This document is published in:

IEEE Communications Letters, vol. 16, n. 11, November 201, pp. 1900-1902, doi 10.1109/LCOMM.2012.100812.121517.

An Integrated Approach to Prevent Address Spoofing in IPv6 Links

Alberto García-Martínez and Marcelo Bagnulo

***Abstract*—We propose an integrated approach to protect from address spoofing for both IPv6 and Layer-2 addresses, and from address resolution attacks. The proposed approach is an extension to the FCFS SAVI specification, and relies on the inspection and generation of standard Neighbor Solicitation messages. It does not require host modification and manual configuration is only needed to indicate the ports to which routers connect.**

***Index Terms*—Network-layer security, link-layer security, address spoofing, address configuration, IPv6.**

I. Introduction

CRITICAL security threats arise from the ability of malicious nodes to use layer-2 identities (e.g., MAC addresses in Ethernet) or layer-3 identities (IP addresses) belonging to other nodes of the link. In the particular case of Ethernet and IPv6, the major threats resulting from layer-2 and layer-3 identity manipulation are the following:

`T1.` A malicious node M sends packets with IP_T as IP source address. Victim V receives these packets and assumes incorrectly that the packets have been generated by the rightful owner of IP_T.

`T2.` M uses the address resolution protocol of IPv6, Neighbor Discovery (ND, [1]), to make V believe that M's MAC corresponds to IP_T. When V sends traffic to IP_T, it is actually sending it to M. When combined with threat `T1`, this is a masquerade attack in which M appears to V as node T.

`T3.` A variation of `T2` is to use ND to make node V believe that other MAC than M and T is the MAC address of node T. This prevents communication between V and T, so it is a Denial of Service (DoS) attack.

`T4.` M initiates a communication with T with its own address, and then uses ND to associate in T the MAC address of a victim V to IP_M. Then the traffic sent by T to IP_M is deflected to V. If the traffic generated is high, this is a flooding attack.

`T5.` M prevents other nodes from configuring their IPv6 addresses by responding to Duplicate Address Detection (DAD, [2]) requests. This is a DoS attack.

`T6.` M sends frames with MAC_V source address, being MAC_V the MAC address of node V. 802.1D bridges use this frame to update their forwarding information for MAC_V. Future frames destined to MAC_V received by any of these bridges are sent to M. This is a MAC spoofing attack.

The basic approach to secure link communications is to configure layer-3 aware bridges with the layer-2 and layer-3 addresses associated to the nodes connecting to each physical port. ND inspection and filtering according to this configuration protect from the threats related with address resolution, and also protect against attacks to address configuration. While this approach solves all the security issues mentioned before, it results in costly per-device configuration, since it requires access to the end nodes to obtain their MAC address and configure/obtain their IP address, it requires per-port configuration in the switches, and it complicates the support for nodes that change their attachment port.

A more flexible approach results from the use of IEEE 802.1X [3]. IEEE 802.1X allows a node to exchange authentication information with the bridge to which it attaches. The authentication information can be in user/password form, certificate, etc. This information is used to coarsely authorize the node to communicate, so it is not designed to authorize or negotiate the layer-2 or layer-3 addresses to be used by the node. However, the bridge can obtain address authorization from this authentication information either according to a mapping locally stored in the bridge, or by using RADIUS or DIAMETER to access to an AAA server. When an AAA server is used, the bridges can update the filtering information as a node changes its attachment point to the link. Note that MAC spoofing can only be prevented if the link administrator knows in advance the MAC address of each node and configures the AAA server accordingly.

