstly we present the IDSs approach then we protocol driven approaches.

- Intrusion Detection Systems.

Intrusion detection has been defined as "the process of identifying and responding to malicious activity targeted at computing and network resources"[Heady91].

All the IDSs have the same basic principles. First of all there is the monitoring part that surveys the system. This can be done in a number of ways: By measuring network traffic, using probes and sniffers or by audit trails. The collected information must thereafter be analysed in the IDS and reported. As a result an automate response is made to prevent propagation of an attack or to restore the operational status of the system.

Wang [Wang2000] presented the design, implementation, and experimentation of the JiNao intrusion detection system, which focuses on the protection of the network routing infrastructure.

JiNao is used to protect Open Shortest Path First (OSPF) routing protocol. The system features attack prevention and intrusion detection with tightly integrated network management components. The prevention module functions like a firewall, which consists of, a small set of rules. Both misuse **(protocol analysis)** and anomaly **(statistical based)** approaches are implemented as detection mechanisms in order to handle both known and unknown attacks. Four OSPF attacks (i.e., MaxSeq, MaxAge and Seq++ attacks) have been developed for evaluating JiNao's detecting capability. Furthermore, an SNMP based network management interface has been designed and implemented such that the JiNao IDS can be easily integrated with existing network management systems.

A different approach uses the principle of **conservation of flow** in a network to detect and react to routers that drop or misroute packets [Cheung99].

In this approach routers cooperatively diagnose each other to detect, locate and respond to misbehaving routers.

IDSs have a fundamental assumption that we have to live with existing systems and network infrastructures. Thus changes to them should be kept at a minimum when we improve their security, nerveless most of the existing intrusion detection works are ad-hoc in nature [Cheung99], also these systems may be of limited practicality, relying on very strong assumption that are almost impossible to satisfy in practical networks.

- Use The Internet Security Architecture (IPsec)

Kent et al. [kent00] discuss the vulnerabilities and security requirements associated with the Border Gateway Protocol (BGP). BGP is highly vulnerable to a variety of malicious attacks, due to the lack of a secure means of verifying the authenticity and legitimacy of BGP control traffic. In [kent00] the authors propose a secure, scalable architecture, S-BGP, in order to address most of the security problems associated with BGP. S-BGP involves two Public Key Infrastructures (PKIs) and the use of IPsec.

IPsec can provide authentication, integration, integrity, replay detection, and encryption to protect routing protocol traffic, nerveless IPsec can deter external attacks but cannot guarantee correct operation of the routing protocol under an internal attacks. Actually IPSEC does not protect against: Traffic analysis, Non-repudiation and Denial-of-service [FreeS/WAN].

4.5.2.2 Secure Multicast Routing Infrastructure: State of the Art

The multicast infrastructure can be divided into two parts, the core and the edge. The core contains all the multicast routers and multicast routing protocols used to construct the multicast distribution trees. The edge contains multicast edge routers and the host; IGMP is used to communicate the group membership subscriptions [draft-irtf-gsec-smrac-00.txt].

4.5.2.2.1 Secure IGMP

The current multicast model allows any host to join a multicast group by contacting their designated router using the Internet Group Management Protocol (IGMP) [Deering89]. Once the join request succeeds, the multicast distribution tree is effectively expanded towards the router and the host starts to receive the traffic. This lack of security mechanism in this open model poses many problems such as **eavesdropping**, **theft of service** or **denial-of-service attacks**. While security and confidentiality of data are still significant concerns, **receiver access control** and **resistance to denial-of-service (DoS)** attacks have become at least as significant security goals [AuraPhd00].

The term **secure IGMP** has been used to refer the protocol that would provide the possibility of controlling the ability of hosts to join the multicast group.

Traditional multicast receiver access control architecture is composted of the following functions [Judge03]:

Group policy specification: These involve a host requesting to specify a group policy, authentication the host and verifying that the host is the group owner. The group policy is access control policy that specify among other things which hosts have access rights to become members. The group owner is the entity that has been assigned ownership of the multicast group and is allowed to specify the group policy.

Access request functions: These involve a host notifying the system that it wishes to become a member of a certain group.

Access control functions: These involve receiving a host's request, authenticating he host and performing authorization. Authorization requires checking the group policy to determine if that host has the aces rights to become member of the requested group.

Multicast receiver access control architecture also interacts with the routing system and any group key management system that may be in place. The design goals of this architecture: Security and scalability. The scalability objective is achieved by reducing the computational load on network routers and message overhead.

The authors of [ballardie95] first stated the need for multicast group access control. They propose a version of the IGMP protocol that can enforce subnet-level group access control.

Judge and Ammar [ammar99] propose Gothic, architecture for providing group access control. Gothic based on new **authorization architecture**. This is complemented by a proposal for a group policy management system that allows the group owner to be authenticated before being allowed to specify the group access rights. They show how Gothic operates in a number of environments including application-layer multicast, source-specific multicast, application-layer anycast and global IP-anycast. The design goal was to maintain security while proving scalable system that involve **low computation overhead at the routers,** low message overhead and low infrastructure requirements.

In [draft-irtf-gsec-smrac-00.txt] the authors propose a method to achieve the goal of authenticating and authorizing a host's IGMP join requests. The method uses the approach: **Proof of Membership**. A host needs to provide its neighboring multicast routers with a proof of membership before its IGMP join requests are processed. Here the proof-of-membership is an access token that is digitally signed by a trusted authority such as **Group Controller Key Server (GCKS)** using the private key of a public-key pair designated for receiver access control. The solution requires that only legitimate multicast routers posses the public key of the key pair.

Another approach [Murcia300] suggest to extend IGMPv3 to carry authentication information so that the local attached router in a network could decide whether to attend the IGMPv3 report or non. The key idea is to include the result of hashing the IGMP packet with the **Auxiliary Data Field to carry** the user's private key in the IGMPv3 packets.

A secure IGMP can solve the problem of access control; nerveless the traditional solutions are inadequate for preventing a misbehaving receiver to cause certain IGMP DoS attacks. IGMP DoS attacks present a serious threat to the multicast operator infrastructure. Gorinsky et al. [Gorinsky02] present a lightweight multicast group access control based on the congestion status of receivers in order to preventing a misbehaving receiver to elicit unfair bandwidth allocation.

4.5.2.2.2 Secure Multicast Routing protocols

4.5.2.2.2.1 Secure CBT

The first attempt to secure multicast routing protocol came with the CBT protocol [ballardie96]. This work pointed out the need to combine multicast key distribution with group access control.

The author describes a Scalable Multicast Key Distribution scheme (SMKD) where CBT protocol is used to secure joining of a CBT group tree, and to provide a scalable solution to the multicast key distribution problem. SMKD uses Group Key Distribution Centres (GKDCs) instead of using a single trusted entity or Key Distribution Centre (KDC). GKDCs are dynamically constructed during group-member joining process.

