Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 26, 2009

E. Nordmark
Sun
M. Bagnulo
UC3M
January 22, 2009

First-Come First-Serve Source-Address Validation Implementation
draft-bagnulo-savi-fcfs-01

Status of this Memo

Copyright Notice

Abstract

This memo describes FCFS SAVI a mechanism to provide source address

validation for IPv4 and IPv6 networks using the First-Come First-Serve approach. The proposed mechanism is intended to complement ingress filtering techniques to provide a higher granularity on the control of the source addresses used.

Table of Contents

1. Introduction 3
2. Design considerations 3
2.1. Scope of FCFS SAVI 3
2.2. Constraints for FCFS SAVI 3
2.3. Address ownership proof 4
2.4. Special cases 5
3. FCFS SAVI specification 5
3.1. FCFS SAVI Data structures 5
3.2. FCFS SAVI algorithm 6
3.2.1. Processing of data packets 6
3.2.2. Processing of control packets 7
3.3. IPv4 Neighbor Unreachability Detection Procedure 9
3.3.1. ARP-based Neighbor Unreachability Detection procedure 9
3.3.2. ICMP-based Neighbor Unreachability Detection procedure 10
4. Security Considerations 11
5. Acknowledgments 13
6. Normative References 13
Authors' Addresses 13

1. Introduction

This memo describes FCFS SAVI, a mechanism to provide source address validation for IPv4 and IPv6 networks using the First-Come First-Serve approach. The proposed mechanism is intended to complement ingress filtering techniques to provide a higher granularity on the control of the source addresses used.

2. Design considerations

2.1. Scope of FCFS SAVI

The application scenario for FCFS SAVI is limited to the local-link. This means that the goal of FCFS SAVI is verify that the source address of the packets generated by the hosts attached to the local link have not been spoofed. FCFS SAVI can be used in IPv4 and in IPv6 networks.

In any link there usually are hosts and routers attached. Hosts generate packets with their own address as the source address. This is the so-called local traffic. while routers send packets containing a source address other than their own, since they are forwarding packets generated by other hosts (usually located in a different link). This what the so-called transit traffic.

The applicability of FCFS SAVI is limited to the local traffic i.e. to verify if the traffic generated by the hosts attached to the local link contains a valid source address. The verification of the source address of the transit traffic is out of the scope of FCFS SAVI. Other techniques, like ingress filtering [RFC2827], are recommended to validate transit traffic. In that sense, FCFS SAVI complements ingress filtering, since it relies on ingress filtering to validate transit traffic but is provides validation of local traffic, which is not provided by ingress filtering. Hence, the security level is increased by using these two techniques.

2.2. Constraints for FCFS SAVI

FCFS SAVI is designed to be susceptible of deployment in existing networks requiring a minimum set of changes. For that reason, FCFS SAVI does not require any changes in the hosts which source address is to be verified. Any verification must solely rely in the usage of already available protocols. This means that FCFS SAVI cannot define a new protocol nor to define any new message on existing protocols nor to require that a host uses an existent protocol message in a different way. In other words, the requirement is no host changes.

FCFS SAVI validation is performed by the FSFC SAVI function. Such function can be placed in different type of devices, including a router or a layer-2 bridge. The basic idea is that the FCFS SAVI function is located in the points of the topology that can enforce the correct usage of source address by dropping the non-compliant packets.

2.3. Address ownership proof

The main function performed by FCFS SAVI is to verify that the source address used in data packets actually belongs to the originator of the packet. Since FCFS SAVI scope is limited to the local-link, the originator of the packet is attached to the local-link. In order to to define any source address validation solution, we need to define some address ownership proof concept i.e. what it means to be able to proof that a given host owns a given address in the sense that the host is entitled to send packet with that source address.

Since no hast changes are acceptable, we need to find the means to proof address ownership without requiring a new protocol. In FCFS SAVI the address ownership proof is based in the First-Come first Serve approach. This means that the first host that uses a given source address is the owner of the address until further notice. More precisely, whenever a source address is used for the first time, a state is created in the device that is performing the FCFS SAVI function binding the source address to the layer-2 information that the FCFS SAVI box has available (e.g. the MAC address in a LAN, or the port in a switched LAN). Following data packets containing that IP source address must use the same layer-2 information in order to be compliant.

