A Filtering Approach for an IGMP Flooding Resilient Infrastructure

Zainab KHALLOUF [1][2] Vincent ROCA [2] Sébastien LOYE [1] Renaud MOIGNARD [1]

[1] *France Telecom R&D, Lannion, FRANCE; firstname.name@francetelecom.com*
[2] *INRIA Rhône-Alpes, Planète project, Grenoble, FRANCE; firstname.name@inrialpes.fr*

The recent advent of the Internet multicast service has enabled a number of successful real-time multimedia applications. However, the wide-scale commercial deployment of multicast has run into significant challenges, and in particular security. The threats to the multicast infrastructure, whether they are intentional or not, mainly come from the edge. Several attacks arise from the use of group management protocols, IGMP for IPv4 and MLD for IPv6. In this paper we adopt the point of view of the network operator, who is in charge of the deployment and management of the physical infrastructure. More specifically we propose and evaluate a simple yet efficient filtering approach to thwart some DoS attacks that are based on IGMP or MLD flooding, and that threat the operator's infrastructure. A key feature of our proposal is that it follows a realistic and pragmatic approach, and in particular it does not require any modification to the existing, widely deployed protocols.

Mots-clés: Infrastructure security; multicast; IGMP; MLD; flooding DoS attack

1 Introduction and Motivations

The current Any Source Multicast (ASM) service model [Dee89] is, by definition, an open model: any host can join any multicast group as a listener, and similarly, any host can start to transmit multicast traffic to a multicast group. To become a member of a particular group, end-hosts use a dedicated group management protocol, namely one of the three variants of the IGMP protocol [Dee89] with IPv4, or one of the two variants of the MLD protocol [DFH99] with IPv6. Multicast-capable routers then construct a distribution tree by exchanging routing messages with each other according to an intra-domain routing protocol. A number of protocols exist for building multicast forwarding trees, the most widely used being currently the PIM-SM protocol [EFH$^+$98].

IP multicast combines the best attributes of content-on-demand and bandwidth efficiency and allows operators to deliver a higher quality service to consumers. However, the wide-scale commercial deployment of multicast has run into significant challenges, and in particular security. Such attacks as the "Ramen Worm" attack [RRA02] have revealed that the multicast routing infrastructure is highly vulnerable to Denial of Service (DoS) attacks. If some of them specifically try to break the IP multicast service, others (like the "Ramen Worm" attack already mentioned) had a different goal but, as a side effect, also seriously damaged the network operators' infrastructure. Since these threats are mainly conducted at the edge, either intentionally or not, they are particularly easy to set up [KRML04].

Many proposals have been introduced to counter them. Yet most of them are unrealistic, for instance because they require to modify existing and widely deployed protocols, or they introduce authentication mechanisms, which is in practice almost impossible to deploy in legacy networks, and even useless, as we explained in a previous work [KRML04], since a corrupted host may be the source of a DoS attack, even if it has been authenticated.

Our proposal follows a completely different approach and offers a realistic and pragmatic solution to the problem. The proposal is based on a filtering component, placed between the end-users and the first hop multicast router, and we assume that this router is managed by the network operator.

The remaining of the paper starts with a detailed study of the problem and of related works in section 2 and 3. Then we introduce our proposal in section 4, explaining its implementation, and give an account of several experiments we conducted in section 5. Several extensions are possible to the basic scheme. We list some of them in section 6 and conclude this work in section 7.

2 A Survey of Group Management-Based Attacks

In this work we only consider edge attacks (see [KRML04] for a detailed description of other attacks), and more specifically *edge attacks that rely on the use of the multicast group management protocols*. Two major categories of edge attacks exist:

- data plane attacks: these attacks try to disturb the data forwarding functions in the routers.
- control plane attacks: these attacks try to disturb the signaling functions of the routers.

2.1 Data Plane Attacks

The current multicast model allows any host to join any multicast group by issuing an IGMP or (MLD with IPv6) *REPORT* message. Therefore, an attacker can simply join a certain number of high-bandwidth groups, causing the tree to expand and multicast traffic to be forwarded to him, thereby consuming some bandwidth along the distribution tree. In such an attack, the main victims are both the attacker's network and the multicast operator's network.

Congestion control attacks and high-rate data transmission attacks both belong to this category. Since they are not related to group management protocols, we do not discuss them.