SMKD utilizes the hard-state approach of CBT in which routers on the delivery tree know the identities of their tree-neighbors. Once a CBT group is initiated the core of the tree operates as the group controller and generates the group session keys and key distribution keys. As routers join the delivery tree they are delegated the ability to authenticate joining members and provide them with the group key. This approach is scalable. However, it is tied to a specific routing protocol, and does not provide a separation between the routing and the security mechanism. (In particular, it puts high trust in the routers, since each router in the delivery tree obtains the same keys as the group controller.) [Draft-canetti-secure-multicast-taxonomy-01.txt].

A recent study [Matsuura] analyzes the SMKD and proposes several improvements such as: Avoid the overuse of encryption and signature and introduce a hybrid trust model by a simple mechanism For controlling the GKDC distribution.

4.5.2.2.2.2 Keyed Hierarchical Multicast Routing (KHIP)

Gong and Shacham [Gong95] show that the consideration of security brings a new set of tradeoffs in routing private multicast traffic. They describe these trade-offs between security, trust, and knowledge of group membership and of topology, types of multicast tree, bandwidth consumption, and latency. This work introduces four methods to improve efficiency: Message pruning, message reprocessing, hot-start authentication, and continuous authentication. These

methods aim to reduce the size and number of control messages needed to authenticate group members and to distribute encryption keys to the group.

Their ideas appeared in many later work [Iolus][ballardie96]. Moreover their works motivate the design of Keyed Hierarchical Multicast Routing (KHIP), a secure, hierarchical multicast routing protocol [clay99].

In his thesis clay shields shows that the shared-tree multicast routing protocols are subject to attacks against the multicast routing infrastructure, these attacks can isolate receivers or domains or introduce loops into the structure of the multicast routing tree. KHIP changes the multicast routing model so that only trusted members are able to join the multicast tree. This protects the multicast routing against attacks that could form branches to unauthorized receivers, prevents replay attacks and limits the effects of flooding attacks. Untrusted routers that are present on the path between trusted routers cannot change the routing and cannot mount denial-of-service attack stronger than simply dropping control messages. KHIP provides a mechanism for distributing data encryption keys while adding little overhead to the protocol [clay99]. Nerveless, KHIP uses a bi-directional, core-based multicast routing tree, and lacks facilities for excluding specific source from the tree. In fact all bi-directional shared tree protocols break down into the same state as per-source trees when individual source must authenticated for each receiver [Doit00].

4.5.2.2.2.3 Protocol Independent Multicast Sparse Mode (PIM-SM)

PIM-SM is a unidirectional, receiver-initiated, soft state, unicast routing independence multicast routing protocol specifically designed to cope with sparse groups. That is, groups that are numerous, that spread over large networks like the Internet, and that involve a small proportion of the domain's switching and transmission resources in the ongoing process of building forwarding trees [draft-ietf-pim-sm-v2-new-06.txt].

PI-SM supports both shared and source distribution trees. For shared trees, PIM-SM uses a central router, called the Rendezvous Point (RP), as the root of the shared tree. All source hosts send their multicast traffic to the RP, which in turn forwards the packets through a common tree to all the members of the group. Source trees directly connect sources to receivers. There is a separate tree for every source. Source trees are considered shortest-path trees from the perspective of the unicast routing tables. PIM-SM can use either type of tree or both simultaneously.

While PIM-SM emerge as the multicast routing protocol standard in the networking industry, the security issues represents a crucial factor for an effective deployment.

The latest specification of PIM [draft-ietf-pim-sm-v2-new-06.txt] contains an overview of possible attacks that are based on forged PIM control messages and describes optional and recommended authentication mechanisms. The IPsec authentication header [RFC2401] is recommended to provide data integrity protection and groupwise data origin authentication of PIM protocol messages, nerveless the multiparty aspects of PIM-SM control messages introduces special problem to apply Ipsec AH or to set up the required Security Associations (SAs) [draft-irtf-gsec-pim-sm-security-issues-01.txt] [Alchaal02].

Many works have been made to define a key management protocol in PIM, in [draft-ietf-pim-simplekmp-00.txt], the authors present a key distribution method based on the use of a combination of symmetric et asymmetric to authenticate the control message in a single PIM domain, in order to allow some control-messages from a PIM-entity in a PIM-domain D1 to

cross the domain boundary to another PIM-entity in a different PIM-domain D2, A Domain Key Distribution DKD) entity is introduced.

4.5.2.2.3 Discussion

Multicast is an efficient service but it is very vulnerable to DoS attacks. In the light of the previous overview, IDSs appear to be an attractive solution for protecting routing infrastructures from denial of service attacks. Moreover, a protocol specific IDS appear to be desired extension of protocol driven techniques.

4.6 Summary of DoS Avoidance Techniques

The table below provides a summary of the functionalities provided by each solution. Proposals are gathered by classes. Original proposals in the class are marked with a star.

Proposal/Function	Preventive	Detection	Tracking	Suppression
Ingress Filtering:				
Ingress Network Filtering*	X			
URPF	X			
Distributed Packet Filtering (Park & al.)	X			
Packet header marking:				
Doeppner & al.			X	
Savage &al.*			X	
Song & al.			X	
Dean & al.			X	
Adler			X	
Control plane based approaches:				
ICMP Traceback (Regular)*			X	
ICMP Traceback (Intention Based)			X	
ICMP Traceback (Reverse)			X	
Pushback (Floyd & al.)		X	X	X
Blackhole Routing (Regular)*				X
Centertrack (stone)			X	X
Blackhole Routing (with ICMP backscatter)			X	X
Destination Class Usage with BGP			X	
Qos Policy Based Routing (QPPB)				X
Packet logging:				
Log ACL*			X	
Netflow			X	
IP source Tracking			X	
D-Ward (Mirkovic & al.)		X		X
MULTOPS (Gil & al.)		X		X
MIB monitoring (Cabrera & al.)		X		
SPIE (Snoeren & al.)*			X	
Trajectory Sampling (Duffield & al.)			X	
Other approaches:				
Burch & al.*			X	
Overlay network (Keromytis & al.)*	X			

Figure 4-20: Proposals classification

5. DENIAL OF SERVICE MEASUREMENT PARAMETERS FOR 6QM PROBES

The goal of this section is to identify measurement parameters that may be used to perform DoS prevention operations. In order to do so we will examine existing DoS prevention schemes and extract the set of measurement parameters that should be used to implement such solutions.

We include two subsections providing relevant measurement parameters regarding DoS attacks detection and DoS attacks tracking. We analyze existing detection and tracking approaches and identify key parameters whether they can be related to QoS measurement metrics or not. In the next section we try to match parameters previously identified with standardized QoS measurement metrics.

5.1 Denial of Service Detection and Tracking Parameters

In this section we identify which parameters would be necessary to provide existing DoS prevention schemes. These parameters are divided into two sets: Parameters given in 5.1.1 can be used to identify or detect DoS attacks. Parameters given in 5.1.2 can be used to track attacks in the network. DoS prevention proposals that do not cover detection or tracking aspects are indicated as not relevant. For more information on DoS prevention approaches the interested reader may refer to [Pa02] or any original article indicated in the reference section.

