A New Approach for Detection of Host Identity in IPv6 Networks

Libor Polčák, Martin Holkovič and Petr Matoušek
Faculty of Information Technology, Brno University of Technology, Božetěchova 2, 612 66 Brno, Czech Republic

Keywords: Computer Network Security, Host Identity, IPv6 Monitoring, SLAAC, Neighbor Discovery.

Abstract: For security, management and accounting, network administrators benefit from knowledge of IP and MAC address bindings. In IPv6, learning these bindings is not as straightforward as it is in IPv4. This paper presents a new approach to track IPv6 address assignments in LANs. The method is based on a study of implementation of IPv6 (mainly neighbor discovery) in current operating systems. The detection is passive for end devices and does not require any software or hardware modifications. In contrast with current methods, our approach does not poll routers in the network and works also in networks where IPv6 multicast is not broadcasted (active Multicast Listener Discovery snooping – MLD snooping). Moreover, our approach detects that an address is no longer used. The approach was successfully tested in a campus network.

1 INTRODUCTION

Users require reliable computer networks so that they can perform their daily work. In order to achieve such state, network operators have to monitor the managed network and its weak points, detect misuse of the network, backtrack security incidents, provide accountings for the offered services etc. The knowledge of the identity of computers and their users in the managed network is essential to achieve these tasks.

One possibility of user identity tracking is through authentication. For example, RADIUS authenticates a user and MAC address of his or her device connected to a network. In some networks, such as campus network at our university, users have to register their MAC address before they are allowed to access the Internet and services offered in the network. However, unlike network layer addresses, MAC addresses are not propagated outside of LANs. Hence, the knowledge of bindings between network layer addresses and MAC addresses is crucial for network management, security, and accounting.

In IPv4, a device usually leases an IPv4 address from a DHCP server in the custody of the network operator. Then, the device uses this unique IPv4 address for all its communication until the lease expires. As the DHCP server keeps logs of MAC and IPv4 address bindings, it is straightforward for network operators to obtain the identity of their users at any given time from DHCP and authentication logs.

The imminent exhaustion of IPv4 address space was acknowledged a long time ago. Today, two of the regional registrars (Huston, 2013) are already in the state in which the IPv4 addresses are allocated according to very strict policies. Even though the adoption of the IPv6 is still in an early phase, it is maturing (Dhamdhere et al., 2012). Hence, for security, management and accounting reasons, network administrators of IPv6-enabled networks needs to keep track of IPv6 addresses used in the managed network.

IPv6 guarantees at least 2^{64} addresses allocated to each LAN. There are several mechanisms that manages the allocation of addresses. For example, Stateless Address Autoconfiguration (SLAAC) (Thomson et al., 2007) allows an end device to generate as many IPv6 addresses as it needs as long as the addresses are not already used by another device in the network, e.g. for privacy concerns (Narten et al., 2007a; Groat et al., 2010). Note that the addresses are not handled centrally but generated by end devices. Moreover, the network operator is not able to influence the address generation process. In addition, there is not any node in the network that keeps track of IPv6 addresses being used by devices connected to the network. Even more, a host does not send any specific message when an address is no longer used by the host.

In this paper we propose a new approach for learning the MAC and IPv6 address pairing in LANs. Our approach is based on the study of implementation of IPv6 in current operating systems (OSes). Therefore, it provides a solution for user identification problem in IPv6 networks for network operators running a network that is directly accessed by users with devices that are not under direct control of the network op-

Polcák L., Holkovič M. and Matousek P..
A New Approach for Detection of Host Identity in IPv6 Networks.
DOI: 10.5220/0004608700570063
In *Proceedings of the 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems* (DCNET-2013), pages 57-63
ISBN: 978-989-8565-72-3

erators. These networks include campus-wide networks, networks of companies that allow their staff or customers to connect own devices to the network (”bring your own device” policy), Wi-Fi and Ethernet hot spots, and hotel networks.

The first contribution of this paper highlights the differences in behaviour of current OSes and their violations of RFCs concerning IPv6 address assignments. The main contribution is the new approach for user identification in IPv6 networks, which is based on monitoring control messages that are already present in the network. It does not influence or modify IPv6 in any way so it does not require any additional changes in the network hardware or software. Moreover, the privacy of the users in the network, with respect to the outside world, is not influenced by the proposed mechanism. The approach was successfully tested in a real network.