There are however additional considerations to be taken into account. For instance, consider the case of a host that moves from one segment of a LAN to another segment of the same subnetwork and it keeps the same IP address. In this case, the host is still the owner of the IP address, but the associated layer-2 information has changed. In order to cope with this case, FCFS SAVI performs an active check to verify if the host is still reachable using the previous layer-2 information. In order to do that FCFS SAVI uses ARP protocol in IPv4 and ND in IPv6. If the host is no longer reachable at the previously recorded layer-2 information, FCFS SAVI assumes that the new location is valid and creates a new binding using the new LAyer-2 information. In case the host is still reachable using the previously recorded information, the packets coming from the new layer-2 information are dropped (see some caveats described in the following section).

Note that this only applies to local traffic. Transit traffic generated by a router would be verified using alternative techniques,

such as ingress filtering. ARP or ND checks would not be fulfilled by the transit traffic, since the router is not the owner of the source address contained in the packets.

Layer-2 considerations:TBD

2.4. Special cases

The following special cases that need to be considered

- Hosts with multiple physical interfaces, potentially connected to different networks.
- Anycast i.e. multiple hosts using the same source address to send packets.
- Proxy ARP/ND i.e. host sending packets on behalf of other, in a layer-3 transparent manner.

3. FCFS SAVI specification

3.1. FCFS SAVI Data structures

FCFS SAVI function relies on state information binding the source address used in data packets to the layer-2 information that contained the first packet that used that source IP address. Such information is stored in FCFS SAVI Data Base (DB). The FCFS SAVI DB will contain a set of entries about the currently used IP source addresses. So each entry will contain the following information:

- IP source address
- Layer-2 information, such as Layer-2 address, port through which the packet was received, etc
- Lifetime
- Status:either tentative or valid
- Creation time: the value of the local clock when the entry was firstly created

In addition to this, FCFS SAVI need to know what are the prefixes that are directly connected, so it maintains a data structure called the the FCFS SAVI prefix list, which contains:

- Prefix
- Interface where prefix is directly connected

Finally, FCFS SAVI keep a list of the routers that are directly connected, since the FCFS SAVI checks will not directly apply to them. In the FCFS SAVI Router List, the following information is stored:

- Router IP address (of the directly connected interface)

- Router Layer-2 information such as layer-2 address or port which the router is connected to

3.2. FCFS SAVI algorithm

3.2.1. Processing of data packets

The FCFS SAVI function is located in a forwarding device, such as a router or a layer-2 bridge. Upon the reception of a data packet, the packet will be passed to the FCFS SAVI function which will perform the processing detailed in this section. The outcome of such processing can be that the packet is discarded or that is forwarded as usual.

After a data packet is received, the FCFS SAVI function checks whether the received data packet is local traffic or transit traffic. It does so by verifying if the source address of the packet belongs to one of the directly connected prefixes available in the receiving interface. It does so by searching the FCFS SAVI Prefix List.

- If the IP source address belongs to one of the local prefixes of the receiving interface, the data packet is local traffic and the FCFS SAVI algorithm is executed as described next.
- If the IP source address does not belong to one of the local prefixes of the receiving interface, this means that the dat packet is transit traffic. The FCFS SAVI SHOULD verify if the layer-2 information of the packet corresponds to one of the routers available in the receiving interface, by using the information available in the FCFS SAVI router list. If the packet comes from one of the know routers for that interface, then the packet is passed so additional checks such as ingress filtering can be performed. If the packet does not comes from one of the known routers, then the packet SHOULD be discarded. The FCFS SAVI function MAY send an ICMP Destination Unreachable Error back to the source address of the data packet. (In ICMPv4, code 0 (net unreachable) should be used and in ICMPv6, code 5 (Source address failed ingress/egress policy) should be used) (Note; we could skip this verification altogether and simply pass it to the ingress filters, but it think this could be useful, especially if used along with SeND)

After checking that the data packet is local traffic, the FCFS SAVI function will verify the source address used in the packet. In order to do so, it searches the FCFS SAVI DB using the IP source address as a key.