5.1.1 Detection Measurement Parameters

Proposal/Function.	Parameters	Where
Pushback (Floyd & al.)	The number of datagrams dropped in the output queue for all line-cards. The number of bytes dropped in the output queue for all line-cards. The number of datagrams dropped in the input queue for all line-cards. The number of bytes dropped in the input queue for all line-cards.	All routers.
D-Ward (Mirkovic & al.)	The ratio of TCP datagrams sent/acknowledge for all destination addresses. The ratio of ICMP “timestamp” request/response for all destination addresses. The ratio of ICMP “information request” request/response for all destination addresses. The ratio of ICMP “echo” request/response for all destination addresses. The number of ICMP datagrams sent to all destination addresses. The number of UDP “connections” to all destination addresses. The number of UDP datagrams sent on all UDP “connections”. The number of UDP datagrams sent per second on all UDP “connections”.	Provider Edge (Routers).
MULTOPS (Gil & al.)	The ratio of datagrams received/transmitted to a specific	Provider Edge

Proposal/Function.	Parameters	Where
	destination or prefix. The ratio of bytes received/transmitted to a specific destination or prefix.	(Routers) or Peering points.
MIB monitoring (Cabrera & al.)	For TFN2K: The number of ICMP Echo Reply messages received. The total number of segments received in error (e.g., bad TCP checksums. The total number of segments received, including those received in error. This count includes segments received on currently established connections. The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port. The total number of UDP datagrams delivered to UDP users. The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission. For Trin00: The total number of UDP datagrams delivered to UDP users. The total number of UDP datagrams sent from this entity. The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission.	Hosts (terminal).

Figure 5-1: Detection Parameters

5.1.2 Tracking Measurement parameters

Proposal/Function.	Parameters	Where
Blackhole Routing (Regular)	The number of datagrams/s received and matching a specific datagram filter per interface (regular). The number of datagrams/s routed to a specific destination/ using a specific route (edge extension)	All routers. (regular) Edge routers and blackhole router (edge extension).
Centertrack (stone)	The number of datagrams/s received and matching a specific datagram filter per interface.	All routers.
Blackhole Routing (with ICMP backscatter)	A list including the number of ICMP unreachable datagrams/s received by the router for a given list of datagram source addresses.	Blackhole router.
Destination Class Usage with BGP	The number of packets being counted pertaining to a specified DCU entry. [DCU00] The number of bytes being counted pertaining to a specified DCU entry. [DCU00]	Peering points and Provider Edge (Routers).
Pushback (Floyd & al.)	The number of datagrams/s received and matching a specific datagram filter per interface.	All routers.
Log ACL	The number of datagrams/s received and matching a specific datagram filter per interface.	All routers.

Proposal/Function.	Parameters	Where
Netflow	The number of flows/s received and matching a specific datagram filter per interface.	All routers.
IP source Tracking	An ordered list of the n types of datagram generating the largest part of the traffic for an outgoing interface. Two types of datagrams are said to be different if one of the following conditions holds: • The network or transport layer headers (H1, H2) are different. • The datagrams originate from two different incoming interfaces.	All routers.
SPIE (Snoeren & al.)	A Boolean answer indicating if a given datagram has been routed by the router. The number of such datagrams. Transformations that may have been performed on the datagram (NAT, Tunneling).	All routers or Peering points and Provider Edge (Routers).
Trajectory Sampling (Duffield & al.)	A hash value indicating that a given datagram has been routed by the router for: • A given *capture* hash function or hash parameters. • A given *capture* hashing level. • A given *datagram coding* hash function.	Peering points and Provider Edge (Routers).
Burch & al.	The number of datagrams/s received and matching a specific datagram filter per interface.	Peering points and Provider Edge (Routers).

Figure 5-2: Tracking Parameters

5.2 QoS Measurement Parameters

In this section we examine existing IPPM QoS measurement parameters to understand which DoS detection and tracking architecture could be implemented using these parameters by trying a map between QoS measurement metrics and DoS parameters identified in the previous section. Figure 5-3 provides the QoS measurement parameters for existing tracking architectures. It should be however noted that since most parameters identified in the previous section do not relate directly to QoS measurement, QoS metrics/parameters associations should be considered with caution. As a result we often provide several mappings between QoS metrics and DoS parameters when we feel that the usage of several metrics may improve the overall detection or tracking scheme.

5.2.1 Detection Architectures

Proposal/ Function	IPPM QoS Parameters	Equivalent DoS parameters	Observation Point	Change to be observed
Pushback (Floyd & al.)	*Type-P-One-way-Packet-Loss from attacker to victim.* IPFIX packet counter. IPFIX byte counter.	The number of datagrams dropped in the output queue for all line-cards. The number of bytes dropped in the output queue for all line-cards.	Before and After each line-card (Adjacent routers or ingress, egress line-cards).	Increase Increase
	Type-P-One-way-Packet-Loss from attacker to victim. IPFIX packet counter. IPFIX byte counter.	The number of datagrams dropped in the input queue for all line-cards. The number of bytes dropped in the input queue for all line-cards.	Before and After each line-card (Adjacent routers or ingress, egress line-cards).	Increase Increase
D-Ward (Mirkovic & al.)	*Type-TCPSYN-TCPACK-Interval-Temporal-Connectivity.* TCP-SYN Instantaneous Throughput from Attacker to Victim. TCP-ACK Instantaneous Throughput from Victim to Attacker.	The ratio of TCP datagrams sent/acknowledge for all destination addresses.	Provider Edge (Routers).	Decrease Ratio Increase
	Type-ICMPTimeStampRequest-ICMPTimeStamp Response-Interval-Temporal-Connectivity. ICMP-TimeStamp-request Instantaneous Throughput from Attacker to Victim. ICMP-TimeStamp-response Instantaneous Throughput from Attacker to Victim.	The ratio of ICMP “timestamp” request/response for all destination addresses.	Provider Edge (Routers).	Decrease. Ratio Increase.

Proposal/ Function	IPPM QoS Parameters	Equivalent DoS parameters	Observation Point	Change to be observed
	Type-ICMPInformationRequestRequest-ICMPInformationRequestResponse- Interval-Temporal-Connectivity. ICMP-InformationRequest-request Instantaneous Throughput from Attacker to Victim. ICMP- InformationRequest - response Instantaneous Throughput from Attacker to Victim.	The ratio of ICMP "information request" request/response for all destination addresses.	Provider Edge (Routers).	Decrease. Ratio Increase.
	Type-ICMPEchoRequest-ICMPEchoResponse-Interval-Temporal-Connectivity. ICMP-Echo-request Instantaneous Throughput from Attacker to Victim. ICMP-Echo-response Instantaneous Throughput from Attacker to Victim.	The ratio of ICMP "echo" request/response for all destination addresses.	Provider Edge (Routers).	Decrease. Ratio Increase.
	ICMP Instantaneous Throughput. UDP Instantaneous Throughput. IPFIX flow packet/byte counters IPFIX number of UDP "flows"	The number of ICMP datagrams sent to all destination addresses. The number of UDP datagrams sent on all UDP "connections". The number of UDP datagrams sent per second on all UDP "connections". The number of UDP "connections" to all destination addresses.	Provider Edge (Routers).	. Increase.