2.2 Control Plane Attacks

Control plane attacks consist in disturbing the routing functionalities. Several threats exist:

- control message forging attacks
- control message flooding attacks

Let's start with message forging attacks. More specifically we can identify:

- *QUERY* message forging: A forged *QUERY* message from a machine with a lower IP address than the address of the current Querier will cause Querier duties to be assigned to the forger.
- *REPORT* message forging: A forged *REPORT* message may cause multicast routers to think there are some members for a new group on a subnet whereas this is false. A forged IGMP Version 1 *REPORT* message may put a router into "version 1 members present" state for a particular group, meaning that the router will ignore *LEAVE* messages, which will affect the IGMP efficiency (in particular the traffic to groups with no member will continue to be distributed after the departure of the last member, until the next *QUERY* request).
- *LEAVE* message forging: A forged *LEAVE* message will cause the Querier to send out Group-Specific *QUERY* messages for the group in question. This causes extra processing on the router and on each member.
- State-change *REPORT* Messages with IGMPv3: A forged State-Change *REPORT* message will cause the Querier to send out Group-Specific or Source-and-Group-Specific *QUERY* messages for the group in question. This causes extra processing on the router and on each member, but can not cause any data packet loss.

Let's consider control message flooding attacks now. These attacks consist in flooding the access router with a large number of IGMP control (*REPORT*, *LEAVE* and *QUERY*) messages. The flash crowd problem, which is the result of legitimate hosts joining massively a popular group at a given time, could be considered

as a special case of this family of threats. These flooding attacks essentially exhaust the memory resources used by the routers to maintain the states information, and its processing power used to process the control messages. Issuing a high number of *REPORT* messages for new groups will trigger the transmission of `JOIN` PIM-SM messages up to the RP, and will also impact the routing infrastructure, not only the first hop multicast router. The same is true when an attacker issues a high number of *LEAVE* messages. The target of our work is to make the routing infrastructure more resilient to these flooding attacks, even if preventing them totally is probably infeasible.

3 Related Works

Numerous techniques have been proposed by which an end system experiencing a DoS attack filters the offending traffic. [HTD03] [GR02] [Gli03] present techniques to mitigate the Dos through rate limiting approach. However these proposals differ from ours by the fact they aimed to tolerate the DoS attacks in the data plane while our approach addresses control plane attacks which is more important from the multicast operator's point of view.

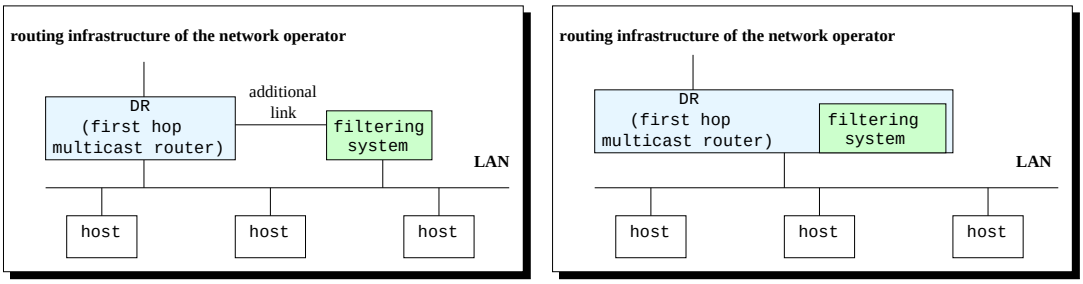
[JA02] [JA03] introduces Gothic, a group access control architecture for secure multicast and anycast, which is based on dedicated authentication and group key management schemes. On the opposite, we do not require the presence of security building blocks in our work since we believe this is neither realistic nor efficient [KRML04].

Finally our work shares some similarities with the MAFIA proposal [RA03], from the architectural point of view, since this latter also introduces an assistance node beside the first hop multicast router. However our approach has different goals and a completely different internal architecture.

4 Our Filtering Proposal

We now detail our filtering proposal.

4.1 Architectural Overview



(a) External deployment

(b) Internal deployment

Fig. 1: External versus internal deployment of the filter.

The proposed solution relies on a filter, managed by the network operator, and located between the clients and the first hop multicast router (figure 1). This filter captures the IGMP (or MLD) traffic generated by clients, filters them according to specific rules that will be detailed later, and sends them back to the network. The filter does not take part in any way to the packet forwarding functionality (still managed by the router as if the filter was not present). The multicast router is configured to accept IGMP packets exclusively from the filter, other IGMP packets being automatically rejected.