This paper is organized as follows. Section 2 overviews the address assignment mechanisms in IPv6. Section 3 summarizes the related work. The results of the study of behaviour of current OSes during IPv6 address assignments are presented in Section 4. Section 5 outlines the proposed approach for tracking user identity in LANs. Our experiments are summarised in Section 6. Section 7 lists open questions in our work and sketches our plans for future work. Section 8 concludes the paper.

2 PRELIMINARIES

This section reviews the basics of *Duplicate address detection* (DAD), a part of *Neighbor Discovery* (ND), and overviews common methods for IPv6 address assignments.

When a new IPv6 address is about to be used by a device, the device needs to test that the address is not already used in the network (Narten et al., 2007b; Thomson et al., 2007). Until the new address is proven to be unique, it is called *tentative*.

In order to prove that the tentative address is unique, the device has to send *Neighbor Solicitation* (NS) request to the solicited-node multicast group (Hinden and Deering, 2006) whose address is derived from the tentative address. In this paper, NS requests issued during DAD are denoted as *DAD-NS*. If the tentative address is already used by another device, the other device should reply with a *Neighbor Advertisement* (NA) to the multicast group for all nodes in the network (ff02::1). Only if no NA is received, the new address can be used. To avoid race conditions in address assignments, RFC 4862 orders (Thomson et al., 2007) that each host has to join the solicited-node multicast group before it sends the DAD-NS.

Consequently, there is no central point in the network that gathers all active addresses; the knowledge is spread over the network and is available through the solicited-node multicast groups.

SLAAC (Thomson et al., 2007) is a basic method for address assignments in IPv6. In contrast to DHCP, dominant in IPv4, SLAAC is not based on leases. Instead, a device itself generates its addresses. First, the device learns the network (higher) part of IPv6 address from a *Router Advertisement* (RA), a message periodically send by gateways in the network. Then, the device generates the lower part of the IPv6 address called *interface identifier* (IID) (Hinden and Deering, 2006). The original method for selecting an IID uses modified EUI-64 IID. Later, privacy extensions (Narten et al., 2007a) introduced completely random IIDs which may change during time.

Although there is a variant of DHCP called stateful DHCPv6 (Droms et al., 2003), it does not provide the same information as DHCP since DHCPv6 assigns IPv6 address according to DHCP Unique Identifier (DUID). DUID is generated by each host, mostly during OS installation. As a consequence, DUID is changed when a host is rebooted to another OS. Therefore, the MAC and IP address pairings are not stored in DHCPv6 logs. On the other hand, the assigned address has to be confirmed by DAD.

Finally, it is possible to assign a static address. Whenever a new static IPv6 address is entered on a host, it has to be validated by DAD.

As mentioned above, ND is a part of each mechanism for address assignments. Hence, we choose ND as a basis for our network monitoring approach. However, as discussed in Section 4, ND is not implemented in the same way among current OSes.

3 RELATED WORK

Several attempts have been made to study IPv6 IIDs. (Groat et al., 2010) proposed to use *ping* or *traceroute* to monitor a location of a node that uses a static interface identifier of any sort. (Dunlop et al., 2011) list an example of Windows using a random, yet static networks IIDs. However, both papers aimed at global tracking of a movement of a specific user. In contrast, we want to monitor only the local network. In addition, both Groat et al. and Dunlop et al. need to know the address in advance. Our research is concerned with unknown IPv6 addresses. Another difference is that we want to learn all addresses that belong to every device connected to a network.

Similarly to our goal, (Grégr et al., 2011) are

Table 1: OSes tested for compliance with RFCs specifying ND.