- If no valid entry is found, then a new entry is created, using the information of the data packet, including all the related layer-2 information of where the packet was received from and the lifetime of the entry is set to LIFETIME. The status is set to valid. The

packet is forwarded as usual. (NOTE: AS defined FCFS SAVI treats tentative entries as if they did not existed i.e. a data packet preempts the DAD procedure, this probably requires more discussion)

- o If a valid entry is found and the layer-2 information of the received data packet matches to the information contained in the existing entry, then the lifetime is set of LIFETIME and the packet is forwarded as usual.
- o If a valid entry is found and the layer-2 information of the received data packet does not match the information contained in the existing matching entry, then the FCFS SAVI performs a Neighbor Unreachability Detection procedure as described in [RFC4861] for IPv6 and in Section 3.3 for IPv4. It uses the IP source address and Layer-2 information available in the FCFS SAVI DB entry.
  - * If the procedure determines that the neighbor is no longer reachable using the information available in the FCFS SAVI DB entry, then the entry information is modified to include the new information about the data packet received (in particular the new layer-2 information) and lifetime of the entry is updated to LIFETIME. The packet is forwarded as usual.
  - * If the procedure determines that the neighbor is still reachable using the information available in the FCFS SAVI DB, then the data packet is discarded and the lifetime of the entry is set to LIFETIME. The FCFS SAVI function MAY send an ICMP Destination Unreachable Error back to the source address of the data packet. (In ICMPv4, code 0 (net unreachable) should be used and in ICMPv6, code 5 (Source address failed ingress/ egress policy) should be used)

3.2.2. Processing of control packets

Processing of IPv6 ND packets

The FCFS SAVI function will also create state based on control packets. In particular, in IPv6, when a host configures an address, it performs the Duplicate Address Detection (DAD) procedure, to verify that the address is unique in the link. FCFS SAVI keeps track of the DAD procedure and creates modify the FCFS SAVI DB state accordingly.

Upon the reception of a Neighbor Solicitation message containing the unspecified source address FCFS SAVI retrieves the address contained in the Target Address filed of the NSOL message and performs the following actions:

- o If no valid entry is found in the FCFS SAVI DB for that address, then it creates a new entry, includes the Target Address and the link layer information contained in the NSOL message and sets the

status to tentative. At that point FCFS SAVI will keep track of the Neighbor Advertisement messages.

* If a NADV message containing the address in the NADV Target Address field is received before DADTimeout then the entry is deleted.
* If no NADV message for that Target Address is received in DADTimeout, then the status of the entry is change to valid and the lifetime of the entry is set to LIFETIME. In addition, if the address contained in the newly created entry is a link local address, FCFS SAVI MAY as well create entries for the global addresses resulting from concatenating the Interface Identifier of the link local address and the global prefixes contained in the Prefix List for the Interface through which the NSOL message was received.

o If a valid entry is found in the FCFS SAVI DB for that address, no additional processing is performed. (Note: there is no point of tracking the NADV at this point. Either the SAVI DB is updated and there is no new information or it is not, which we will find out when we receive a data packet. Moreover, tracking NADV messages could enable an attacker to overwrite an existing entry.)

Processing of IPv4 ARP packets

IPv4 Address Conflict Detection (ACD) is defined in [RFC5227] and provides the means to verify if there is an address conflict in IPv4. The FCFS SAVI function will also create state based on IPv4 ACD control packets. In IPv4, when a host configures an address, it performs the Address Conflict Detection (ACD) procedure, to verify that the address is unique in the link. FCFS SAVI keeps track of the ACD procedure and creates modify the FCFS SAVI DB state accordingly.

Upon the reception of a ARP Probe (defined as an ARP Request message, broadcast on the local link, with an all-zero 'sender IP address'), FCFS SAVI retrieves the address contained in the 'target IP address' field of the ARP Request message and performs the following actions:

o If no valid entry is found in the FCFS SAVI DB for that address, then it creates a new entry, includes the 'target IP address' and the link layer information contained in the ARP Request message and sets the status to tentative. At that point FCFS SAVI will keep track of ARP Announcement messages. ARP Announcement messages are defined as an ARP Request message, broadcast on the local link, with a non all-zero 'sender IP address')
    * If an ARP Announcement message containing the tentative address in both the 'sender IP address' and the 'target IP address' and it contains the same link-layer information stored in the tentative entry in the SAVI DB is received before 3*ACDTimeout. (default value of 3*2 secs) then the entry status is set to valid and the lifetime of the entry is set to LIFETIME.