Deployment

Two deployments are possible:

- an external deployment (figure 1(a)) where the filter is implemented in an independent host, located at the operator's premises, beside the first hop multicast router, and it is connected both to the public LAN and to the multicast router through a dedicated link. Therefore this solution is universal, the only feature required from the router being the possibility to use an Access Control List (ACL) to only consider IGMP packets coming from the filter.
- an internal one (figure 1(b)) where the filter is integrated to the first hop multicast router.

Because we want to test the filter's efficiency using commercial equipments, we only consider the external deployment option in this paper.

Packet Classification

The filter captures incoming IGMP packets and classifies them according to their source IP address. Two categories of packets are defined:

- those coming from a client that is already known by the system, and
- those coming from a new, unknown client.

To that goal the filter keeps a context for each known client. This context contains the source IP address, the date of the last IGMP packet received, and a queue containing a maximum number of g packets for g different group addresses.

The Case of Known Clients

An IGMP *REPORT* or *LEAVE* packet issued by a *known client*, S_i, and related to group G_1, is enqueued to the associated queue of S_i. It may erase a previously enqueued packet if the queue already contains a packet related to group G_1. If the filter is correctly initialized, then IGMP packets issued by legitimate clients should not accumulate in the filter. This point is further discussed in section 4.4.

Periodically, a certain number of the IGMP packets enqueued in the known client lists are selected and sent back to the network. Those packets are the ones that will be accepted by the first hop multicast router. In order to guaranty fairness within the set of known clients, this scheduling follows a round-robin policy.

The Case of Unknown Clients

On the opposite, an IGMP *REPORT* or *LEAVE* packet issued by an *unknown client*, S_j, is systematically enqueued in a dedicated FIFO whose maximum size is strictly enforced.

Periodically, a certain number of IGMP packets enqueued in the unknown-client FIFO queue are elected, and the associated clients are accepted by the filter. A context is created for each client, and they are now "known" by the system. Therefore future incoming IGMP packets arriving from these clients will be directly accepted by the system and enqueued in their associated list.

Clients Purging

Because clients will disappear, a purging system is set up in the filter. Periodically a thread monitors all known clients and checks if each of them has been active during the period, i.e. has sent at least one IGMP packet. If the client has been silent, then it is dropped from the list of known clients and its context is removed.

Because the IGMP version 1 and version 2 *REPORT* suppression mechanism does not oblige each client to reply to a *QUERY* request, the purging period must be an order of magnitude larger than the IGMP *QUERY* polling period (often equal to 125 seconds, but this can be changed).

In case of IGMP version 3, there is no *REPORT* suppression mechanism and the purging period can be set to a value a little bit higher than the IGMPv3 *QUERY* polling period[†]. This is the optimal situation, and the list of active clients closely matches the reality, which warrants optimal classification performances and minimum memory requirements. Besides, if an attack occurs, creating contexts for ghost clients, then these context will quickly be removed from the system.

[†] Note that the explicit tracking of clients associated to a multicast group functionality can be enabled on Cisco routers when using IGMPv3. Network operators are typically interested by this functionality, even if it is not made mandatory by the IETF documents, since it will help to improve the network behavior.