Windows	XP SP3, Vista, Vista SP3, Server 2008 R2, 7, 7 SP1, and 8
Linux	various distributions including Debian, CentOS, Debian, and Ubuntu (kernels 2.4.27–3.2)
Mac OS X	10.6.2 (kernel 10.2)
Unix	FreeBSD 9.0, OpenBSD 5.0, and Solaris 5.11

also interested in learning the IPv6 addresses that were used by a host with a specific MAC address. They presented a campus network monitoring system which gathers data from the *neighbour cache* (NC) of the routers in the network using SNMP. However, two conflicting requirements needs to be balanced. In order to have sound information about the IPv6 addresses in the network, they need to poll routers sufficiently often. Since routers are critical devices and the polling results in additional workload, the polling cannot be too frequent. As a consequence, a new address selected by a device in the network is learned with a delay. During a security incident, an attacker can use an address for a limited time. Consequently, the monitoring system can miss that the IPv6 address was used.

(Groat et al., 2011) studied the possibility of using DHCPv6 for monitoring the identity of users in the network. Our approach is more general as it is not restricted to DHCPv6.

A tool called *addrwatch* (Kriukas, 2012) can monitor ND messages in the network. However, we identified several weaknesses of the tool. Firstly, *addrwatch* just reports ND messages. They are not put in any context. Secondly, *addrwatch* completely ignores multicast messages, therefore, it cannot detect address assignments in a network with *MLD snooping* (multicast is not broadcasted). Finally, *addrwatch* does not detect that an address is no longer used.

(Asati and Wing, 2012) deal with the same problem as we do. However, their solution involves changes in router behaviour. Our approach does not need any change on any critical network device.

Although we have not found a reference, we expect that some administrators use port mirroring and parse the mirrored traffic with a sniffer to learn the MAC and IP address bindings. Our approach does not depend on processing all network traffic and consequently it is more energy efficient. In addition, the bandwidth of the mirroring port could be insufficient for all traffic traversing the mirroring switch or even one full-duplex port. Moreover, in case of only mirroring the port connected to the router in the network, the learned MAC and IP address pairings are more or less the same to these stored in the NC of the router as discussed Section 6.

4 OPERATING SYSTEMS STUDY

This Section describes implementation of ND in current OSes. We studied (Polčák and Holkovič, 2013) the exact sequences of messages that are issued during DAD after an address is assigned or automatically generated. The main goal was to validate that OSes follow the sequence ordered by RFC 4862 (Thomson et al., 2007). However, the results are not positive and some OSes diverge.

Firstly, we selected OSes (see Table 1) that we believe are the most common in current LANs. Then, we connected hosts running these OSes into our laboratory network and captured all packets that were sent or received by each host. Performed tests included 1) SLAAC with both enabled and disabled RFC 4941 privacy extensions addresses, 2) DHCPv6, 3) static addresses, and 4) duplicate addresses in the network. We have revealed following anomalies:

1. Windows Vista and later (for all IPv6 addresses) and FreeBSD (for static and EUI-64 IPv6 addresses) ignore DAD-NS if the other host has the same MAC address.
2. Windows Vista and later use a tentative link local address to join multicast groups before they start DAD for the address.
3. Solaris (for all addresses), FreeBSD (for static addresses), and Windows Server 2008 R2, Windows 7 and later (for link local addresses) diverge from the recommended sequence of actions during DAD because they send DAD-NS before they join the solicited-node multicast group derived from the tentative address.
4. Windows 8, Solaris, and Mac OS X send NAs during DAD that are not mandatory.
5. OpenBSD does not join solicited-node multicast groups at all.

In general, if a router in the network behaves as an MLD querier, it periodically queries multicast groups to verify that some hosts are still part of each multicast group. Hosts that are part of a multicast group randomly generate a timeout during which they wait for reply of another host in the group. If there is no such acknowledgement before the timeout passes, the host