  * If no such message is received for that Target Address in 3*ACDTimeout (default value of 3*2 secs), then the entry is deleted.

- If a valid entry is found in the FCFS SAVI DB for that address, no additional processing is performed.

3.3. IPv4 Neighbor Unreachability Detection Procedure

As opposed to IPv6, there is no general Neighbor Unreachability Detection procedure defined for IPv4. Since this is needed in order to verify if the original node is still using the IP address it once used, in this section, we define the procedure to perform such verification. However, unlike IPv6 Neighbor discovery, the IPv4 ARP protocol [RFC0826] cannot be assumed to be available in all link layers. So, we will define a ARP based procedure to be used in layers 2 that the ARP protocol is available and an ICMP based [RFC0792] procedure for the cases where the ARP protocols is not available. The ARP based procedure is used whenever it is possible and when ARP is not available, the ICMP based procedure is used.

3.3.1. ARP-based Neighbor Unreachability Detection procedure

Consider two nodes, S and T, directly connected through a layer 2 where the ARP protocol is available. Node S has with IP address IPS and layer 2 address MACS and Node T has IP address IPT and layer 2 address MACT.

Node S wants to perform the ARP based Neighbor Unreachability Detection Procedure for node T. Node S has both IPT and MACT available. So, node S generates an ARP REQUEST packet, containing the following information:

Ethernet transmission layer:
- Ethernet address of destination: MACT
- Ethernet address of sender: MACS
- Protocol type = ether_type$ADDRESS_RESOLUTION

Ethernet packet data:
- (ar$hrd) Hardware address space (e.g., Ethernet, Packet Radio Net.)
- (ar$pro) Protocol address space:0x0800 Internet Protocol Version 4 (IPv4)
- (ar$hln) byte length of each hardware address
- (ar$pln) byte length of each protocol address: 4
- (ar$op) opcode (ares_op$REQUEST)
- (ar$sha) Hardware address of sender of this packet: MACS
- (ar$spa) Protocol address of sender of this packet: IPS
- (ar$tha) Hardware address of target of this packet: MACT

(ar$tpa) Protocol address of target: IPT

Upon the reception of the ARP REQUEST, if node T follows current ARP specification [RFC0826], it will reply with an ARP REPLY packet with the following information:

Ethernet transmission layer:
Ethernet address of destination: MACS
Ethernet address of sender: MACT
Protocol type = ether_type$ADDRESS_RESOLUTION
Ethernet packet data:
(ar$hrd) Hardware address space (e.g., Ethernet, Packet Radio Net.)
(ar$pro) Protocol address space:0x0800 Internet Protocol Version 4 (IPv4)
(ar$hln) byte length of each hardware address
(ar$pln) byte length of each protocol address: 4
(ar$op) opcode (ares_op$REPLY)
(ar$sha) Hardware address of sender of this packet: MACT
(ar$spa) Protocol address of sender of this packet: IPT
(ar$tha) Hardware address of target of this packet: MACS
(ar$tpa) Protocol address of target: IPS

If node S receives the ARP REPLY message, the Neighbor Unreachability procedure was successful and the neighbor T is still reachable with the available information. If node S does not receives the ARP REPLY message after ARPTIMEOUT, then the Neighbor Unreachability procedure has failed and the neighbor T is no longer reachable with the current information.

3.3.2. ICMP-based Neighbor Unreachability Detection procedure

Consider two nodes, S and T, directly connected through a layer 2. Node S has with IP address IPS and layer 2 address LLAS and Node T has IP address IPT and layer 2 address LLAT.

Node S wants to perform the ICMP based Neighbor Unreachability Detection Procedure for node T. Node S has both IPT and LLAT available. So, node S generates an ICMP ECHO packet [RFC0792] , containing the following information:

Link Layer fields:
Source address: LLAS
Destination address: LLAT
IP header fields:
IP Source Address: IPS
IP Destination Address: IPT
ICMP fields

Type: 8
Identifier: set to random number by S

Upon the reception of the ICMP ECHO message, if node T follows current ICMP specification [RFC0792], it will reply with an ECHO REPLY packet with the following information:

Link Layer fields:
Source address: LLAT
Destination address: LLAS
IP header fields:
IP Source Address: IPT
IP Destination Address: IPS
ICMP fields
Type: 0
Identifier: copied from the ECHO message received

If node S receives a ECHO REPLY message, it will verify that the source IP address and the source link layer address match to the original ones used in the ECHO message. Besides, it will check that the identifier matches to the one contained in the original ECHO message. If these checks are successful the Neighbor Unreachability procedure was successful and the neighbor T is still reachable with the available information. If node S does not receives the ECHO REPLY message after ICMPTIMEOUT, then the Neighbor Unreachability procedure has failed and the neighbor T is no longer reachable with the current information.