4.2 A Closer View of the Various Building Blocks

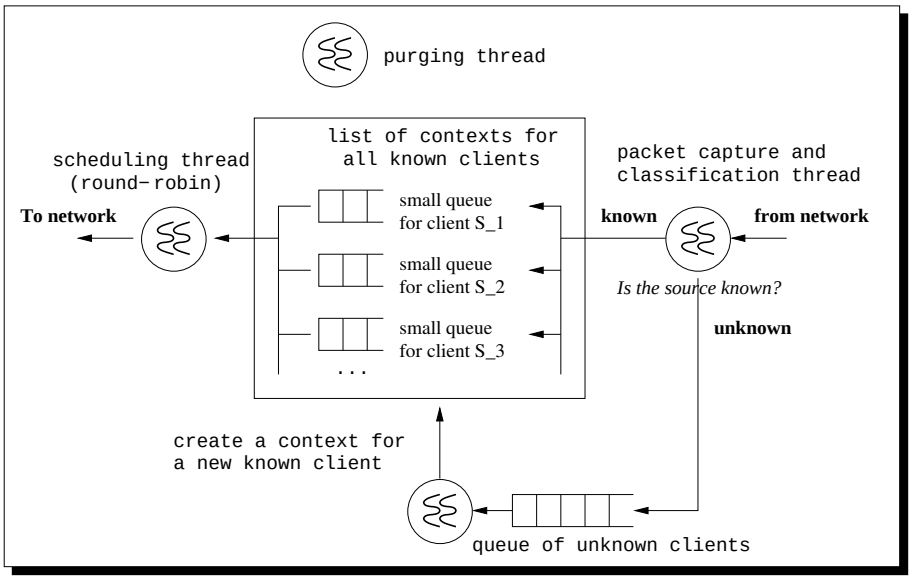


Fig. 2: Architecture of the filter.

Our filter is currently implemented at application level and involves various threads (figure 2):

- The packet capture and classifying thread: This thread relies on the `libpcap` library to capture all IGMP *REPORT* and *LEAVE* packets. The *QUERY* IGMP packets are not considered, since clients are not expected to issue them. This thread also performs a classification of each packet, as explained above. For performance purposes, a hash-based search algorithm is used by the classification function.

- The known clients queues creation thread: This thread elects, periodically, a certain number of IGMP packets enqueued in the unknown-client FIFO queue and creates queues for the associated clients.

- The main scheduling thread: This thread implements the round-robin scheduling of the waiting packets for the known clients. Each packet is then sent back to the dedicated link that links directly the filter and the first hop multicast router. The original source address of the IGMP packet must be kept in order to enable a correct operation of IGMPv3 (which unlike IGMPv2 is a stateful protocol).

- The purging thread: This thread periodically checks whether a client has been active or not. As explained above, this thread is only activated with a very low frequency in order to prevent "false positive" problems in IGMPv1 and v2 mode. This is different in IGMPv3 mode since the frequency will much higher.

4.3 Benefits in Front of a DoS Flooding Attack

The filtering mechanism has been designed to improve the resiliency of the multicast routing infrastructure in front of IGMP (or MLD) flooding DoS attacks. More specifically, two kinds of situations must be considered:

- the naive DoS attack, where the IGMP packets contain the attacker's IP address,

- the DoS attack where the IGMP packets contain a spoofed source IP address.

4.3.1 Naive Flooding DoS Attack Without IP Address Spoofing

If the source address of packets used by the attacker is its real address, this latter is most probably already known by the system, or if it is not the case, he will quickly be known. So packets are systematically enqueued in the associated list, whose size is by definition limited to at most *g* packets (all with a different group address). Since the attacker's IGMP packet arrival rate will most probably exceed its fair share of the filter outgoing rate, the associated queue will always overflow and the attack will easily be defeated.

4.3.2 Flooding DoS Attack With Random IP Address Spoofing

In the second case, the attacker uses systematically forged source IP addresses. If these addresses are chosen randomly, or if the subnet addressing space is much in excess of the possible legitimate clients, then most of the attacker's packets will not be known by the system, and therefore will enter the "unknown clients" FIFO queue. Since the service rate of this queue is low in front of the attacker's sending rate (by definition of a flooding attack), the FIFO will overflow. Of course some of the forged addresses will be accepted by the system and a context will be created for them. Yet if the attacker continues to randomly use source IP addresses, the accepted addresses will probably represent only a small fraction of the total number of addresses. Besides the maximum number of packets that will go through the filter for these accepted forged addresses will anyway be limited to the fair share of the filter outgoing rate. Both mechanisms will automatically defeat the attack, especially if this one has a limited duration.

4.3.3 Flooding DoS Attack With Targeted IP Address Spoofing

A more intelligent variant of this attack consists in using the range of possible addresses of the subnet, which in case of IPv4 networks, will most probably be limited to a few tens or hundreds of hosts. This kind of attack will more easily limit the benefits of our filter than the previous two kinds of attacks: (1) either these hosts are already known by the system (i.e. the legitimate client using this address has recently issued *REPORT* or *LEAVE* packets) and the attacker's packets will immediately be accepted, or (2) since there is a limited number of possibilities, if the attack duration is long enough, most addresses will finally be known by the system, after which further packets will immediately be accepted. In both cases, even if the impacts on legitimate clients will be serious, for instance preventing their legitimate IGMP packets from being sent to the first hop multicast router, the outgoing rate of packets sent to this router will not exceed the nominal outgoing rate of the filter. Since this rate takes into account the capabilities of the multicast routing system (section 4.4), the attack will not have any other impact than preventing legitimate clients of this subnet from using multicast services. The DoS attack is, in that case, confined to a few clients, but does not impact the whole multicast routing infrastructure of the operator.