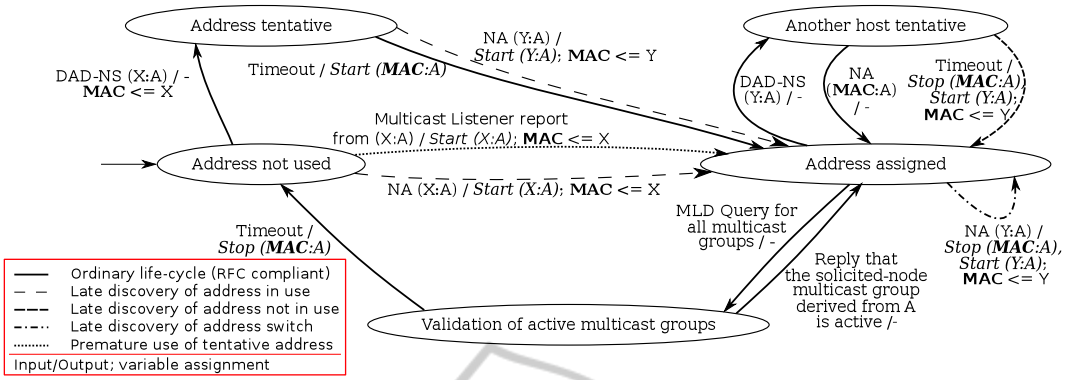

Figure 1: The life-cycle of an IPv6 address A can be monitored by the Mealy FSM extended by the variable **MAC** to store the MAC address of the interface that currently uses A. Binding between MAC address X and A is denoted as (X:A).

replies that the multicast group is still active. Hence, if no address corresponding to the queried solicited-node multicast address is used, there is no reply in the multicast group. However, if more than one IPv6 address in the network coincide into one solicited-node multicast group, only one of the hosts replies.

All selected OSes that register to multicast groups reply to MLD queries. However, we discovered that some OSes do not meet the maximal timeout specified in an RA. Although the RA announced maximal timeout of 0.1 s, some acknowledgements were received after 0.7–0.8 s.

5 APPROACH FOR ADDRESS ASSIGNMENT DETECTION

The approach for address assignment detection was created according to the study of the behaviour of current operating systems presented in Section 4. The life-cycle of an IPv6 address A can be tracked by the extended Mealy FSM depicted in Figure 1. Network control traffic is used as the input of the FSM, the output is a signalisation that the address started or ceased to be used. The FSM is extended with a variable to store the MAC address of the interface that uses the tracked IPv6 address.

The FSM tracks ND messages to detect address assignments. MLD queries and responses are used to detect that the address is no longer used. Hence, we recommend to enable MLD querying on a router in the network. In addition, if MLD snooping is active in the network (multicast traffic is not broadcasted), to capture DAD-NSes and NA replies, we recommend to 1) join the multicast group for *all nodes* (ff02::1) and *all MLDv2-capable routers* (ff02::16), 2) detect all requests to join solicited-node multicast groups, and 3) join the detected groups.

The initial state of the FSM is the *Address not used*. When a host generates the address A, it issues DAD-NS, and the FSM shifts to *Address tentative*.

If everything worked according to RFCs, the only trigger for transition from *Address tentative* would be the timeout as the address would not be used in the network. However, Solaris, some versions of Windows, and FreeBSD do not join the solicited-node multicast groups before issuing the DAD-NS. As a consequence, in networks with MLD snooping, some address assignments may have been unnoticed earlier and the address can already be used. Therefore, it is possible to receive an NA from another host. In both cases the FSM detects the MAC address that is bind to the IPv6 address and shifts to *Address assigned*.

Similarly to the NA received after DAD-NS, the FSM may detect an NA for the address A in the initial state (e.g. non-mandatory NA during DAD). Consequently, the FSM shifts directly to *Address assigned*. Additionally, the FSM shifts between these two states in case of Windows using a tentative link local address as the source address to join multicast groups.

In order to detect that the address was dropped by the host, *Validation of active multicast groups* is reached after an MLD query is received. In case that the solicited-node multicast group derived from address A is acknowledged, the address is most likely being used as the solicited-node multicast groups were designed so that two hosts are not likely to be in one solicited-node multicast group. If the MLD query expires, the address is definitely not used any more, and the FSM returns to the initial state.

While the FSM is in *Address assigned*, another host might try to use the address. When DAD-NS is received, the FSM shifts to *Another host tentative*. Ordinarily, the address is still in use and the NA follows. If the address was no longer used but it had not been detected (e.g. because the MLD query was not issued, yet), the NA would not be sent. In such case,

the FSM also shifts back to *Address assigned*, however, the detected MAC address changes. In a rare occasion when an NA from another MAC address is seen in *Address assigned*, the FSM loops in this state and the MAC addresses are swapped. This loop is present in the FSM only for safety reason as the study of OS behaviour does not suggest that it is needed.