SAVI is a new standard that provides link-scope layer-3 antispoofing protection. A SAVI device, typically a bridge, associates IP addresses to physical ports and filters out any packet with a source IP address that does not correspond to an existing binding. To create this binding, SAVI devices inspect the IP address configuration messages exchanged by the nodes. Different solutions of SAVI for DHCP [4], IPv6 locally configured addresses (FCFS, First-Come First-Served SAVI [5]) and IPv6 nodes using SEND [6], have been defined. The configuration required is limited to the specification of the ports through which routers are connected. In addition, SAVI solutions support layer-2 node mobility. However, SAVI protection is restricted to threats `T1` and `T5`, providing no security for ND manipulation (unless the SEND security extension is used, which requires specific support in the hosts) and MAC address spoofing.

In this letter we extend FCFS SAVI to provide protection

A. García-Martínez and M. Bagnulo are with Departamento Ingeniería Telemática, Universidad Carlos III de Madrid, Av. de la Universidad 30, Leganes, Spain. Email: alberto@it.uc3m.es, marcelo@it.uc3m.es.

The work of Alberto García-Martínez is supported by the CCG10-UC3M-TIC-5624 project funded by Comunidad de Madrid. The work of Marcelo Bagnulo has been partially funded by the Ministry of Science and Innovation of Spain under the QUARTET project (TIN2009-13992-C02-01).

Manuscript received July 9, 2012; revised September 19, 2012.

to all the threats described without requiring any per host configuration, greatly reducing the costs of securing the link compared with existing solutions.

II. FCFS SAVI

FCFS SAVI [5] defines a mechanism to prevent the use of IPv6 addresses by unauthorized hosts. It does so by creating bindings between IPv6 addresses and the bridge port that the rightful owner of the IPv6 address is attached to. The mechanism is based on information obtained from the DAD process [2]. DAD operation proceeds as follows: every IPv6 node configuring an address is required to issue a Neighbor Solicitation (hereafter DAD_NS) message with the Solicited Node multicast destination address, to which any other node with the same last 24 bits in its address must be associated. If a node with the same IP address exists, it receives the DAD_NS message and responds with a Neighbor Advertisement (DAD_NA) message, so that the node initiating the DAD procedure must stop configuring the address. If no response is obtained in a short period of time, the address is deemed available and the node configures it.

FCFS SAVI main behavior is as follows: when a SAVI device B receives a DAD_NS message for IPv6 address IP_A from a validating port P, i.e., a port which does not connect to another bridge, it forwards the DAD_NS message to any port of B with a pre-existing binding to IP_A (if any), and to the rest of the bridges. A SAVI binding for IP_A is created in tentative state. Other bridges receiving the DAD_NS do the same, so the message is propagated through the spanning tree, and forwarded to any port for which a binding for address IP_A exists (if any). If there is no host in the link with address IP_A configured and an existing SAVI binding in its closest SAVI bridge, then no response is received at bridge B, a timer at B expires and the binding between IP_A and port P is set to valid. Otherwise, the host configured with address IP_A responds with a DAD_NA message. B receives the DAD_NA message, discards the SAVI binding and forwards the message through P to prevent the node initiating the procedure from configuring address IP_A.

III. FCFS SAVI WITH LAYER-2 EXTENSIONS

The inspection by SAVI devices of the DAD message exchange is a valid foundation for assuring that only one node is allowed to use an IP address at a given time, in particular the first node which tried to configure the IP address when it was unused. This is a 'First-Come, First-Served' behavior, as stated in the name of the mechanism.

We propose extending FCFS SAVI to prevent MAC address duplication as follows: FCFS SAVI bridges are modified to include the MAC address in the data structure storing SAVI information, the SAVI binding database. Therefore, an IP/MAC address pair is associated to a physical port. Only packets coming from a physical port with both the IP and the MAC address included in an existing binding are forwarded.