In another variant, the attacker deliberately uses the filter's source IP address, in order to make its IGMP packets be accepted by the first hop multicast router. This attack is in fact easily defeated by having a direct link between the multicast router and the filter (the "additional link" of figure 1(a)). The first hop multicast router only accepts IGMP packets arriving from this link, rather than from the shared subnet. Similarly, the filter is configured to ignore IGMP packets having one of its IP addresses as a source address. Note also that in an integrated implementation, the filter will be integrated to the first hop multicast router itself, rather than being implemented within an independent host attached to the multicast router. In that case the attack is also easily defeated.

4.4 Initializing the System

The filter is, thanks to its design, easy to configure. In particular it automatically learns the set of trusted clients (even if some of them may be spoofed IP addresses). Yet there is a small set of key parameters that must be initialized suitably for a given environment:

- the size of the unknown client FIFO queue; this queue cannot be too small, since many new but legitimate clients may be interested by joining a session at a given time, for instance because they have been informed of its existence synchronously;
- the unknown client queue service rate: this is the number of new clients that can be accepted by the system per time unit. Its value is a balance between the reactiveness of the system, in front of new

legitimate clients, and the protection in front of an attacker that would spoof the source IP address of the IGMP packets it generates;

- the scheduling rate: this is the number of IGMP packets for known clients that can be accepted by the round-robin scheduling thread per time unit. Its value is directly related to the processing capabilities of the first hop multicast router (CPU and memory), and that of the whole multicast routing infrastructure since IGMP packets may trigger the creation or pruning of multicast branches in the operator's infrastructure;
- the maximum number of waiting IGMP packets for different groups per known client: this parameter protects the filter from an attacker known by the system that would generate a large number of IGMP packets. Since it is expected, in a correctly dimensioned filter, that IGMP packets for known clients do not stay too long in the filter, this maximum number of waiting packets per client should be low;
- the maximum number of known clients managed by the system: this parameter should be adjusted according to the simultaneous number of potential clients, which is usually known by the operator. An upper bound exists (especially in IPv4 environments), namely the number of IP addresses made possible by the IP subnet. Of course the maximum number of clients should not exceed the processing capabilities of the filter (CPU and memory);
- the purging period: the purging period must be adapted to the specificities of the target environment, and in particular whether only IGMP version 3 hosts are deployed or not (section 4.1);

There is clearly trade-offs to find when initializing the filter. These trade-offs must take into account various external parameters that are specific to the target environment: the simultaneous number of potential clients, the number of groups they may be interested in, the frequency with which they may join or leave these groups (e.g. because of zapping in a TV/ADSL environment). Finding appropriate trade-offs and good heuristics will be addressed in future works, even if this is of utmost practical importance.

5 Experimental Evaluation

5.1 Experimental Environment and Scenario

To validate the effectiveness of our filter, we carried out several experiments on a small testbed. This testbed consists of one end-host, running the IGMP traffic generator for both legitimate clients and the attacker. The filter is attached to the same Ethernet LAN, as well as the first hop multicast router, a Cisco 7500 RSP 12.2, running PIM-SM and acting as the IGMP querier. Behind this router is the RP, a Linux router running PIM-SM. PCs are equipped with Intel Pentium IV-2.5 GHz processors, 1 GB RAM and Mandrake Linux.

The test lasts 540 seconds and follows a three step scenario:

- $[0; 540s]$ (whole test): 500 legitimate sources send 15 IGMP *REPORT* per second over 50 groups during the whole test duration;
- $[180; 540s]$: in parallel, another 500 legitimate sources send 5 IGMP *REPORT* per second over 50 groups different from the groups reported by the first legitimate clients. The goal of introducing new clients here is to evaluate the impact of the attack and the filter on new unknown but legitimate clients;
- $[180; 360s]$: in parallel, an attacker spoofs 50 source addresses and sends 100 IGMP *REPORT* packets per second, with group addresses chosen within a set of 500 addresses other than those requested by the legitimate clients.

To summarize, there are three periods: (1) a first period where there are only legitimate clients, (2) a second period where a flooding DoS attack is launched, and in parallel new clients arrive, and (3) a third period where there are only legitimate clients.