Proposal/ Function	IPPM QoS Parameters	Equivalent DoS parameters	Observation Point	Change to be observed
MULTOPS (Gil & al.)	TCP Instantaneous Throughput from Attacker to Victim. TCP Instantaneous Throughput from Victim to Attacker.	The ratio of datagrams received/transmitted to a specific destination or prefix.	Provider Edge (Routers) or Peering points.	Decrease. Ratio Increase.
	IPFIX byte/packet counters from Attacker to Victim. IPFIX byte/packet counters from Victim to Attacker.	The ratio of bytes received/transmitted to a specific destination or prefix.	Provider Edge (Routers) or Peering points.	Decrease. Ratio Increase.
MIB monitoring (Cabrera & al.)	For TFN2K: • ICMP Echo Reply Instantaneous Throughput. • IPFIX ICMP Echo Reply packet counter.	For TFN2K: • The number of ICMP Echo Reply messages received.	Hosts (terminal).	Increase.
	For TFN2K: • None. .	For TFN2K: • The total number of segments received in error (e.g., bad TCP checksums.	Hosts (terminal).	Increase.
	For TFN2K: • TCP Ingress Instantaneous Throughput. • IPFIX TCP packet counter.	For TFN2K: • The total number of segments received, including those received in error. This count includes segments received on currently established connections.	Hosts (terminal).	Increase.
	For TFN2K: • UDP Ingress Instantaneous Throughput. • IPFIX UDP packet counter.	For TFN2K: • The total number of UDP datagrams delivered to UDP users.	Hosts (terminal).	Increase.

Proposal/ Function	IPPM QoS Parameters	Equivalent DoS parameters	Observation Point	Change to be observed
	For TFN2K: • IP egress Instantaneous Throughput. • IPFIX IP packet counter.	For TFN2K: • The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission.	Hosts (terminal).	Increase.
	For Trin00: • UDP Ingress Instantaneous Throughput. • IPFIX Ingress UDP packet counter.	For Trin00: • The total number of UDP datagrams delivered to UDP users.	Hosts (terminal).	Increase.
	For Trin00: • UDP egress Instantaneous Throughput. • IPFIX egress UDP packet counter.	For Trin00: • The total number of UDP datagrams sent from this entity.	Hosts (terminal).	Increase.
	For Trin00: • IP egress Instantaneous Throughput. • IPFIX egress IP packet counter.	For Trin00: • The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission.	Hosts (terminal).	Increase.

Figure 5-3: Detection Architectures.

5.2.2 Tracking Architectures

Proposal/Function.	IPPM QoS Parameters	DoS Parameters	Where
Blackhole Routing (Regular)	Type-P ingress Throughput. IPFIX Packet counter. IPFIX Next Hop IP Address Packet Counter.	The number of datagrams/s received and matching a specific datagram filter per interface (regular). The number of datagrams/s routed to a specific destination/ using a specific route (edge extension)	All routers. (regular) Edge routers and blackhole router (edge extension).
Centertrack (stone)	Type-P ingress Throughput. IPFIX Packet Counter.	The number of datagrams/s received and matching a specific datagram filter per interface.	All routers.

Proposal/Function.	IPPM QoS Parameters	DoS Parameters	Where
Blackhole Routing (with ICMP backscatter)	Type-ICMP-Unreachable ingress Throughput. IPFIX ICMP packet counter.	A list including the number of ICMP unreachable datagrams/s received by the router for a given list of datagram source addresses.	Blackhole router.
Destination Class Usage with BGP	None None.	The number of packets being counted pertaining to a specified DCU entry. [DCU00] The number of bytes being counted pertaining to a specified DCU entry. [DCU00]	Peering points and Provider Edge (Routers).
Pushback (Floyd & al.)	Type-P Packet Instantaneous Throughput. IPFIX packet counter.	The number of datagrams/s received and matching a specific datagram filter per interface.	All router interfaces.
Log ACL	Type-P Packet Instantaneous Throughput. IPFIX packet counter.	The number of datagrams/s received and matching a specific datagram filter per interface.	All router interfaces.
Netflow	Type-P Packet Instantaneous Throughput. IPFIX packet counter.	The number of flows/s received and matching a specific datagram filter per interface.	All router interfaces.
IP source Tracking	Type-P Packet Instantaneous Throughput. IPFIX packet counter. IPFIX byte counter.	An ordered list of the n types of datagram generating the largest part of the traffic for an outgoing interface. Two types of datagrams are said to be different if one of the following conditions holds: • The network or transport layer headers (H1, H2) are different. • The datagrams originate from two different incoming interfaces.	All routers.
SPIE (Snoeren & al.)	None. IPFIX packet counter. IPFIX byte counter. None	A Boolean answer indicating if a given datagram has been routed by the router. The number of such datagrams. Transformations that may have been performed on the datagram (NAT, Tunneling).	
Trajectory Sampling (Duffield & al.)	(Should be possible with PSAMP but no information model is currently defined)	A hash value indicating that a given datagram has been routed by the router for: • A given *capture* hash function or hash parameters. • A given *capture* hashing level. • A given *datagram coding* hash	

Proposal/Function.	IPPM QoS Parameters	DoS Parameters	Where
		function.	
Burch & al.	Type-P Packet Instantaneous Throughput. IPFIX packet counter. IPFIX byte counter.	The number of datagrams/s received and matching a specific datagram filter per interface.	

Figure 5-4: Tracking Parameters

6. DOS RESILIENT ARCHITECTURE REQUIREMENTS

In order to define the requirements for a DoS resilient QoS measurement architecture we first need to analyze the potential risks. In order to do so we analyze possible relations between the various components that may be used to develop a measurement architecture and identify the risks associated with each one. We then express requirements in order to reduce the risks previously identified. We focus on two components; the aggregation point and the metric computation point.

6.1 Aggregation Point

The aggregation point supports several functions:

- Packet Sampling.
- Packet Filtering.
- Packet Hashing.

Functions may be combined to provide PSAMP, IPFIX or IPPM related functionalities. Additionally these functions may be implemented on separate components.