6 EXPERIMENTS

We implemented *ndtrack* (Holkovič and Polčák, 2013), a tool which follows the approach for IPv6 address assignments that is sketched in Section 5. This Section describes the experiments with *ndtrack*. The first experiment was aimed at real network monitoring. As most of the devices were not under our control, we could not confirm that *ndtrack* detected all devices. However, we checked that all our devices were detected. The other three experiments were performed in a laboratory network; they focused on the quality of the monitoring approach.

6.1 Real Network Deployment

The first experiment aimed at long-term monitoring (almost a month) of SLAAC in a network with MLD querying enabled. The network spans two buildings and is available for all employees of the faculty (see Figure 2). We successfully tested that *ndtrack* detected IPv6 addresses that are being used in the network and that the addresses are correctly identified as no longer assigned after the hosts disconnect or stop using them (see Figure 3 for the statistics).

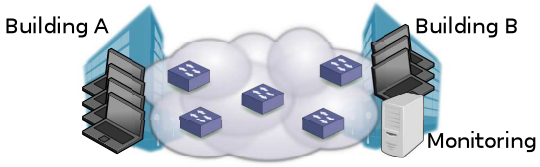

Figure 2: The network topology of the real network monitoring. Some of the computers were not under our control.

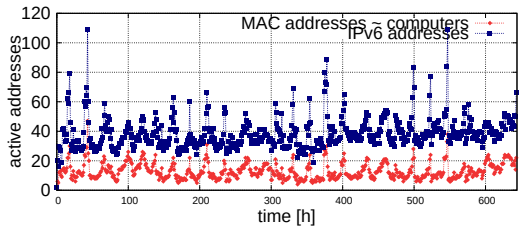

Figure 3: Statistics of the real network monitoring. IPv6 addresses were successfully detected to be used (the number of known addresses rises during working hours) and dropped (the number of IPv6 addresses lowers at night).

During the experiment, we verified that the addresses used by our devices are correctly recognized together with other devices that were not under our control (see Table 2). In order to make the experiment more convincing, we connected a device to the network in a different building than the one in which our monitoring station was located. All addresses assigned to the device were correctly identified and later dropped when we disconnected the device.

Table 2: A snippet of a table created by *ndtrack*.

IP addr	time	MAC addr
fe80:*:777	14/2 11:07–14:40	1c:*:77
2001:*:777	14/2 11:07–14:40	1c:*:77
2001:*:335f	14/2 11:07–14:40	1c:*:77
2001:*:aa63	12/2 14:21–15/2 0:58	00:*:84

6.2 Network with MLD Snooping

We verified the behaviour of *ndtrack* in the presence of MLD snooping in our laboratory. A monitoring station running *ndtrack* and a testing computer were connected to a switch with MLD snooping enabled as depicted in Figure 4. In the first set of experiments, *ndtrack* did not follow the advice given in Section 5 and did not join the appropriate multicast groups. In the second set of experiments *ndtrack* joined the multicast groups as recommended in Section 5. Several OSes were tested on the testing computer during each set of experiments.

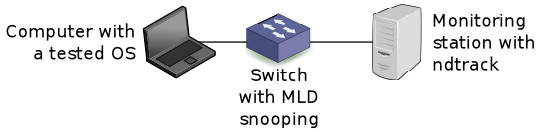

Figure 4: Network topology for experoments with network with active MLD snooping.

The results of the experiment are summarised in Table 3. When *ndtrack* joined the solicited-node multicast groups, it successfully detected all OSes but OpenBSD and static addresses in FreeBSD due to the following reasons:

- OpenBSD does not join the solicited-node multicast groups derived from the tentative address. As a consequence, *ndtrack* did not know that it should join the solicited-node multicast group. As a result, the switch with MLD snooping activated did not propagate the DAD-NS to the monitoring station.
- As already stated in Section 4, FreeBSD sends DAD-NS before it joins the solicited-node multicast groups derived from the tentative address.

Therefore, *ndtrack* joined the solicited-node multicast groups derived from the tentative address after the DAD-NS was sent and consequently did not learn about the address assignment. Windows and Solaris that also join the solicited-mode multicast group late (see Section 4) send additional NAs during DAD and therefore were detected by *ndtrack*.

Table 3: Effectivity of our approach in networks with active MLD snooping — without/with joining the solicited-node multicast group (✓= Detected).