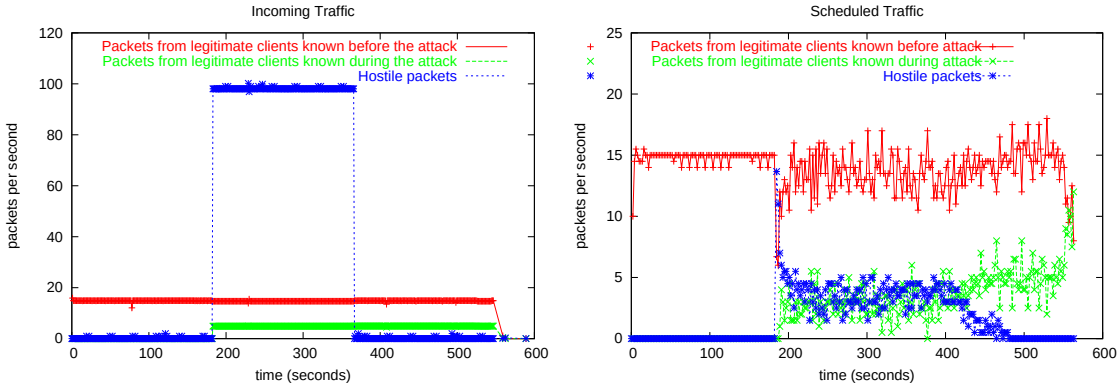
The filter is initialized as follows:

- size of the unknown client queue = 20 packets

- unknown client queue service rate = 20 pps
- scheduling rate = 20 pps
- maximum number of waiting packets per client = 6 packets
- unlimited maximum number of known clients

Note that the purging period is not significant in our tests since IGMP *REPORT* packets are not generated when the IGMP querier requests them but automatically. These values have been set experimentally. As explained in section 4.4, future works will consist in finding good heuristics to initialize the filter, depending on the environment.

5.2 Effectiveness of the Filter on the Traffic



(a) Incoming traffic

(b) outgoing traffic

Fig. 3: Traffic Entering and Leaving the filter.

Max. memory requirements for IGMP, **without filter**	601 multicast routes, or 293 288 bytes
Max. memory requirements for IGMP, **with filter**	458 multicast routes, or 223 504 bytes

Tab. 1: Maximum memory requirements of the IGMP process, on the Cisco router, without/with filter.

Figure 3 (a) shows the traffic entering the filter and resulting from both the legitimate clients and the attacker, and figure 3 (b) the traffic leaving the filter and entering the first hop multicast router. These curves show the effectiveness of the filter in limiting the IGMP flooding traffic. The attackers receives a share of the outgoing flow that is significantly lower (≈ 20 times) than the attacker's traffic incoming rate. Besides, new clients can still be accepted during the attack, even if they are not all elected.

The filter behavior in case of a flooding attack without IP address spoofing would have been excellent. Since there are at most 6 waiting IGMP packets per client (section 5.1), the attacker receives a maximum share of: $6/total\_number\_known\_clients \leq 6/500 = 0.012$ of the outgoing rate.

Let's now look at the router. Table 1 shows clearly the benefits of the filter on the number of groups known by the Cisco router, and the associated memory requirements[‡]. The filter reduces the maximum memory requirements significantly (23.8% reduction) compared to the situation where the router receives directly the attacker's traffic. Yet this test does not catch benefits in terms of CPU activity and PIM signaling traffic on the router. This is left to future works.

[‡] Obtained through the following command: *show ip mroute count | include routes*

6 Extensions to the System

The filter detailed so far defines the basic architecture. Several extensions can be added to it, depending on the specificities of the target environment (we only considered a deployment within a LAN so far). For instance the filter may be linked to a policy server that lists the authorized clients, or specifies a maximum number of groups an authorized client can join at any time, or provides access control directives to the clients. This information can be used by the filter to enforce the desired policy, by refusing or not some incoming packets, and therefore extending the classification criteria beyond the question of whether a client is known or not.

