Monitoring the Neighbor Discovery Protocol

Frédéric Beck, Thibault Cholez, Olivier Festor, Isabelle Chrisment

Monitoring the Neighbor Discovery Protocol

Frederic Beck, Thibault Cholez, Olivier Festor and Isabelle Chrisment
LORIA - INRIA Lorraine
Campus Scientifique - BP 239
54506 Vandoeuvre-ls-Nancy Cedex, France
Email: {frederic.beck,olivier.festor,isabelle.chrisment}@loria.fr
thibault.cholez@esial.uhp-nancy.fr

***Abstract*— While IPv6 is increasingly being deployed in networks, including ISPs, the need to monitor and manage the associated protocols increases. In this paper we focus on the Neighbor Discovery Protocol and we motivate the importance to monitor it. We also present our approach for this task together with the functionalities we provide and the software, NDPMon, that we developed.**

I. Introduction

In most IP networks, administrators deploy several tools to manage and monitor their networks, either to collect measurements about the performance, or to ensure the integrity and security of them. One of these tools is ArpWatch [1] which monitors the pairing between IPv4 and Ethernet addresses, and ARP [1] activities. This simple tool is very useful in IPv4 networks to know about the hosts present on the network, discover the new ones appearing, or detect misconfigurations or misbehaviors.

The exponential growth of the Internet in the nineties brought several problems, due to drawbacks in the IPv4 protocol. To overcome these limits, a new version of the IP protocol, IPv6 [2] was defined, bringing with it advanced built-in services. One of the objectives of IPv6 was to ease the addressing of the hosts, the discovery of network components... To achieve this goal, the Neighbor Discovery Protocol [3] was defined. This protocol does not only replace ARP, it is more complex and offers different services.

As IPv6 is being deployed on more and more networks, the same needs in terms of management and monitoring appear. In the last years, many studies have been made, and robust tools for monitoring and configuring IPv6 networks have been developed. However, some lacks subsist in the management plane, including the monitoring of the Neighbor Discovery Protocol.

In this paper, we present an adaptation of the ArpWatch model for IPv6. The paper is structured as following. Section 2 presents the Neighbor Discovery Protocol and its vulnerabilities. In Section 2 we highlight the interest in monitoring this protocol. Section 4 describes our approach and the monitoring architecture we propose. The NDPMon software that we implemented is detailed in Section 5. Before concluding, we give an overview of our future work.

[1] http://ee.lbl.gov/

II. Neighbor Discovery Protocol

In this section, we will first present the Neighbor Discovery Protocol, and then describe the vulnerabilities of this protocol and the attacks which can be done against it.

A. Presentation

The Neighbor Discovery Protocol [3] is part of the ICMPv6 [4] protocol. It is used by IPv6 nodes to interact with each others via ICMPv6 messages, and replaces ARP [1], IPv4 Router Discovery [5], IPv4 Redirection, and offers new functionalities. The most known one is the IPv6 Stateless Address Autoconfiguration [6], which makes possible for a node to configure the addresses and routes on an interface automatically, simply by connecting it to the network link. The Duplicate Address Detection (DAD) algorithm [6] ensures during this procedure that the address assigned on the interface is not already used on the network.

But in order to communicate, the link-layer's address is not sufficient for a node, it needs several types of informations. Different procedures exist within the Neighbor Discovery Protocol to discover these particular parameters and/or options:

- Router Discovery: hosts locate the on-link routers,
- Prefix discovery: hosts discover the set of on-link IPv6 prefixes,
- Address Resolution: IPv6 addresses are resolved to their data link-layer address (see figure 1),
- Next-Hop Determination: used to determine the IPv6 address (link local or global address) of the neighbor which acts as gateway for the given destination; it can be a router or the node itself,
- Neighbor Unreachability Detection: check the reachability of a neighbor,
- Redirect: routers inform a hosts that a better first hop exist to reach a given destination.

