ERNW
providing security.

ERNW WHITEPAPER 62

RA GUARD EVASION REVISITED

Version:	1.0
Date:	11.12.2017
Classification:	Public
Author(s):	Omar Eissa;Christopher Werny

Nichole Vonderwell

Digitally signed by Nichole Vonderwell
DN: c=DE, o=ERNW Insight GmbH, cn=Nichole Vonderwell, givenName=Nichole Christine, sn=Vonderwell, serialNumber=DTRWM741237304264922
Date: 2017.12.12 10:29:16 +01'00'

ERNW
providing security.

TABLE OF CONTENT

1 Motivation

On designing the IPv6 to be the successor to IPv4, multiple design changes have been taken into consideration; one of which was to have a simpler header format and a better support for extensions and headers[1]. Extensions support is already implemented within IPv4 header as it contains the Options field. This field is mainly used to give some additional information on the packet and the way it should be processed. The Options field has a significance effect on the performance. As a result, the Options field has been replaced with optional customized headers called extension headers. These extension headers can be added as needed after the IPv6 header.

These extension headers act as optional internet-layer information that is placed between the IPv6 Header and upper-layer header in the packet. These headers can be used in case of fragmentation or defining a path for the packet and other multiple cases that are out of scope of this paper.

Although RFC 8200 recommends that some of these headers can only appear at most once or twice within the packet. However, it is only a recommendation that one can stick to or not depending on the situation. There is no restriction on how many extension headers can be used within the packet. Regarding the processing of the packet, intermediate nodes should not examine or process any extension headers along the path except of that Hop-by-Hop Options header.

Speaking about fragmentation, unlike IPv4, intermediate nodes cannot fragment the packet; fragmentation is only performed by source nodes.

Another very important specification in the world of IPv6 is RFC 4861[2], which defines the Neighbor Discovery protocol for IPv6. For the scope of this paper, we concentrate on the Router Advertisement (RA) Message, it is the message sent by the router to the hosts to provide information about the network prefix, default routes and potential DNS servers within the network.

In the past, the general extension header concept created several security problems that include the evasion of IDPS devices/appliances[3] and first hop security features (e.g. RA Guard) on typical enterprise grade access layer switches. While these problems are not new, the goal of this whitepaper is to get an impression whether the situation (on an infrastructure level) has improved in the meantime specifically in the context of RA Guard.

[1] *https://tools.ietf.org/html/rfc8200*
[2] *https://tools.ietf.org/html/rfc4861*
[3] *https://www.ernw.de/download/Atlasis_Rey_Schaefer_IETF93_IPv6Hackers_Evasion_of_HighEnd_IPS_Devices.pdf*

2 Problem Statement

RA messages, which are part of the Neighbor Discovery (ND) protocol specification, are the core of IPv6 functionality, by which hosts can automatically configure themselves, get information about the router, and add the default routes in addition to the DNS information. However, ND was developed without security being the main focus as the local link deemed to be secure. There is no feasible authentication mechanism implemented within the protocol[4]. This assumption created security problems that enabled the generation of malicious router advertisements from a presumably legitimate source. These problems include (but are not limited) to

- Rogue RA attacks: In such an attack, the adversary sends fake RAs to get a man-in-the-middle position by installing a malicious default route on the victims. Another attack scenario is the flooding of RAs within the local segment. This primarily results in a DoS condition on the systems within the segment.
- Rogue DHCPv6 Server: The Attacker sends malicious DHCPv6 messages that contain falsified information about prefixes, DNS servers, and so on that could be used to redirect the traffic.
- Neighbor Spoofing: An Attacker sends malicious Neighbor Advertisement to poison the neighbor cache of the victim.

2.1 First Hop Security

The IPv6 First Hop Security (FHS) Suite[5] was developed by Cisco primarily to provide defense mechanisms on access layer switches to mitigate the aforementioned attacks. Depending on the hardware platform, the number of supported features is different, but all typical access layer platform support at least RA Guard nowadays.

- RA Guard: Blocks any unauthorized (malicious) router advertisements received on a RA Guard port.
- DHCPv6 Guard: Blocks messages that originates from malicious DHCP servers and relay agents.
- IPv6 Snooping: Analyzes IPv6 NS/NA traffic by detecting IPv6 / MAC address bindings where neighbor discovery messages that do not have valid bindings are dropped.