OS	static addr	SLAAC
Windows 7 and earlier	-/✓	-/✓
Windows 8	✓/✓	✓/✓
Linux	-/✓	-/✓
Mac OS X	✓/✓	✓/✓
FreeBSD	-/-	-/✓
OpenBSD	-/-	-/-
Solaris	✓/✓	✓/✓

6.3 Network with Stateful DHCPv6

The next experiment tested stateful DHCPv6. We connected computers running Windows 7, 8, 2008 R2, Ubuntu 12.10, and Solaris (one computer for each OS) to a laboratory network depicted in Figure 5. We verified that *ndtrack* detected all address assignments. Hence, DHCPv6 leases are detected by the proposed approach.

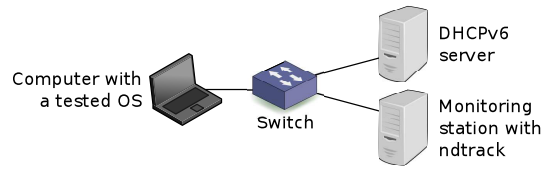

Figure 5: Network topology for experoments with network with active MLD snooping.

6.4 Comparison to Other Methods

In the last experiment, we compared *ndtrack* to the *neighbour cache polling* (NCP) and *addrwatch* (both described in Section 3), in the network with topology depicted in Figure 6. We ran *ndtrack* and *addrwatch* on the monitoring station in three runs: *ndtrack* was tested with MLD snooping enabled on the switch whereas *addrwatch* was tested with MLD snooping both enabled and disabled. Each experiment followed this scenario:

1. Two Linux hosts were connected to the network.
2. Host A opened a connection to host B and the hosts transferred a file in this connection.
3. Host A initiated an one-way UDP connection outside the network.
4. Host A opened a TCP session to the remote host.
5. Hosts A and B were disconnected.

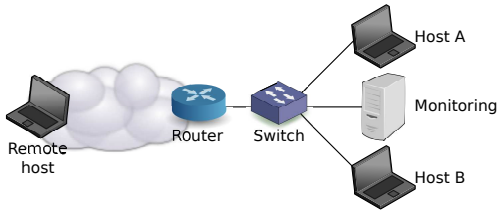

Figure 6: Network topology used to compare our approach with older methods.

During the experiment, we monitored NC of the router, and the outputs of the monitoring tools. The results are summarized in Table 4.

Table 4: Comparison of our approach with older methods (✓= detected).

	NCP	addrwatch		*ndtrack*
MLD snooping	Y/N	No	Yes	Yes
A, B connected	-	✓	-	✓
Local TCP	-	✓	-	✓
One-way UDP	-	✓	-	✓
Remote TCP	✓	✓	-	✓
A, B disconn.	-	-	-	✓

An entry for the monitored hosts appeared in the NC of the router only after a packet destined to the host arrived from the Internet. Data transfers inside the LAN and the outgoing UDP session were undetected. In addition, the record stayed in the NC (*stale* state) after the host was disconnected. Port mirroring and analysis of the traffic traversing the router would detect the outgoing UDP stream. However, local communication would be unnoticed.

While MLD snooping was disabled, *addrwatch* detected the DAD-NSes issued by the hosts when they connected to the network. Additionally, *addrwatch* reported NS messages, issued by the hosts or the router, during the data transfers. However, active MLD snooping did not leak any ND message to the monitoring computer and consequently caused that *addrwatch* did not report any activity in the network. Moreover, *addrwatch* did not report that the hosts disconnected from the network even without MLD snooping as no ND message was sent to the network.

Both hosts were successfully identified by *ndtrack* although the tool was behind MLD snooping. In addition, *ndtrack* was able to detect that the addresses were no longer assigned.

7 FUTURE WORK

During future work, we want to focus on the open questions identified during the study of the behaviour of the OSes and the design of the FSM. Specifically, we plan to study the loop in the *Address assigned* state of the FSM. In addition, we plan to evaluate more devices, such us phones and tablets, to validate that the proposed FSM does not miss any transition.