The Neighbor Discovery Protocol defines five messages which are used during these procedures:

- Router Solicitation: sent by a node which wants to configure itself, to ask information about the on-link routers and prefixes,
- Router Advertisement: sent periodically, or in response to a Router Solicitation, by a router; contains the default gateway's address, the router's validity, the list of IPv6

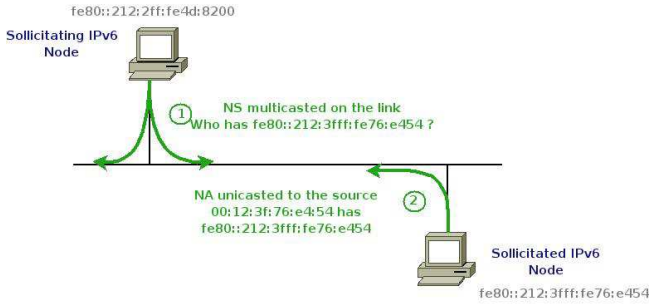

Fig. 1. Address Resolution

prefixes the router handles, MTU, Mobile IPv6 [7] options...

- Neighbor Solicitation: sent by a node to query a neighbor (physical address, reachability, or availability of an address during the DAD),
- Neighbor Advertisement: sent periodically by a node, or in answer to a Neighbor Solicitation, in order to advertise the physical address of an interface,
- Redirect: used by routers to advertise a better route.

B. Vulnerabilities and Attacks

The Neighbor Discovery Protocol, as it was defined, is vulnerable to different types of threats and attacks [8]. There are three main types of threats.

- Redirect attacks: a malicious node redirects packets away from the receiver to another node on the link,
- Denial of Service (DoS) attacks: a malicious node prevents communication between the node under attack and all other nodes, or a specific destination address,
- Flooding DoS attacks: a malicious node redirects other hosts' traffic to the victim.

A Redirect attack can lead to a DoS, if the packets redirected to a node are dropped, or forwarded to another host. These attacks can be performed on purpose by a malicious node, but they can also simply reflect a host's misconfiguration.

We are going to describe more precisely the attacks in the following subsections, organizing them in three types: non routing related attacks, routing related attacks and remotely exploitable and replay attacks.

1) Non routing related threats: These are attacks against the "pure" Neighbor Discovery functions.

By spoofing Neighbor Advertisements, a malicious node can make the Duplicate Address Detection process fail for a host which is trying to assign itself an address with autoconfiguration, and the host will stop to claim that address. That host may never be able to obtain a valid IPv6 address.

Sending spoofed advertisements can also overwrite the neighbors' caches on nodes. Then, some packets may be sent to the wrong destination, which leads to services DoS.

When a service or host is not responding, the Neighbor Unreachability Detection is launched, to check the reachability of the node. A Neighbor Solicitation is sent to the node, and a Neighbor Advertisement is expected in return. A malicious node can keep sending fake Neighbor Advertisements. If the procedure fails, the upper-layer's traffic is stopped, but here it wouldn't be the case, as the host would still be considered as reachable. This may introduce infinite delays to detect services' outage.

2) Routing related threats: These threats are relevant to router discovery or router related mechanisms.

If a malicious node sends a spoofed Router Advertisement, hosts may select the attacker as default router. Thus, the attacker can siphon off traffic from hosts, or mount man in the middle attacks (see figure 2). If it then forwards the traffic to a legitimate router, this can be completely transparent for the hosts. If an invalid prefix is advertised in the advertisement, it may be used by hosts for autoconfiguration. As a result, an invalid source address may be used, and the packets sent by the misconfigured hosts may never reach the destination. If the spoofed prefix is a valid prefix, but from another link or network, the hosts will believe this prefix is on-link, and will never send packets for this prefix to the default router, and thus, this prefix will be unreachable for the nodes. Such a spoofed router Advertisement may also duplicate the ones from the legitimate router, but modify other parameters, such as disabling autoconfiguration for hosts or specifying a hop limit too small for packets' routing. The default router can also be "killed", either by launching a classical DoS on the default router, and spoofing advertisements, so that the nodes believe all the destinations are on-link, or by duplicating the Router Advertisements with a router lifetime of zero. Finally, the attacker may send spoofed Redirect messages with a legitimate source address, to send packets for a given destination to any link-layer address on the link. This address can be invalid, as long as the attacker responds to Neighbor Unreachability Detection probes.