2.2 Research

Within this whitepaper, we concentrate on the countermeasures implemented by Cisco to stop RA based attacks.

RA attacks on Cisco switches can be stopped by either one of two ways:

- Implementing RA Guard on the untrusted interfaces

[4] While the IETF developed SeND in RFC 3971 to theoretically address these issues, it won't be implemented in typical enterprise environments due to the increased complexity and the missing support in Microsoft operating systems.
[5] https://www.cisco.com/c/en/us/products/ios-nx-os-software/ipv6-first-hop-security-fhs/index.html

- Implementing an Access-list (ACL) on the untrusted interfaces

Unfortunately, RA Guard can be easily circumvented by using fragmentation! In the end, this led for calls to forbid IPv6 fragmentation in the Neighbor Discovery protocol as indicated by RFC 6980. The RFC also stresses that nodes must ignore RAs if they contain a fragmentation header[6]. While out of scope of this document, it could be observed that not all operating systems implement RFC 6980 correctly[7].

Since RA-Guard can be easily circumvented using both extension headers and fragmentation, Cisco decided to introduce a new entry to the ACL to stop rogue RAs. The new entry is called *undetermined transport* and it allows the switch to drop all fragments with unknown upper-layer protocol information.[8] We included this entry into our test setup to evaluate the effectiveness of this ACE.

In case of failure of RA Guard and the undetermined-transport ACL to stop the adversary, we shall use a tailored ACL that stop fragmentation and RA messages. Throughout this paper, we tested more than 20 different ways to evaluate the protection controls introduced by Cisco.

[6] *https://tools.ietf.org/html/rfc6980*
[7] *https://insinuator.net/2017/12/lets-talk-about-rfc-6980/*
[8] *https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_acl/configuration/xe-16/sec-data-acl-xe-16-book/ipv6-undeter-trans-xe.html*

3 Lab Setup and Testing Methodology

Our lab consists mainly of 4 machines and our test is divided into 4 phases.

3.1 Lab Setup

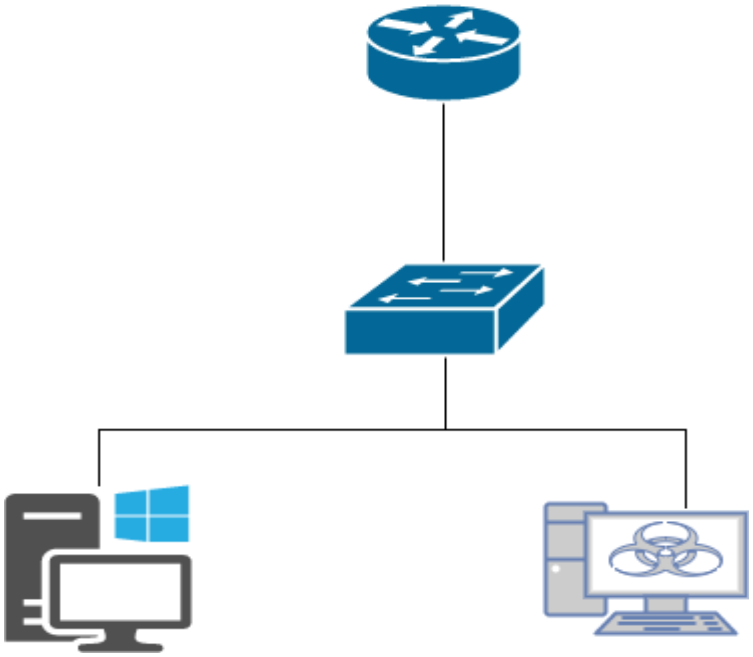

Figure 1: Lab Setup

Our lab consists of the following devices:

- Windows 10 Fall Creators update acting as the victim
- Ubuntu 16.04 LTS with Chiron 0.8.1 acting as the attacker
- Both machines are connected through a Catalyst 3650 with Everest 16.5.1a as the software image
- A legitimate router sending RA messages every 10 seconds

3.2 Testing Methodology

The main aim of this research is to find new ways to evade the RA protection mechanisms on Cisco switches. We try to use different combinations of extension headers and multiple fragments to bypass the security mechanism implemented. Our test is divided mainly into 4 phases:

1. Try different combinations without the presence of any protection on the switch to see which combinations will be accepted by the Windows machine.