The filter may also be integrated into a multicast enabled broadband access network (xDSL). The core network delivers video content from the head-end to the appropriate end points, where Ethernet or ATM Layer 3 switches aggregate traffic from Digital Subscriber Line Access Multiplexers (DSLAMs) and Multi-service Access Platforms (MAPs). One approach to integrate the filter within the broadband environment is to modify the architecture of the filter to maintain a mapping of layer 3 group addresses to ATM addresses or to VLAN tags. In this case clients are automatically identified by unforgeable identifiers (rather than IP addresses that can be spoofed), which significantly improves the efficiency of the filtering system and almost completely prevents (non distributed) flooding attacks.

7 Conclusions and Future Works

We have introduced in this paper a filtering approach to make the multicast routing infrastructure resilient (up to a certain point) to IGMP or MLD flooding DoS attacks. These attacks are among the most easily launched, essentially because they can be launched by end-hosts with trivial flooding programs, while having potentially major impacts on the network operators' routing infrastructure,

The proposed solution follows a realistic and pragmatic approach, in particular because it does not require any modification to the existing, widely deployed, group management and multicast routing protocols. In particular it does not assume the presence of authentication mechanisms that cannot prevent flooding attacks to occur.

We have shown how an appropriate filter can largely reduce the impacts of an IGMP based flooding attack, even if some of the attacker's traffic will go through the filter. Preliminary experimental results seem to support our claim. Yet these experiments are for sure in a preliminary stage and more exhaustive tests, also involving the multicast routers (DR and RP), are needed. Several other attack scenarios will be added, in order to evaluate the efficiency and the limitations of the filter under several traffic models.

As we explained in section 4.4, the filter must be carefully initialized to be efficient. We are currently elaborating several heuristics to that purpose, that take into account some key features of the target environment.

The filter can also be coupled to a policy server, that can help the filter to take appropriate classification and scheduling decisions with respect to the incoming data flow. This is a direction that will be further analyzed in a near future.

To conclude the filter building block introduced in this work can turn out to be very helpful to network operators, even if it has limits. How far it can help is still an open question.

References

[Dee89] Steve Deering. Host Extensions for IP Multicasting, August 1989. IETF (Request for Comments) RFC 1112.

[DFH99] Steve Deering, Bill Fenner, and Brian Haberman. Multicast Listener Discovery (MLD) for IPv6, October 1999. IETF (Request For Comments) RFC 2710.

[EFH+98] Deborah Estrin, Dino Farinacci, Ahmed Helmy, Dave Thaler, Steve Deering, Van Jacobson, Mark Handley, Charley Liu, Puneet Sharma, and Liming Wei. Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification, June 1998. IETF/IDMR (Request for Comments) RFC 2362.

[Gli03] Virgil Gligor. Guaranteeing Access in Spite of Service-Flooding Attacks. In *International Workshop on Security Protocols, Sidney Sussex College, Cambridge, UK*, 2003.

[GR02] Aman Garg and A.L.Narasimha Reddy. Mitigating Denial of Service Attacks Using QoS Regulation. In *Tenth International Workshop on Quality of Service (IWQoS 2002)*, Miami Beach, USA, May 2002.

[HTD03] Kai Hwang, Sapon Tanachaiwiwat, and Pinalkumar Dave. Proactive Intrusion Defense against DDoS Flooding Attacks. In *IEEE Security and Privacy Magazine (submitted under review)*, 2003.

[JA02] Paul Judge and Mostafa Ammar. Gothic: A Group Access Control Architecture for Secure Multicast and Anycast. In *IEEE INFOCOM 2002*, NY,USA, June 2002.

[JA03] Paul Judge and Mostafa Ammar. Security Issues and Solutions In Multicast Content Distribution: A Survey. *IEEE Network magazine special issue on Multicasting*, January/February 2003.

[KRML04] Zainab Khallouf, Vincent Roca, Renaud Moignard, and Sébastien Loye. Infrastructure Sécurisée de Routage Multipoint : le Point de Vue de l'Opérateur de Réseau. In *3ème Conférence sur la Sécurité et Architectures Réseaux (SAR'04), La Londe, France*, June 2004.

[RA03] Krishna N. Ramachandran and Kevin C. Almeroth. MAFIA: a Multicast Management Solution for Access Control and Traffic Filtering. In *IEEE/IFIP Conference on Manamgement of Multimedia Networks and Services*, September 2003.

[RRA02] Prashant Rajvaidya, Krishna Ramachandran, and Kevin C. Almeroth. Detection and Deflection of DoS Attacks Against the Multicast Source Discovery Protocol. Technical report, Department of Computer Science, University of California, Santa Barbara, July 2002.