3) Replay attacks and remotely exploitable threats: All Neighbor Discovery messages are prone to replay attacks. An attacker which is able to capture valid messages, will be able to replay them later, even if the messages are cryptographically protected, unless the cryptographic mechanism is protected itself.

A malicious node can also perform a classical DoS against the Neighbor Discovery Protocol. If it generates well formated IPv6 addresses for a given prefix, and sends packets to these, the last hop router will have to resolve these addresses by sending Neighbor Solicitation packets. If the frequency is high enough, a new host entering the target network may not be able to perform autoconfiguration, as the router will be busy resolving the fake addresses.

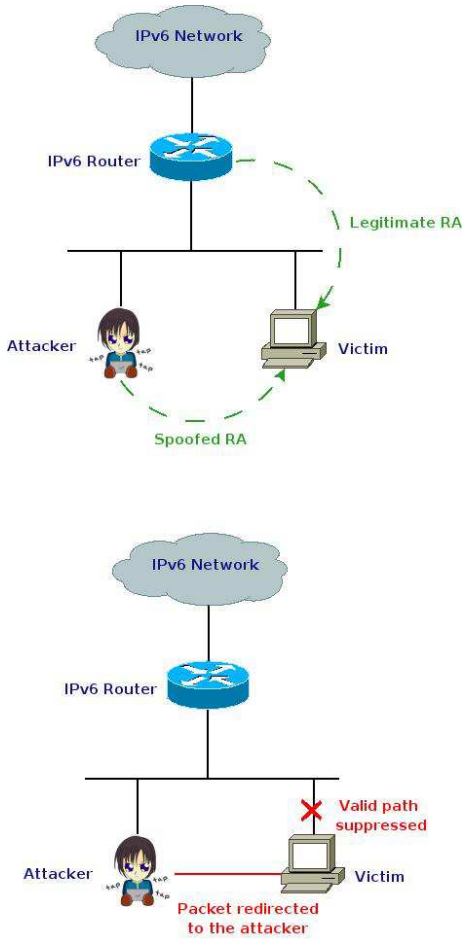

Fig. 2. Redirect traffic with spoofed RA

III. Securing or Monitoring ?

Section II-B highlighted some vulnerabilities in the Neighbor Discovery Protocol. As some of them are quite easy to exploit, there is a real need to secure the various functions in this protocol.

To respond to this need, SEND, SEcure Neighbor Discovery [9], was defined. A new set of Neighbor Discovery options is used, in order to protect Neighbor Discovery messages. Besides these new options, this solution introduces several new components in the classical neighbor discovery architecture.

The first component is certificate paths. Each host on the link must be configured with an anchor to which the router has a certificate path before selecting it as its default router. Certificate Path Solicitation and Advertisement messages are used to discover these paths.

To make sure the sender of a Neighbor Discovery message is the owner of the claimed address, Cryptographically Generated Addresses [10] are used. Before claiming an address, all hosts generate a public-private key pair. A new option, the CGA option, is used to carry the public key and its associated parameters. Figure 3 shows how these addresses are generated.

The RSA Signature option is used to secure all Neighbor

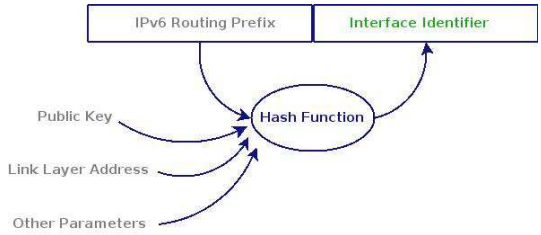

Fig. 3. Cryptographic Addresses Generation

and Router Discovery messages. This option protects the integrity of the message and authenticates the identity of their sender. The authority of a public key is established either with the authorization delegation process, by using certificates, or through the address ownership proof mechanism, with Cryptographically Generated Addresses, or even with both.

To prevent the replay attacks, the options Timestamp and Nonce have been introduced. The Timestamp option provides replay protection without any previously established state or sequence number. When the messages are used in solicitation/advertisement pairs, the Nonce option is used.