2. The successful combinations from the 1st phase will be tested again, however this time while RA guard is configured on the switch.
3. The combinations that managed to bypass the RA guard will be tested against the undetermined transport ACL on the switch.
4. If still there are some viable combinations, these will be tested against a tailored ACL

4 Performed Tests and Results

We used Chiron 0.8.1[9] as the basis for our tests, with the basic command running on the attacker machine looks like the following:

```
chiron_local_link.py eth0 -ra -pr 2001:1:db8:10:50:: -pr-length 64 -mtu 1400 -s
fe80::3aea:a7ff:fe85:c926
```

This command can be understood as, send RA message on eth0 interface with a specific prefix and with a specific source address. A comprehensive tutorial for Chiron is available on this link.

The tested combinations vary between adding the extension headers sometimes to the fragmentable part and other times to the unfragmentable part.

The combinations that are processed in the 1st phase will be tested again with RA Guard enabled. RA Guard is configured on a per interface basis using the following command:

```
Switch(config-if)#ipv6 nd raguard
```

In case the RA Guard has been bypassed, the following ACL (with the *undetermined-transport* ACE) is used on the interface facing the attacker.

```
Switch(config)#ipv6 access-list undetermined
deny ipv6 any any undetermined-transport
permit ipv6 any any
```

In case the undetermined transport ACL cannot stop the attack, the following tailored ACL will be used

```
Switch(config)#ipv6 access-list no_RAs
deny icmp any any router-advertisement
deny ipv6 any host FF02::1 fragments
deny ipv6 any host FF02::C fragments
deny ipv6 any host FF02::FB fragments
deny ipv6 any host FF02::1:3 fragments
deny ipv6 any FF02::1:FF00:0/104 fragments
deny ipv6 any FE80::/64 fragments
permit ipv6 any any
exit
```

This ACL stops any fragments sent on the local address and stops any RA sent to any address regardless of its type, it also stops any fragments sent on the node solicited multicast address. This ACL is believed to be able to stop most RA attacks as it blocks fragmentations to local-link addresses and multicast node solicitation addresses but increases the complexity as one must implement this ACL on every switch port throughout the network.

[9] *https://www.secfu.net/tools-scripts/*

4.1 Test Cases

To test how effective the security mechanisms of Cisco are, we created and used 18 different combinations. The combinations can be mainly divided into 3 categories:

4.1.1 No Extension Headers

For the first 3 test cases, we send a plain RA and observe its effect on the victim machine. We further went through cases 2 and 3 to fragment this RA to see how effective this methodology in bypassing the 1st phase of the protection mechanism which is “RA guard only”. RA Guard proved its efficiency in blocking plain RAs and hence no packet was forwarded to the victim.

4.1.2 Extension Headers in the Unfragmentable Part

This is the approach used for cases from 4 till 9. We start using extension headers:

Case 4: We added the Destination Options extension header to the RAs. However, RA guard managed to stop such an attack.

Case 5: For this case, we decide to add multiple extension headers; we use both Hop-by-Hop (HBH) and the Destination Options extension headers. RA guard managed also to stop such an attack.

Case 6: As per RFC 8200 (S. Deering, R. Hinden, 2017) the HBH must come directly after the IPv6 header. We send the same combination as Case 5, however this time we put the Destination Options before the HBH. Even without RA guard, this packet is rejected by the operating system.

In the next following 3 cases, we combine both some extension headers with fragmentation. The fragmented packet consists mainly of 2 parts, a fragmented and a non-fragmented part as shown in Figure 2.

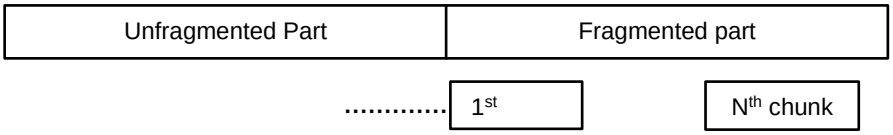

Figure 2: Fragmented Packet

The fragmented part can be divided after that into multiple smaller chunks. Using Chiron, the adversary can choose to which part to add the extension headers.