The main risks associated with the functions executed in an aggregation point are associated with physical components saturation. The saturation of a physical component can either create instability in the whole system resulting in system freezing or reboot and or loss of traffic thus leading to faulty or biased QoS metrics measurement results. The following attacks can be performed in order to create components saturation:

- Most filtering processes do not provide performances independent from the traffic to be filtered. This means that the filtering of some traffic will necessitate more time or memory space than the one needed for the filtering of a regular traffic. As a result an attacker may generate a specific type of traffic to either saturate the filtering process or limit the resources that may be available to other process.
- The various operations that can be performed for the aggregation function can be executed by different physical components. Some components may have a lower capacity than others. As a result an attacker may try to saturate a lower capacity component with the traffic originating from a higher capacity component.
- In addition to the biasing measures, the attacker may additionally hide his attack if packet losses or system instability are not monitored sufficiently well.

Finally in the case where the aggregation point only performs predictive operations (filtering, hashing, some forms of sampling) an attacker may take advantage of the way operations are performed to influence the result of measurement operations. For example if a periodic sampling function (either using time or number of packets) is used it may be possible for an attacker to generate traffic that bypass measurement operations.

Type of requirement	RID	Requirement	Level of requirement
Sampling	D1.1	The sampling process is non-periodic	Should

	D1.2	The sampling process uses a memory – cpu efficient algorithm.	Must
	D1.3	The classification process uses a widely tested algorithm.	Should
	D1.4	If several sampling schemes are available on a device the most efficient schemes can be used to reduce the number of sampling operations for slower sampling processes.	Must
	D1.5	The sampling process has predictive worst cases (memory & CPU)	Must
	D1.6	The sampling process has CPU performance independent of the number of sampling configuration	Should
	D1.7	The sampling process provides fail safe operations (i..e. The failure of a sampling process does not generate DoS on other components)	Must
	D1.8	The sampling process has CPU performance independent of the composition or content of the traffic.	Should
	D1.9	The sampling process indicates how many items can not be sampled because of insufficient CPU resources	Must
	D1.10	If several sampling schemes are available on several devices the device providing the most efficient schemes has to be used to reduce the number of sampling operations for devices with slower sampling processes.	Must
	D1.11	Transmission of sampled information between sampling components uses congestion aware protocols	May
	D1.12	Transmission of sampled information between sampling components uses loss aware protocols	Must
	D1.13	Transmission of sampled information between sampling components uses a protocol as lightweight as possible.	Should
	D1.14	Number of concurrent sampling configuration operations can be limited	Should
	D1.15	Time spent by individual sampling configuration operation can be limited	Should
	D1.16	Resources (cpu/memory) needed for the execution of a specific sampling configuration can be evaluated in advance.	May
	D1.17	Resources (cpu/memory) needed for the execution of a specific sampling configuration can be evaluated afterward.	Should
Filtering	D2.1	The classification process uses a memory – cpu efficient algorithm. (e.g. on a Linux box, nf-hipac, a packet filter implemented on top of the netfilter framework, is preferred over the regular packet classification algorithm).	Must
	D2.2	The classification process uses a widely tested algorithm.	Should
	D2.3	If several classification schemes are available on a device the most efficient schemes can be used to reduce the number of classification operations for slower classification processes.	Must
	D2.4	The classification process has predictive worst cases (memory & CPU)	Must
	D2.5	The classification process has CPU performance independent of the number of classification rules	Should
	D2.6	The classification process provides fail safe operations (i..e. The failure of a classification process does not generate DoS on other components)	Must

	D2.7	The classification process has CPU performance independent of the composition or content of the traffic.	Should
	D2.8	The classification process indicates how many packets can not be classified because of insufficient CPU resources	Must
	D2.9	The classification process indicates how many classification rules are not implemented because of insufficient memory performance	Must
	D2.10	The classification process indicates which rules have not been implemented because of insufficient memory performance	May
	D2.11	The classification process provides basic classification of packets that can not classified because of insufficient CPU resources	Should
	D2.12	If several classification schemes are available on several devices the device providing the most efficient schemes has to be used to reduce the number of classification operations for devices with slower classification processes.	Must
	D2.13	Transmission of classified packets between classification components uses congestion aware protocols	May
	D2.14	Transmission of classified packets between classification components uses loss aware protocols	Must
	D2.15	Transmission of classified packets between classification components uses a protocol as lightweight as possible.	Should
	D2.16	The statefull information kept for each flow has a know size	Should
	D2.17	The number of states kept at a single moment can be limited	Must
	D2.18	Lack of space to store new states are indicated	Must
	D2.19	Lack of state storage space does not prevent other classification operations	Must
	D2.20	Number of states that can not be stored is indicated	Must
	D2.21	Number of concurrent classification configuration operations can be limited	Should
	D2.22	Time spent by individual classification configuration operation can be limited	Should
	D2.23	Resources (cpu/memory) needed for the execution of a specific classification configuration can be evaluated in advance.	May
	D2.24	Resources (cpu/memory) needed for the execution of a specific classification configuration can be evaluated afterward.	Should
Hashing	D3.1	The hashing process can be seeded with a random value	Should
	D3.2	The hashing process uses a memory – cpu efficient algorithm	Should
	D3.3	The hashing process uses a cryptographically strong algorithm (i.e. MD5 or SHA-1 would be preferred over CRC-32).	Must
	D3.4	The hashing process uses a widely tested algorithm.	Should
	D3.5	If several hashing schemes are available on a device the most efficient schemes can be used to reduce the number of hashing operations for slower hashing processes.	Must
	D3.6	The hashing process has predictive worst cases (memory & CPU)	Must
	D3.7	The hashing process has CPU performance independent of the number of hashing configuration	Should

	D3.8	The hashing process provides fail safe operations (i..e. The failure of a hashing process does not generate DoS on other components)	Must
	D3.9	The hashing process has CPU performance independent of the composition or content of the traffic.	Should
	D3.10	The hashing process indicates how many items can not be hashed because of insufficient CPU resources	Must
	D3.11	If several hashing schemes are available on several devices the device providing the most efficient schemes has to be used to reduce the number of hashing operations for devices with slower hashing processes.	Must
	D3.12	Transmission of hashed information between hashing components uses congestion aware protocols	May
	D3.13	Transmission of hashed information between hashing components uses loss aware protocols	Must
	D3.14	Transmission of hashed information between hashing components uses a protocol as lightweight as possible.	Should
	D3.15	Number of concurrent hashing configuration operations can be limited	Should
	D3.16	Time spent by individual hashing configuration operation can be limited	Should
	D3.17	Resources (cpu/memory) needed for the execution of a specific hashing configuration can be evaluated in advance.	May
	D3.18	Resources (cpu/memory) needed for the execution of a specific hashing configuration can be evaluated afterward.	Should
Functions - Coordination	D4.1	If several functions are available on a device the most efficient schemes can be used to reduce the number of operations for slower classification processes (e.g. if sampling is faster than filtering then sampling can be used before filtering)	Must
	D4.2	If several basic functions are available on several devices the device providing the most efficient schemes has to be used to reduce the number of operations for devices with slower processes	Must
	D4.3	Transmission of information between components uses congestion aware protocols/methods	Must
	D4.4	Transmission of information between components uses loss aware protocols (i.e. loss should be either indicated or solved)	Must
	D4.5	Transmission of information between components uses a protocol as lightweight as possible and minimize delay.	Should
	D4.6	Saturation of a link between component and the resulting loss of information should be explainable. (i.e. The extraction of the piece of information generating the saturation can performed)	Should
	D4.7	Links connecting aggregation components are only used for measurement operation (i.e. configuration and information retrieval)	Must
	D4.8	Configuration operations of components can be limited to a share of the resources (memory, cpu)	Should
	D4.9	Number of concurrent configuration operations can be limited	Should
	D4.10	Time spent by individual configuration operation can be limited	Should