But this infrastructure is complex to deploy, especially because of the usage of certificates. Moreover the usage of Cryptographically Generated Addresses introduces a longer treatment time for the Neighbor Discovery messages, which can be harmful, for example if Mobile IPv6 is deployed. Finally, there are not many implementations of SEND. for these reasons, SEND has not been deploy widely on the Internet yet.

However, if IPv6 is known as easily vulnerable and open to attacks, it will not be deployed neither. For IPv4 networks, ARP is not secured, but a tool, ArpWatch, for monitoring this protocol is widely used. This tool monitors the ARP activities, and sends alerts if required. The main advantage to the secure version, is that this tool is easy to deploy, and does not required an architecture as complex as SEND's one.

For these reasons, we decided to propose a monitoring architecture for the Neighbor Discovery Protocol, which we will present in the section IV. This tool will detect unexpected behaviors of the Neighbor Discovery Protocol and send reports to the administrator or to another management application which will be in charge of taking the required action on the network.

IV. Our Monitoring Architecture

In this section, we will present our proposal for monitoring the Neighbor Discovery Protocol. Basically, the idea is to implement an IPv6 version of ArpWatch. But, as the Neighbor Discovery Protocol is more complex than ARP, some other functionalities are required. We will first describe ArpWatch and how we used it as a reference to define our monitoring architecture, and then we will present the additional functionalities in our proposal.

A. Monitor the Neighbor Discovery Activities

ArpWatch has been used as a reference to define the monitoring architecture. It was developed by the Lawrence Berkeley National Laboratory. Its role is to monitor ARP's activity. By analyzing ARP packets, it maintains up-to-date a cache in which are stored the pairing between IPv4 and Ethernet addresses for all hosts on the link. A Timestamp is associated to each entry in this cache, which enables a monitoring over the time of host's activity. When an ARP packet is captured, it is compared to the information in the cache, and if a suspicious behavior, or special activities (a new station has appeared on the network) is detected, a report is sent to the administrator. These reports are sent via syslog and mail for the most important ones.

Our first objective is to define a tool which would perform the same monitoring tasks, but for IPv6. This tool is in charge of monitoring the Neighbor Discovery Protocol's activities and maintains up-to-date a neighbor database which contains the correspondences between IPv6 and Ethernet addresses, alongside with a Timestamp. When a Neighbor Discovery packet is captured, the content is compared to the entries in the database. Usually, in IPv4, only one address was assigned to an interface. In IPv6, multihoming is one of the key features, for Network Renumbering [11] for example. When defining the neighbors' cache entries, we have to take this specificity into account. In the same way than ArpWatch, activities and suspicious behaviors raise alerts and reports. All these alerts are sysloged, and depending on the syslog daemon's configuration, they can be sent to a remote station. The most severe ones, namely the suspicious activities, raise alerts which are sent by mail to a defined address, by using an external Mail Transfer Agent (MTA).

The Neighbor Discovery activities monitored by the tool are:

- new station: a new Ethernet address appears on the network, a new node has appeared
- new activity: the source node had no activity during the last month
- bogon: the IPv6 source address is not local to the link
- ethernet mismatch: the Ethernet address specified in the ICMP option is not the same than the Ethernet source address of the packet
- changed ethernet address: a node changed its Ethernet address while keeping the same IPv6 address
- flip flop: a node is switching between two different Ethernet addresses
- reused address: a node is re-using an old Ethernet address

The tool is aimed to being deployed on each subnet or link of the network, as shown in figure 4. It integrates a learning phase. During this phase, it builds the neighbors' database by capturing the Neighbor Discovery messages, while not sending any report. The tool makes the assumption that, when it enters this phase, the network is healthy and that all the activities are legitimate. Once this phase is over, the tool can be restarted in monitoring mode.