When a node N starts configuring its IP address, it sends a DAD_NS for IP_N, using MAC_N as MAC source address. The SAVI bridge B1 receives the message through $PORT_P$ and creates an entry for $[IP_N, MAC_N, PORT_P]$ in tentative state. Then, the following occurs:

- If B1 already has an entry associated to a different port $PORT_Q$ containing IP_N, i.e., $[IP_N, MAC_N, PORT_Q]$, it forwards the DAD_NS through $PORT_Q$, as specified by FCFS SAVI. If the node at this port has IP_N configured, it responds with a DAD_NA. Then, B1 removes the $[IP_N, MAC_N, PORT_P]$ binding and forwards the DAD_NA to node N, aborting IP address configuration at node N.
- Else, if B1 has an entry $[IP_B, MAC_N, PORT_Q]$ (note that IP_B is different than IP_N because otherwise we would be in the previous case), the switch B1 issues a NS to resolve the IP address for IP_B through $PORT_Q$. This is to check that the node with MAC_N is still at its previous location, $PORT_Q$. If B1 receives a NA responding to the NS, including MAC_N as the link-layer address, B1 removes the binding in tentative state for $[IP_N, MAC_N, PORT_P]$. B1 also generates a DAD_NA, and sends it to node N to prevent the configuration of address IP_N. Then, node N is not allowed to communicate using neither IP_N nor MAC_N.

As occurs with FCFS SAVI, to check for nodes connected in other bridges having either IP_N or MAC_N, B1 propagates the DAD_NS message to other bridges. These bridges do the same as if they had received the packet through a validating port, propagating DAD_NS and/or NS. According to the responses, they propagate to B1 any DAD_NA received from the nodes or generate it when NA messages are received.

B1 configures the binding if a timer expires and it has not received a response indicating that either the MAC or the IP address is configured in other port.

Probing the port for which an IP or a MAC was configured provides support for node mobility: if a node changes its attachment point, the DAD_NS (or NS) is propagated to the port to which the node attached before, but no response is generated, so the node is allowed to communicate in its new location.

To protect from address resolution attacks, each SAVI device checks that NS and NA messages coming from validating ports and containing link-layer addresses are consistent with the binding created for the port. Any ND message not fulfilling this condition is discarded.

A SAVI bridge may receive a data packet from a validating port for which a DAD_NS has not been received, because the DAD_NS was lost or the node changed its attachment point without issuing a DAD_NS message. In this case, the bridge itself creates a DAD_NS message and continues operation as described above, in a similar way to the operation described in the FCFS SAVI standard.

To illustrate the protection provided, consider a node M attached to bridge B2 which aims to impersonate node N's MAC_N (figure 1). To do so, M sends a data packet with MAC_N, but with IP_M, an IP address for which a valid binding exists in B2 for the port through which it is recived, port 3. Then, B2 creates a binding for $[IP_M, MAC_N, port\ \#3]$ in tentative state, and generates a DAD_NS message requesting for nodes having either IP_M or MAC_N configured. This message is propagated to B1. B1 determines that IP_M is not included in any of its bindings, but MAC_N is. Then, B1 generates a NS message for IP_N to request the MAC address associated to it, that is sent through the port for which a binding to $[IP_M, MAC_N]$ existed. Node N responds with a NA message including MAC_N. Then B1 generates a DAD_NA message for IP_M which is propagated through the bridges without validation, since this domain is considered the trusted infrastructure. B2 interprets this message as an indication to remove the binding for $[IP_M, MAC_N, port\ \#3]$. In this case B2 does not propagate the DAD_NA message to node M, since M sent a data packet instead of a DAD_NS message.

IV. Conclusion

FCFS SAVI with layer-2 extensions provides protection for IP and MAC spoofing attacks, as well as for address resolution attacks. In particular, it prevents all listed (`T1` to `T6`) attacks. The mechanism does not require any host modification, so it can be easily deployed. In addition, it does not require any kind of per-node configuration.

The mechanism can be successfully combined with DHCP SAVI. When DHCP SAVI for IPv6 is used, nodes still use DAD_NS to configure their address. DAD_NS messages for addresses allowed by DHCP are used to check that there are no duplicated MACs. Communication is only allowed if the MAC used is unique. In the IPv4 case, SAVI switches generate ARP requests to perform this test.