Furthermore, we want to focus on networks without an MLD querier and compare the quality of the monitoring with respect to its location in a network. The proposed FSM does not detect that an address is not used any more without an MLD querier. We plan to implement a timeout that would shift the FSM from *Address assigned* to *Address not used* when no NA is seen for a suitable amount of time. However, the suitable timeout value is a subject for future research. One of the prerequisites is a study of NA in the network. Another approach for dealing with networks without an MLD querier is to use additional information from the default gateway NC (Grégr et al., 2011).

8 CONCLUSIONS

The advent of IPv6 protocol unveils a need to redesign mechanisms for user identification in LANs. Whereas in IPv4, network administrators can extract MAC and IPv4 pairings from DHCP logs, in IPv6, the pairing of IPv6 and MAC addresses is not available on a single device in the network. We studied behaviour of implementation of ND in current OSes (Polčák and Holkovič, 2013). Based on this study, we proposed a mechanism that deals with the problem of the identification of MAC and IPv6 address pairings in networks with MLD snooping both active and inactive. The approach detects all address assignments in networks without MLD snooping. When MLD snooping is active, the approach deals with all addresses and OSes but OpenBSD and static addresses in FreeBSD.

Our approach differs to current methods in several aspects. Firstly, it is completely passive for end devices in the network. In addition, the approach does not need any modification of software or hardware used in the network. Moreover, for most OSes, the proposed approach detects that a new address was generated immediately without polling of active devices in the network. Furthermore, the described approach detects that an address is no longer used. Even more, the approach works for all common methods for IPv6 address distribution, namely SLAAC, stateful DHCPv6, and static assignments.

ACKNOWLEDGEMENTS

This work is a part of the project VG20102015022 supported by Ministry of the Interior of the Czech Republic. This work was also supported by the research plan MSM0021630528 and BUT project FIT-S-11-1.

REFERENCES

Asati, R. and Wing, D. (2012). Tracking of Static/Autoconfigured IPv6 addresses. Internet Draft, version 00 (Work in progress).

Dhamdhere, A., Luckie, M., Huffaker, B., claffy, k., Elmokashfi, A., and Aben, E. (2012). Measuring the deployment of ipv6: topology, routing and performance. In *Proc. of IMC '12*, pages 537–550, New York, NY, USA. ACM.

Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and Carney, M. (2003). Dynamic Host Configuration Protocol for IPv6 (DHCPv6). RFC 3315.

Dunlop, M., Groat, S., Marchany, R., and Tront, J. (2011). The good, the bad, the ipv6. In *CNSR 2011*, pages 77–84, Ottawa, Canada.

Groat, S., Dunlop, M., Marchany, R., and Tront, J. (2010). The privacy implications of stateless ipv6 addressing. In *Proc. of CSIIRW '10*, pages 52:1–52:4, New York, NY, USA. ACM.

Groat, S., Dunlop, M., Marchany, R., and Tront, J. (2011). What dhcpv6 says about you. In *WorldCIS 2011*, pages 146–151, London, UK.

Grégr, M., Matoušek, P., Podermański, T., and Švéda, M. (2011). Practical ipv6 monitoring - challenges and techniques. In *Proc. of IM 2011*, pages 660–663, Dublin, Ireland. IEEE CS.

Hinden, R. and Deering, S. (2006). IP Version 6 Addressing Architecture. RFC 4291.

Holkovič, M. and Polčák, L. (2013). ndtrack. http://www.fit.vutbr.cz/ ipolcak/prods.php?id=308.

Huston, G. (2009–2013). IPv4 Address Report. http://www.potaroo.net/tools/ipv4/index.html.

Kriukas, J. (2012). addrwatch: A tool similar to arpwatch for ipv4/ipv6 and ethernet address pairing monitoring. https://github.com/fln/addrwatch.

Narten, T., Draves, R., and Krishnan, S. (2007a). Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 4941.

Narten, T., Nordmark, E., Simpson, W., and Soliman, H. (2007b). Neighbor Discovery for IP version 6 (IPv6). RFC 4861.

Polčák, L. and Holkovič, M. (2013). Behaviour of various operating systems during SLAAC, DAD, and ND. http://6lab.cz/?p=1691.

Thomson, S., Narten, T., and Jinmei, T. (2007). IPv6 Stateless Address Autoconfiguration. RFC 4862.