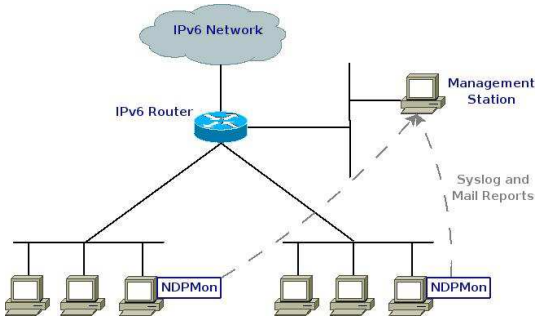

Fig. 4. Monitoring tool deployment

B. Additional Functionalities

In section II-B, we highlighted several attacks possible against the Neighbor Discovery Protocol. Monitoring only the classic Neighbor Discovery activities is not sufficient for detecting these addresses. But to detect these attacks, besides the neighbors' cache, we must also defined the legitimate behavior for the routing on the link. By defining the routers' Ethernet and IPv6 valid addresses, and the legitimate prefixes on the link, we can detect the following attacks:

- wrong router mac: the Ethernet source address of the Router Advertisement is not defined as valid
- wrong router ip: the IPv6 source address of the Router Advertisement is not defined as valid
- wrong prefix: the prefix advertised in the Router Advertisement is not legitimate on the link
- wrong router redirect: the source of the Redirect message is not a legitimate router
- NA router flag: the Neighbor Advertisement has the Router flag set whereas it is not defined in the known routers list
- DAD DoS: Denial of Service toward the Duplicate Address Detection mechanism
- ethernet broadcast: the Ethernet source address is the broadcast address
- ip broadcast: the IPv6 source address is a specific multicast address

When detected, all these behaviors trigger the sending of syslog and mails alerts. Some of them are not only the manifestation of a malicious behavior or a misconfiguration, they symbolize vulnerabilities in the IPv6 stack. For example, depending on the version of the Linux kernel, and thus of the IPv6 stack, it is possible or not to assign an Ethernet broadcast address on an interface. If such a behavior is detected, it means that the host on which this address is set has an old version of the IPv6 stack, and that it could be judicious to update it.

C. Integration in a monitoring framework

This architecture is aimed at being integrated in a wider monitoring framework based on NetSV [2], the IPv6 Network

[2]http://netsv.sourceforge.net

Renumbering SuperVision tool. NetSV monitors the renumbering procedure and validates the addressing of end hosts. Our architecture monitors the Neighbor Discovery Protocol activities and detects some attacks against this protocol. The renumbering procedure is based on this protocol. Our architecture can thus be used to give more precise informations about the network's state, and help NetSV to validate the transitions between the different steps of the procedure. By working together, NetSV and our monitoring architecture provide a more complete monitoring of the renumbering procedure, and make possible to detect attacks performed by a malicious host to prevent the renumbering from succeeding.

V. Implementation and Tests

In this section we present the implementation of our architecture that we described previously, and the tests performed to validate the tool. The tool is called NDPMon, Neighbor Discovery Protocol Monitor.

A. Implementation

The software, we called NDPMon, has been implemented in C language, must be launched with root privileges and runs as a daemon. To limit as much as possible the resources consumption, we performed some code profiling with the tool ValGrind [3]. NDPMon uses two XML files which are parsed and modified by libxml2 [4] (cf Figure 5).

The first file is the configuration file which contains two main parts. Firstly, there are some options for the configuration of the daemon itself: syslog facility to use, Email address for the reports... Secondly, the administrator must give the list of the legitimate routers on the link (Ethernet and IPv6 addresses) and a list of network prefixes authorized. These informations are used by the daemon as the legitimate routing informations on the link, and are used as references when Neighbor Discovery packets are received.

The second file is the neighbor cache. It contains the pairings between IPv6 and Ethernet addresses. A Timestamp is set for each entry in the cache. This file is filled automatically by the daemon. If NDPMon runs in learning phase, this database will be constructed without sending any report.

When the tool is launched, these two files are parsed. The neighbor cache is loaded and the entries are convert in C structures. The routing informations in the configuration file are converted in the same way. The daemon begins to listen to the network and captures Neighbor Discovery packets thanks to the libpcap [5]. These packets are converted in C structures. The tool implements analysis functions to verify if the packet is legitimate, or if it is a malicious packet (as defined in section IV).