Case 7: We add the Destination Options to the unfragmented part and we divide the RA on 2 chunks. RA guard can stop such an attack.

Case 8: We use 2 extension headers and we divide the RA into 2 chunks. RA guard manage to stop such an attack.

Case 9: We use 3 extension headers and the RA is divided into chunks. No luck here too as RA guard manage to stop such an attack.

4.1.3 Extension Headers in the Fragmentable Part

For the following 9 cases, we use the extension headers in the fragmented part of the packet and we use multiple chunks.

Case 10: One extension header and the RA are both divided into 2 chunks. RA guard spots such an attack and drops the packet.

Case 11: We try to use a different header from that used in Case 10. In this case, we use the Routing Header and use 2 chunks but no luck as the RA guard stops such an attack.

Case 12: We use Hop by Hop header this time but the result still the same. We can't bypass the RA guard.

Case 13: We use 2 Destination Options extension headers in the fragmented part in addition to the RA and we divide all of that into 2 chunks. No joy as RA guard stops such an attack

Case 14: The same as case 13 but with Routing extension headers. No change in the output, still RA stops such an attack.

Case 15: We use 2 extension headers but this time we divide it into 4 chunks. The attack manages to bypass the RA guard, and the undetermined ACL. The tailored ACL manages to stop such an attack.

Case 16: This time we use 4 extension headers but divide them into 2 chunks. RA guard manages to spot the attack and stops it.

Case 17: Same as case 16 but this time with 3 chunks instead of 2. This attack manages to bypass both RA Guard and the undetermined ACL. Only the tailored ACL stops such an attack.

Case 18: Same as case 16 but using 4 chunks. We manage to bypass the RA guard and the undetermined ACL, however we fail to pass the tailored one.

4.2 Results Summary

Table 1 shows the results of our tests, the flags used and the significance of each command. All the flags mentioned within the tables are used with the basic command mentioned earlier.

Test Case No	Description	Chiron additional option used	Impact on Target (without RA guard)	Packets received by Target (without RA guard)	What still got through with RA Guard enabled?	What still got through with the tailored ACL configured?	Overall Result
1(baseline)	No fragmentation, no EHs	none	Added 2nd default GW, created additional address	Full packet	Nothing	Nothing	Enabling RA guard or configuring ACL will stop such an attack
2	Split RA into 2 fragments	-nf 2	Added 2nd default GW, created additional address	1st fragment plus one RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
3	Split RA into 4 fragments	-nf 4	Added 2nd default GW, created additional address	1st three fragments plus RA packet	3 fragments but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
4	No fragmentation one DestOptions added in unfragmentable part	-luE 60	Added 2nd default GW, created additional address	Full packet	Nothing	Nothing	Enabling RA guard or configuring ACL will stop such an attack
5	No fragmentation, one HBH + one DestOptions	-luE 0,60	Added 2nd default GW, created additional address	Full packet	Nothing	Nothing	Enabling RA guard or configuring ACL will stop such an attack
6	No fragmentation, one DestOptions + HBH added	-luE 60,0	None (according to RFC 8200 HBH must 1st EH in the chain)	Full packet, but Wireshark indicates problem	Nothing	Nothing	No need to implement anything here
7	2 fragments, one DestOptions in unfragmentable part	-luE 60 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
8	2 fragments, one HBH + DestOptions in unfragmentable part	-luE 0,60 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
9	2 fragments, one HBH + 2 DestOptions in unfragmentable part	-luE 0,60,60 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
10	2 fragments, 1 DestOptions in fragmentable part	-lfE 60 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack

11	2 fragments, 1 RoutingHdr in fragmentable part	-lfE 43 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
12	2 fragments, 1 HBH in fragmentable part	-lfE 0 -nf 2	None (according to RFC 8200 HBH header must be in unfragmentable part)	Both fragments but Wireshark indicates problem in 2nd	1st fragment but no RA	Nothing	No need to implement countermeasures
13	2 fragments, with 2 DestOptions in fragmentable part	-lfE 60,60 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
14	2 fragments, with 2 RoutingHdr in fragmentable part	-lfE 43,43 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
15	4 fragments, with 2 DestOptions in fragmentable part	-lfE 60,60 -nf 4	Added 2nd default GW, created additional address	3 fragments plus RA	3 fragments plus RA	Nothing	ACL only can stop this attack
16	2 fragments, 2 RHs and 2 DestOptions in mixed order	-lfE 60,43,60,43 -nf 2	Added 2nd default GW, created additional address	1st fragment plus RA	1st fragment but no RA	Nothing	Enabling RA guard or configuring ACL will stop such an attack
17	Same as 16 but 3 fragments	-lfE 60,43,60,43 -nf 3	Added 2nd default GW, created additional address	2 fragments plus RA	2 fragments plus RA	Nothing	ACL only can stop this attack
18	Same as 16 but 4 fragments	-lfE 60,43,60,43 -nf 4	Added 2nd default GW, created additional address	3 fragments plus RA	3 fragments plus RA	Nothing	ACL only can stop this attack

We tested all the successful cases that managed to bypass the RA guard against the undetermined ACL, however the results were disappointing/surprising as it followed exactly the same behavior as RA Guard. We are currently not sure whether this is intended behavior or a bug in the tested IOS-XE release. At least in this specific release, we do not get any advantage by implementing the undetermined ACL compared to plain RA Guard; both can be circumvented easily using fragmentation and extension headers. As a result, the table only included 3 phases.

The following table show the relation between number of extension headers and the minimum number of fragmentations needed to pass the RA guard/undetermined transport ACL.

Number of Extension Headers needed	Minimum number of fragmentations required
1	5
2	4
3	3
4	3
5	3
6	3
7	2

Table 1: Relation of Extension Headers and Fragmentations to pass RA Guard

As a result, the tailored ACL proved to be the best approach to mitigate RA based attacks.

We decided to test other approaches and see whether the tailored ACL can be circumvented by clever combinations of extension header/fragments.

First, we sent the RA to the global unicast address of the victim's machine. This attack failed as the ACL blocks every ICMPv6 RA message regardless of the destination. However, if we fragment the RA message and used the extension headers (any combinations of the previously valid combinations used to pass the RA guard), this RA will pass the ACL and be received by the victim. By using this approach, we managed to bypass the ACL, however the RA is not processed by the victim machine as the message is not received on neither the all nodes address (FF02::1) nor the solicited node multicast address (FF02::1:FF00::/104). These addresses are already blocked by the ACL and any fragmented packets sent to these addresses will be dropped.

In the next attempt, we combined the approaches to get the best of both worlds. The RA is sent in several fragments and destined to the IPv6 global unicast address of the victim, but we used the multicast MAC address of the link local all nodes group (ff02::1 / 33:33:00:00:00:01) as destination. Interestingly, this packet bypassed the ACL and was processed by the victim (which in turn generated 2 addresses from the provided prefix and

installed the malicious gateway). Another approach we found was to use the multicast MAC address of the solicited node multicast group that corresponds to a configured address.

One of the ways to stop such an attack is to deny fragmentation regardless of the destination, however one should be aware of its network environment and how such ACL can affect its operation.

5 Conclusion & Future Work

Throughout this paper, we have discussed the IPv6 extension headers, we have shown that regardless what the specifications recommend, one can have as much as extension headers as they need. We have spoken about fragmentation and how it can be used to bypass security mechanisms on Cisco switches. We briefly discussed the IPv6 Neighbor Discovery Protocol with special emphasis on the RA messages. The goal of this paper is to test the security mechanisms on Cisco switches against rogue RA attacks by implementing RA guard, undetermined transport ACL and even a tailored ACL that drops RA and any fragments sent to the link-local addresses.

We have concluded that none of all introduced mechanisms can stop the attacks. For RA guard, we have introduced a table showing the number of extension header and minimum number of fragments needed to bypass the RA-Guard. Based on our testing the undetermined transport ACL does not provide any additional security[10] than the plain RA guard command.

The tailored ACL appears to be the best option as it stopped all the attacks that managed to bypass the RA guard. However, even with this ACL we determined possible combinations of source/destination IPv6/MAC address that were able to circumvent the ACL as well.

In short, the situation (on an infrastructure level) hasn't improved based on the experience we gained throughout this research. Maybe it has to do with the specific IOS-XE release we used on the Catalyst 3650 and we will further test the effectiveness of the controls with different switches and software versions.

[10] *As already said, this might be a bug.*