	D4.11	Resources (cpu/memory) needed for the execution of a specific QoS measurement configuration can be evaluated in advance.	May
	D4.12	Resources (cpu/memory) needed for the execution of a specific QoS measurement configuration can be evaluated afterward.	Should
Communication MCP to AP	D5.1	Communication rate (MCP to AP) between MCP and AP can be limited	Must
	D5.2	Number of concurrent connections between MCP and AP can be limited. Connections can be refused before being authenticated.	Should
	D5.3	MCP can authenticate himself to AP	Should
	D5.4	Communication rate (MCP to AP) can be limited per connection	May
	D5.5	MCP - AP communication protocol supports fail over MCP in case of MCP failure	Must
	D5.6	Actual Architecture uses several MCP	Should
	D5.7	Communication rejection should be done in way that avoids AP communication requests synchronization (AP to MCP).	Should
	D5.8	Transmission of information between AP and MCP uses congestion aware protocols	Must
	D5.9	Transmission of information between AP and MCP uses loss aware protocols (i.e. loss should be either indicated or solved)	Must
	D5.10	Transmission of information between AP and MCP uses a protocol as lightweight as possible.	Should
	D5.11	Data Integrity and Authentication are insured for communications.	Must
General	D6.1	All aggregation physical components are only accessible to authorized personnel.	Must
	D6.2	All components are operated using resilient power system	Should
	D6.3	Failure of the aggregation component does not alter the operation of the underlying communication devices.	Must
	D6.4	The underlying network infrastructure implements DoS tracking mechanisms.	Must
	D6.5	DoS detection and Tracking mechanisms are not affected by a DoS attack on aggregation operations	Should
	D6.6	APs run minimal services in addition to aggregation services	Must

Figure 6-1: Aggregation Point Requirements

6.2 Metrics Computation Point

The metric computation point can be submitted to several attacks. The attacks depend on the point of view the attacker may take:

- From an external point of view.
  - An attacker may generate traffic in order to generate an AP reaction.
    - By combining several AP reactions the attacker may try to overflow an MCP with AP responses.
    - By generating specific several AP reactions the attacker may try to saturate an MCP by generating time or memory consuming computations.

- From an AP point of view:
  - An attacker may overflow an MCP with aggregation responses (responses may be asynchronous). In order to do so he may take one or several AP identities.
  - An attacker may modify the content of AP responses in order to increase the computation performed by the MCP.
  - An attacker may delay responses by either introducing delay or errors on AP responses in order to generate saturation on the AP or MCP.
- From an MCP point of view:
  - An attacker may simulate an aggregation point to overflow an AP with aggregation requests. In order to do so he may take one or several MCP identities.
  - An attacker may modify the content of MCP requests in order to increase the computation performed by the AP and or the number of results sent to the MCP.
  - An attacker may delay responses by either introducing delay or errors on MCP requests in order to generate saturation on the MCP or AP.
- From an UAP point of view:
  - An attacker may overflow an MCP with measurement requests. In order to do so he may take one or several UAP identities.
  - An attacker may modify the content of UAP requests order to increase the computation performed by the MCP and APs.
  - An attacker may delay responses and requests by either introducing delay or errors on UAP –MCP communications in order to generate saturation on UAPs, MCPs and APs.

Type of requirement	RID	Requirement	Level of requirement
Communication AP to MCP	M1.1	Communication rate (AP to MCP) between AP and MCP can be limited	Must
	M1.2	Number of concurrent connections between AP and MCP can be limited. Connections can be refused before being authenticated.	May
	M1.3	MCP can authenticate himself to aggregation point	Should
	M1.4	Communication rate (AP to MCP) can be limited per connection	Should
	M1.5	MCP - AP communication protocol supports fail over MCP in case of MCP failure	Must
	M1.6	Actual Architecture uses several MCP	Should
	M1.7	Communication component is dimensioned to process n response communication from AP simultaneously without introducing delays or instability in the MCP	Should
	M1.8	Communication rejection should be done in way that avoids MCP communication requests synchronization (MCP to AP).	Should
	M1.9	Transmission of information between AP and MCP uses congestion aware protocols	Must
	M1.10	Transmission of information between AP and MCP uses loss aware protocols (i.e. loss should be either indicated or solved)	Must

	M1.11	Transmission of information between AP and MCP uses a protocol as lightweight as possible.	Should
	M1.12	Data Integrity and Authentication are insured for communications.	Must
Metrics Computation	M2.1	Computation component is dimensioned to process n computations from AP simultaneously without introducing delays or instability in the MCP	Should
	M2.2	Computation operations have predictive cost in term of CPU and memory (cost of operations can be predicted)	May
	M2.3	Computation operations have predictive duration; computation operation can be cancelled or aborted after a given duration.	Should
	M2.4	Computation operations cost (memory, cpu) can be computed afterward	Should
	M2.5	Computation algorithms are efficient	Must
	M2.6	Computation algorithms have been widely tested	Must
MCP Configuration	M3.1	Configuration operations of components can be limited to a share of the resources (memory, cpu)	Should
	M3.2	Number of concurrent configuration operations can be limited	Should
	M3.3	Time spent by individual configuration operation can be limited	May
	M3.4	Resources (cpu/memory) needed for the execution of a specific QoS measurement configuration can be evaluated in advance.	Should
	M3.5	Resources (cpu/memory) needed for the execution of a specific QoS measurement configuration can be evaluated afterward.	Should
Communication UAP to MCP	M4.1	Communication rate (UAP to MCP) between UAP and MCP can be limited	Must
	M4.2	Number of concurrent connections between UAP and MCP can be limited. Connections can be refused before being authenticated.	Should
	M4.3	UAP can authenticate himself to aggregation point	Must
	M4.4	Communication rate (UAP to MCP) can be limited per connection	Should
	M4.5	Communication rejection should be done in way that avoids MCP communication requests synchronization (MCP to UAP).	Should
	M4.6	Transmission of information between UAP and MCP uses congestion aware protocols	Must
	M4.7	Transmission of information between UAP and MCP uses loss aware protocols (i.e. loss should be either indicated or solved)	Must
	M4.8	Transmission of information between UAP and MCP uses a protocol as lightweight as possible.	Should
	M4.9	Data Integrity and Authentication are insured for communications.	Must
General	M5.1	All aggregation physical components are only accessible to authorized personnel.	Must
	M5.2	All components are operated using resilient power system	Should
	M5.3	Failure of the aggregation component does not alter the operation of the underlying communication devices.	Must

	M5.4	The underlying network infrastructure implements DoS tracking mechanisms.	Must
	M5.5	DoS detection and Tracking mechanisms are not affected by a DoS attack on aggregation operations	Should
	M5.6	Load balancing can be performed between MCP and UAP (several MCP are used to serve several UAPs)	May

Figure 6-2: Metrics Computation Point Requirements

7. SUMMARY AND CONCLUSIONS

This document illustrates the needs of security for an ISP network.