If the packet is considered as malicious, reports are sent via syslog, and, depending on the severity, a mail is sent to the administrator. If the packet is legitimate, the informations it contains are added in the neighbor cache if they are not already present, otherwise the Timestamp of the corresponding is simply updated.

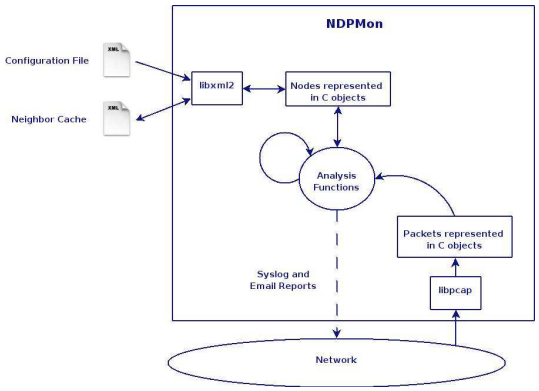

Fig. 5. Implementation Architecture

B. Validation Tests

To validate each monitoring feature described in section IV, NDPMon underwent an experimental validation. In this section, we will present a scenario in which NDPMon is deployed and running on a link, and detects some attacks.

The IPv6 Hacking Tools [6] have been used to generate the attacks. It is a set of tools to attack inherent protocol weakness of IPv6 and ICMPv6. It provides an easy to use API to write small programs to simulate Neighbor Discovery attacks thanks to the included packet factory library. Moreover, some pre-written executable to generate basic ICMPv6, and especially Neighbor Discovery, messages are available, which makes fast and easy the tests.

Let's consider a basic network, with one router and three IPv6 nodes, as shown in figure 6.

At startup, NDPMon first builds the neighbors' cache. Here follows the syslog reports generated by the daemon:

```
NDPMon[26345]:  new station 0:30:b6:51:d4:1c \
    fe80::230:b6ff:fe51:d41c
NDPMon[26345]:  new station 0:13:72:14:c4:58 \
    fe80::213:72ff:fe14:c458
NDPMon[26345]:  new station 0:11:11:4d:82:0  \
    fe80::211:11ff:fe4d:8200
NDPMon[26345]:  new station 0:13:72:14:c4:58 \
    fe80::213:72ff:fe14:c458
```

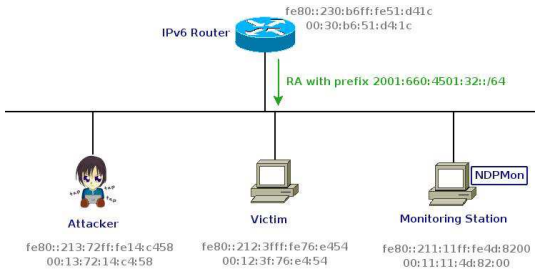

Fig. 6. Validation testbed

[3] http://valgrind.org/
[4] http://xmlsoft.org/
[5] http://www.tcpdump.org/
[6] http://www.thc.org/thc-ipv6/

As long as the network remains in that state, no new reports will be sent. In the rest of the section, all reports shown are syslog messages. Thanks to the IPv6 hacking tools, the attacker can begin to send spoofed Neighbor Discovery messages. In the following example, the attacker sends a fake router advertisement, by spoofing the router's Ethernet and IPv6 addresses, but with an invalid network prefix:

```
root@attacker:~/thc-ipv6-0.7# ./fake_router6 eth0 \
  fe80::230:b6ff:fe51:d41c 2001:660:4501:3201::/64 \
  1500 0:30:b6:51:d4:1c
```

The victim will accept the Router Advertisement as it is well formed, and consider the prefix 2001:660:4501:3201::/64 is on-link. NDPMon receives the message, and compares the informations contained to its database with the analysis functions. The Ethernet and IPv6 source addresses are valid, but the prefix is not legitimate on the link. Thus, it will send a report by mail to the administrator and write the following message in syslog:

```
Warning: wrong prefix 2001:660:4501:3201 \
  0:30:b6:51:d4:1c fe80::230:b6ff:fe51:d41c
```