4. Security Considerations

First of all, it should be noted that any SAVI solution will be as strong as the lower layer anchor that it uses. In particular, if the lower layer anchor is forgeable, then the resulting SAVI solution will be weak. For example, if the lower layer anchor is a MAC address that can be easily spoofed, then the resulting SAVI will not be stronger than that. On the other hand, if we use switch ports as lower layer anchors (and there is only one host connected to each port) it is likely that the resulting SAVI solution will be considerably more secure.

Denial of service attacks

There are two types of DoS attacks that can be envisaged in a SAVI environment. On one hand, we can envision attacks against the SAVI device resources. On the other hand, we can envision DoS attacks against the hosts connected to the network where SAVI is running.

The attacks against the SAVI device basically consist on making the

SAVI device to consume its resource until it runs out of them. For instance, a possible attack would be to send packets with different source addresses, making the SAVI device to create state for each of the addresses and waste memory. At some point the SAVI device runs out of memory and it needs to decide how to react in this situation. The result is that some form of garbage collection is needed to prune the entries. It is recommended that when the SAVI device runs out of the memory allocated for the SAVI DB, it creates new entries by deleting the entries which Creation Time is higher. This implies that older entries are preserved and newer entries overwrite each other. In an attack scenario where the attacker sends a batch of data packets with different source address, each new source address is likely to rewrite another source address created by the attack itself. It should be noted that entries are also garbage collected using the LIFETIME, which is updated using data packets. The result is that in order for an attacker to actually fill the SAVI DB with false source addresses, it needs to continuously send data packets for all the different source addresses, in order for the entries to grow old and compete with the legitimate entries. The result is that the cost of the attack for the attacker is highly increased.

The other type of attack is when an attacker manages to create state in the SAVI device that will result in blocking the data packets sent by the legitimate owner of the address. In the IPv4 case, the simplest way of doing this is for the attacker to claim the ownership of all the addresses available in the prefix assigned to the subnetwork. That is, if an attacker sends data packets with all the source addresses of the on-link prefix, it will claim address ownership for all the available addresses and SAVI will block packets sent by any other host. This is a very severe attack. The proposed solution for this attack is to limit the number of IP addresses bound to a give lower layer anchor. In this way, any host, including the attacker, can only claim the address ownership of a limited number of addresses. Of course, this is only effective if the attacker cannot spoof the lower layer anchor. For instance, in the case where the MAC address is used as lower layer anchor, this measure is hardly sufficient, since the attacker can spoof the source address and still perform the attack. As a result, it is recommended that when the lower layer anchors are spoofable, SAVI should not discard non-compliant packet, but rather log them, to enable proper administrative action. Enabling SAVI in that case could expose the network to the aforementioned DoS attack. If the lower layer anchor is not easily spoofable, the proposed mechanism provides considerable protection, since it limits the impact of the attack. In IPv6 these attacks are not an issue thanks to the 2^64 addresses available in each link.

Compare with Threat analysis and identify residual threats: TBD

5. Acknowledgments

This draft benefited from the input from: Christian Vogt, Fred Baker, Guang Yao, Dong Zhang, Frank Xia and Lin Tao. In particular the usage of ARP and ND packet to create SAVI DB state was suggested by Guang Yao in response to an attack described by Fred Baker.

Marcelo Bagnulo is partly funded by Trilogy, a research project supported by the European Commission under its Seventh Framework Program.

6. Normative References

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, September 1981.

[RFC5227] Cheshire, S., "IPv4 Address Conflict Detection", RFC 5227, July 2008.

Authors' Addresses

Erik Nordmark
Sun Microsystems, Inc.
17 Network Circle
Menlo Park, CA 94025
USA

Phone: +1 650 786 2921
Email: Erik.Nordmark@Sun.COM

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6248814
Email: marcelo@it.uc3m.es
URI: http://www.it.uc3m.es