Acknowledgment

The authors would like to thank the anonymous reviewers of IEEE Communications Letters for their helpful and constructive comments.

References

[1] T. Narten, E. Nordmark, W. Simpson and H. Soliman. *Neighbor Discovery for IP version 6 (IPv6)*, RFC 4861, Sep. 2007.

[2] S. Thomson, T. Narten and T. Jinmei, *IPv6 Stateless Address Autoconfiguration*, RFC 4862, Sep. 2007.

[3] IEEE. *IEEE 802.1X-2010 Port-Based Network Access Control*, Feb. 2010.

[4] J. Bi, J. Wu, G. Yao and F. Baker. *SAVI Solution for DHCP*, draft-ietf-savi-dhcp-15.txt, *Work in progress*, Sep. 2012.

[5] E. Nordmark, M. Bagnulo and E. Levy-Abegnoli. *FCFS SAVI: First-Come First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses*, RFC 6620, May 2012

[6] M. Bagnulo and A. García-Martínez. *SEND-based Source-Address Validation Implementation*, draft-ietf-savi-send-08, *Work in progress*, Sep. 2012.

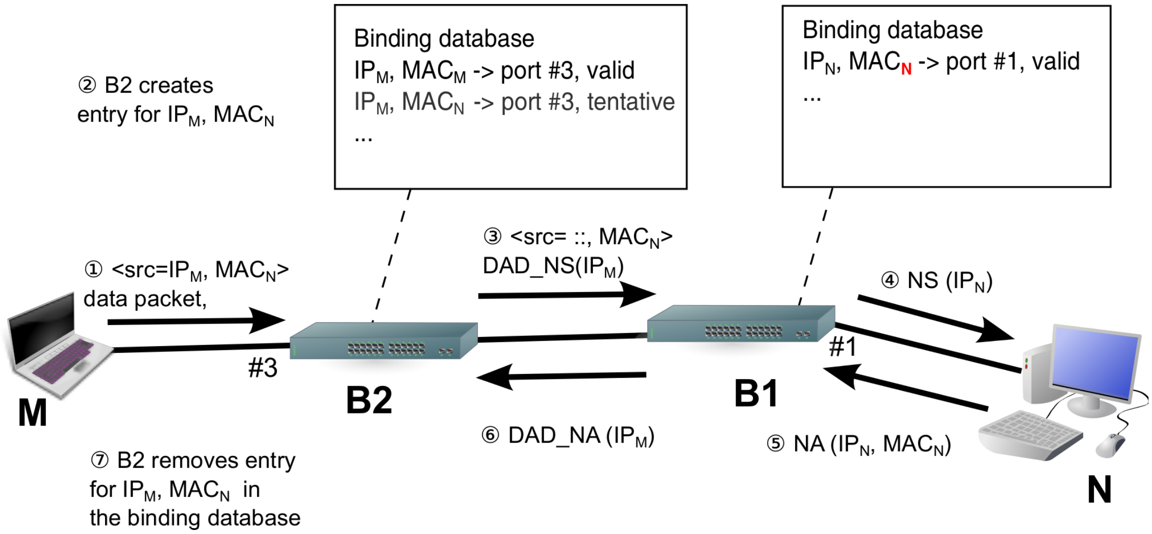
Binding database
IP_M, MAC_M -> port #3, valid
IP_M, MAC_N -> port #3, tentative
...
Binding database
IP_N, MAC_N -> port #1, valid
...
② B2 creates entry for IP_M, MAC_N
① <src=IP_M, MAC_N> data packet,
③ <src= ::, MAC_N> DAD_NS(IP_M)
④ NS (IP_N)
#3
#1
M
B2
B1
N
⑥ DAD_NA (IP_M)
⑤ NA (IP_N, MAC_N)
⑦ B2 removes entry for IP_M, MAC_N in the binding database

Fig. 1. Example of the protection provided by the extensions to FCFS SAVI