The first point concerns architecture security. This security may be split into 3 priorities (from the most important to the lowest):

- Signaling between measurement actors.
- Protection of the measurement actors.
- Confidentiality of the measures.

The second point concerns privacy. It is vital for an ISP to respect customers' privacy.

The third point concerns Denial of Service (DoS). It is important to assure the customers that the bandwidth will not be disturbed (by DoS attacks). This is an advantage in a competition context between ISPs. It may also reduce ISP costs since only customer's data traverses the network, avoiding free riders to use ISP's resources for free.

Despite the fact that denial of service is not the main security issue and is not directly within the scope of the 6QM project, it should be addressed urgently in QoS-based IPv6 networks because DoS attack makes it impossible to guarantee a SLA. Since QoS measurement systems take an important role in detecting DoS attacks it will be an added value for the 6QM project if its tools can be used in this area. This could be achieved in cooperation with other interested projects within the framework of joint cluster activities.

Regarding the measurement system developed in the context of 6QM, it is clear that it should be extremely secure to avoid the measurement system to be used to attack the measurement plane, the network elements processing performance or the back office through the management plane.

As IPsec is mandatory in IPv6 hosts, its usage will increase as privacy mechanism for the customers. Consequently the scope of usage of passive monitoring will reduce as far as the privacy guarantee will increase. That caricatures the main contradiction of network security: The most the privacy increases, the most the capacity of detection decreases.

References

[Adl02]	Tradeoffs in Probabilistic Packet Marking for IP Traceback, Micah Adler, 34th ACM Symposium on Theory of Computing (STOC), 2002.
[Alchaal02]	Offering a Multicast Delivery Service in a Programmable Secure IP VPN Environment, L.Alchaal, V. Roca and M.Habert, NGC2002, Boston, Massachusetts, USA, October 23-25, 2002.
[ammar99]	Paul Judge and Mostafa Ammar, "Gothic: A Group Access Control Architecture for Secure Multicast and Anycast", INFOCOM, 2002.
[AuraPhd00]	Tuomas Aura, "Authorization And Availability Aspects Of Open Network Security", Doctoral dissertation, Helsinki University of Technology, November 2000. Laboratory for Theoretical Computer Science research report HUT-TCS-A642000.
[Ballardie95]	T. Ballardie, J. Crowcroft, "Multicast-Specific Security Threats and Counter-Measures", 1995 Symposium on Network and Distributed System Security (SNDSS'95), February 16 - 17, 1995, San Diego, California.
[BCP38]	RFC 2827, Network Ingress Filtering, Defeating Denial of Service Attacks which employ IP address spoofing, P. Ferguson, D. Senie, May 2000.
[Beh02]	Tracking Attacks, Michael Behringer, Presentation, Cisco Systems, 2002.
[Bell01]	ICMP Traceback Messages, Steve Bellovin, Marcus Leech, Tom Taylor, Internet Draft, October 2001.
[Bur00]	Tracing Anonymous Packets to their Approximate Source, Hal Burch, Bill Cheswick, LISA XIV, December 2000.
[Cheung99]	S. Cheung, "An Intrusion Tolerance Approach for Protecting Network Infrastructures", Ph.D. Dissertation, University of California, Davis, September 1999.
[Cis00]	Unicast Reverse Path Forwarding, Documentation, Cisco Systems, 2000.
[Cis02]	IP Source Tracking on Cisco 12000 Series Internet Routers, Documentation, Cisco Systems, 2002.
[Cis98]	Quality of Service Policy Propagation via Border Gateway Protocol, Cisco Systems, 1998. Available at http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/bgpprop.htm
[Cis99]	Characterizing and Tracing Packet Floods Using Cisco Routers, Tech Note, Cisco Systems, 1999.
[clay99]	KHIP—a scalable protocol for secure multicast routing, Clay Shields, J. J. Garcia-Luna-Aceves, ACM SIGCOMM Computer Communication Review, Volume 29, Issue 4 (October 1999)
[DCU00]	Juniper DCU MIB, Juniper Networks Inc, 2000. Available at http://www.juniper.net/techpubs/software/junos44/swconfig44-install/html/mib-dcu.txt
[DCU02]	Juniper Networks Response to Flow-based Accounting Solutions, Chuck Semeria, Hannes Gredler, White paper, Juniper networks Inc., 2002.
[Dea01]	An Algebraic Approach for IP Traceback, Drew Dean, Matt Franklin, Adam Stubblefield, IEEE INFOCOM'01, March 2001
[Deering89]	S. E. Deering. "Multicast routing in internetworks and extended LANs". Computer Communication Review, 18(4), 1988. ACM SIGCOMM '88 Symposium:

	Communications Architectures and Protocols.
[Diot00]	Deployment issues for the IP multicast service and architecture, C. Diot, B. Neil Levine, B. Lyles, H. Kassem, D.Balensiefen, IEEE Network, 2000.
[Doe00]	Using Router Stamping to Identify the Source of IP Packets, Thomas W. Doeppner, Philip N. Klein, Andrew Koyfman, ACM Computer and Communications Security Conference, November 2000.
[draft-canetti-secure-multicast-taxonomy-00.txt]	A taxonomy of multicast security issues, B. Pinkas, R. Canetti, Internet draft, November 1998.
[draft-irtf-gsec-smrac-00.txt]	Haixiang He, Thomas Hardjono, Brad Cain, "Simple Multicast Receiver Access Control", Internet draft, November, 2001, Work in progress.
[Du00]	Trajectory Sampling for Direct Traffic Observation, N. G. Duffield, M. Grossglauser, ACM SIGCOMM'00, October 2000.
[Du02a]	Trajectory Engine: A Backend for Trajectory Sampling, N. G. Duffield, A. Gerber, M. Grossglauser, IEEE NOMS'02, April 2002.
[Du02b]	A Framework for Passive Packet Measurement, draft-duffield-framework-papame-01, Nick Duffield, Albert Greenberg, Matthias Grossglauser, Jennifer Rexford, Feb. 2002.
[Est02]	New directions in traffic measurement and accounting, Cristian Estan, George Varghese, ACM SIGCOMM'02, August 2002.
[Floy01]	Pushback Messages for Controlling Aggregates in the Network, Sally Floyd, Steve Bellovin, John Ioannidis, Kireeti Kompella, Ratul Mahajan, Vern Paxson, Internet Draft, July 2001.
[FreeS/WAN]	Linux FreeS/WAN, http://www.freeswan.org/
[Gib01]	The Strange Tale of the Distributed Denial of Service Attacks Against GRC.COM, Steve Gibson, July 2001, available at http://grc.com/dos/grcdos.htm
[Gil01]	MULTOPS: A data structure for bandwidth attack detection, Thomer Gil, Massimiliano Poletto, 10th USENIX Security Symposium, August 2001.
[Gong95]	Trade-offs in Routing Private Multicast Traffic, L. Gong and N. Shacham, In Proceedings of GLOBECOM '95, pages p. 2124-8, Singapore, November 1995.
[Goo02]*	Efficient Packet Marking for Large-Scale IP Traceback, Michael Goodrich, Ninth ACM Conference on Computer and Communications Security, November 2002.
[Gorinsky02]	Sergey Gorinsky and Sugat Jain and Harrick Vin and Yongguang Zhang, "Lightweight Protection Against Inflated Subscription in Multicast Congestion Control", To appear in ACM SIGCOMM 2003.
[Heady91]	The prototype implementation of a network level intrusion detection system, Richard Heady, George Luger, Arthur Maccabe, Mark Servilla, and John Sturtevant, Technical Report CS91-11, Department of Computer Science, University of New Mexico, April 1991.
[Hou01]	Trends in Denial of Service Attack Technology, Kevin Houle, George Weaver, v1.0 CERT Coordination Center, October 2001.
[Ioa01]	Implementing Pushback: Router defense against DDoS attacks, John Ioannidis, Steven M. Bellovin, NDSS, February 2002.

* *Not yet reviewed (not published).*

[Iolus]	Iolus: A Framework for Scalable Secure Multicasting, S.Mittra, SIGCOMM, 1997.
[Judge03]	Paul Judge and Mostafa Ammar: Paul Judge, Mostafa Ammar, "Security Issues and Solutions in Multicast Content Distribution: A Survey", IEEE Network, January/February 2003.
[Jun00]	Minimizing the Effects of DoS Attacks, Application Note, Sean Capshaw, Juniper Networks, November 2000.
[Ken01]	Aguri: An Aggregation-based Traffic Profiler, Kenjiro Cho, Ryo Kaizaki, Akira Kato, QofIS'2001, 2001.
[kent00]	Stephen Kent, Charles Lynn, Karen Seo, "Design and Analysis of the Secure Border Gateway Protocol (S-BGP)", Hilton Head, South Carolina, January 25 - 27, 2000
[Ker02]	SOS: Secure Overlay Services, A. Keromytis, V, Misra, D. Rubenstein, ACM SIGCOMM'02, August 2002.
[Ma01]	Intention Driven ICMP Traceback, A. Mankin, D. Massey, C.L.Wu, S.F.Wu, L. Zhang, Internet Draft, November 2001.
[Mah01]	Controlling High Bandwidth Aggregates in the Network, Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker, Draft, 2002.
[Man01]	On Design and Evaluation of Intention-Driven ICMP Traceback, A. Mankin, D. Massey, C.L.Wu, S.F.Wu, L. Zhang, IEEE International Conference on Computer Communication and Networks, October 2001.
[Mir02]	Attacking DDoS at the Source, Jelena Mirkovic, Gregory Prier, Peter Reiher, Technical report, UCLA, 2002.
[Mo01]	Inferring Internet Denial of Service Activity, David Moore, Geoffrey M. Voelker and Stefan Savage, 2001 USENIX Security Symposium, August 2001.
[Murcia300]	A.F. Gomez-Skarmeta, A.L. Mateo-Martinez, P.M. Ruiz-Martinez, "IGMPv3-based method for avoiding DoS attacks in multicast-enabled networks", 25th Annual IEEE Conference on Local Computer Networks (LCN'00), Tampa, Florida, November 08 - 10, 2000.
[Net02]	Netflow Performance analysis, White paper, Cisco Systems, 2002.
[Pa01]	An Analysis of Using Reflectors for distributed Denial of Service Attacks, Vern Paxon, Computer Communication Review, July 2001.
[Par01]	New Protocols to support Internet Traceback, C. Partridge, C. Jones, D. Waitzman, A. Snoeren, Internet Draft, November 2001.
[Park00]	On the effectiveness of probabilistic Packet Marking for IP Traceback under Denial of Service Attack, K. Park, H. Lee, CSD-TR 00-013, Research Report CERIAS, Purdue University, June 2000.
[Park01]	On the effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power Law Internet, K. Park, H. Lee, ACM SIGCOMM'01, August 2001.
[RFC2475]	RFC 2475, Architecture for Differentiated Services, Blake, et. al., December 1998
[San01]	Hardware Support for a Hash-Based IP Traceback, Luis A. Sanchez, Walter C. Milliken, Alex C. Snoeren, Fabrice Tchakountio, Christine E. Jones, Stephen T. Kent, Craig Partridge, and W. Timothy Strayer. 2nd DARPA Information Survivability Conference and Exposition, June 2001.
[Sav00]	Practical Network Support for IP Traceback, Stefan Savage, David Wethrerall, Anna Karlin, Tom Anderson, ACM SIGCOMM'00, August 2000.
[Sav01]	Security of IPv6 Routing Header and Home Address Option, Internet Draft, draft-

	savola-ipv6-rh-ha-security-01.txt, P. Savola, November 2001.
[smith97]	Securing Distance Vector Routing Protocols, Bradley Smith, MS Thesis, Computer Science, University of California, Santa Cruz, CA 95064, June 1997.
[Sno01]	Hash Based IP Traceback, Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer, ACM SIGCOMM'01, August 2001 – also IEEE/ACM Transactions on Networking (ToN), Volume 10, 2002
[Son01]	Advanced and Authenticated Marking Scheme for IP Traceback, Dawn Xiaodong Song, Adrian Perrig, IEEE INFOCOM'01, March 2001.
[Sto00]	CenterTrack, an IP overlay network for tracking DoS Floods, Robert Stone, 9th USENIX Security Symposium, August 2000.
[UU02]	Blackhole routing with ICMP backscatter, Chris Morrow, Uunet, 2002. available at http://www.secsup.org/Tracking/
[Wal02]	GOSSIB vs. IP Traceback Rumors, Marcel Waldvogel, 18th Annual Computer Security Application Conference, December 2002.
[Xu02]	Authenticated Multicast Immune to Denial of Service Attack, S. Xu, R. Sandhu, ACM SAC 2002, March 2002.
[Yam02]	Active Traceback Protocol, draft-yamada-active-trace-00.txt, T. Yamada, October 2002.
[Wang2000]	Feiyi Wang, Ph.D. dissertation: "Vulnerability Analysis, Intrusion Prevention and Detection for Link State Routing Protocols", Carolina State University, 2000.