Thanks to these tools, we tested all the attacks presented in section IV, except for the assignment of an Ethernet broadcast address. To test it, we used the following sequence on the attacker:

```
root@attacker:~/# ifconfig eth0 down
root@attacker:~/# ifconfig eth0 hw ether ff:ff:ff:ff:ff:ff
root@attacker:~/# ifconfig eth0 up
```

When the interface is set to up again, the attacker sends a Neighbor Advertisement with this broadcast address as source. It is detected by NDPMon:

```
Warning: ethernet broadcast ff:ff:ff:ff:ff:ff 0:0:0:0:0:0:0:0
```

The attacker is running a Linux 2.6.15 kernel, which allows to set such an address on an interface. The victim is running a newer version of the kernel (2.6.17), and thus of the IPv6 stack, which does not allow such an operation. As NDPMon detects such behavior, it makes possible to identify hosts running outdated IPv6 stacks, and which could be vulnerable to more attacks, and may require an update.

VI. Planned Extensions

The first prototype of NDPMon has been released in September 2006 and its development is pursued within the team. Some extensions are already planned. In IPv6, a node has exactly one link-local address, but can have more than one global address. At the moment, in NDPMon, for each pairing Ethernet and IPv6 address (link-local or global), one entry is present in the cache. The data structure will be updated to be more compliant to IPv6's standards. In the same way, the legitimate routing informations are defined in three different lists: a list of Ethernet addresses, a list of IPv6 addresses and a list of authorized IPv6 prefixes. These three lists will be combined in a single list of routers, with as attribute: the Ethernet address, the IPv6 link-local and eventually global addresses, and a list of prefixes advertised by the router. Thanks to this improvement, we will be able to make a more accurate description of the legitimate routing infrastructure on the link, and detect more attacks or misconfigurations.

Moreover, NDPMon includes a learning phase where the tool learns the network's default behavior. But this learning phase only fills the neighbor cache. We will extend it and also fill the routers' list automatically.

VII. Conclusion

In this paper, we showed that, if the Neighbor Discovery Protocol is an important building block in IPv6, it is also vulnerable to attacks or misconfigurations. Since approaches for securing this protocol, are very complex to deploy in real networks, we proposed a monitoring architecture based on what exists in the IPv4 world, while extending it.

We presented a working prototype developed in our team, NDPMon [7]. NDPMon is distributed under the LGPL license and available on SourceForge [8] project. This tool is easy to deploy and configure, and is able to detect many attacks against the Neighbor Discovery Protocol or IPv6 stacks vulnerabilities.

The development NDPMon will be continued within the team, and improvements will be brought to it. We are working on the distribution of the tool in the Debian [9] GNU/Linux distribution.

References

[1] D. Plummer, "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware," RFC 826 (Standard), Nov. 1982. [Online]. Available: http://www.ietf.org/rfc/rfc826.txt

[2] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460 (Draft Standard), Dec. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2460.txt

[3] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)," RFC 2461 (Draft Standard), Dec. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2461.txt

[4] A. Conta and S. Deering, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," RFC 2463 (Draft Standard), Dec. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2463.txt

[5] S. Deering, "ICMP Router Discovery Messages," RFC 1256 (Proposed Standard), Sept. 1991. [Online]. Available: http://www.ietf.org/rfc/rfc1256.txt

[6] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," RFC 2462 (Draft Standard), Dec. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2462.txt

[7] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," RFC 3775 (Proposed Standard), June 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3775.txt

[8] P. Nikander, J. Kempf, and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats," RFC 3756 (Informational), May 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3756.txt

[9] J. Arkko, J. Kempf, B. Zill, and P. Nikander, "SEcure Neighbor Discovery (SEND)," RFC 3971 (Proposed Standard), Mar. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc3971.txt

[10] T. Aura, "Cryptographically Generated Addresses (CGA)," RFC 3972 (Proposed Standard), Mar. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc3972.txt

[11] F. Bak, E. Lear, and R. Droms, "Procedures for Renumbering an IPv6 Network without a Flag Day," RFC 4192 (Informational), Sept. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4192.txt

[7] http://ndpmon.sourceforge.net

[8] http://www.sourceforge.net

[9] http://www.debian.org