Researching IPv6 Security Capabilities (RISC)

Bio:
Antonios Atlasis

- An IT engineer for more than 20 years, developer and instructor in several Computer Science and Computer Security related fields.
- Penetration tester, incident handler and intrusion analyst and cyber-researcher for the last 7 years.
- MPhil (University of Cambridge), PhD (National Technical University of Athens).
- More than 25 scientific papers published in several international Journals and Conferences.
- Several GIAC certifications (GCIH, GWAPT, GREM, GPEN, GCIA and GXPN), and a Giac Gold Adviser (having supervised more than 20 Giac Gold papers).
- Latest security researching interests: IPv6, IDS/IPS and WAF evasions, SCADA systems. Some of the work has been presented at BlackHat Europe 2012, BlackHat Abu Dhabi 2012 and Troopers 13 International security conferences.
- Currently working as an independent IT security analyst/consultant. Can reach me at **aatlasis@secfu.net**

Disclaimer

- Neither ERNW nor Antonios Atlasis are vendor affiliated.
- The presented results will simply demonstrate are pure findings.
- Performed tests were chosen based solely on the best of our knowledge / imagination.
- Devices were tested under exactly the same conditions.

Outline of the Presentation

- Introduction to the RISC project.
  - Goal of the project.
  - List of the tested devices.
  - Used tools
- (quotes from) RFC guidelines.
- Description of the tests.
- Results (per device)
- Conclusions

Goal of the Project

- To test some representative IPv6 security devices regarding:
  - Their IPv6 Security Capabilities.
  - The IPv6 security-related configuration capabilities that they offer.
  - Their RFC-compliance.

Tested Devices

- **Firewalls**:
  - Cisco ASA 5505 running firmware 9.1(4)
  - Checkpoint Gaia Release 77.10 running on commodity hardware
  - Juniper SRX 100H2 running JunOS 12.1X46-DH.2
  - Fortinet Fortigate 200B running v5.0,build0252 (GA Patch 5)
- **IDS**
  - Tipping Point, TOS Package 3.6.1.4036 and digital vaccine 3.2.0.8530.
    - Used as an IPS and inline.
- **Layer-2 switch**
  - Cisco Catalyst 4948E running Cisco IOS Release 15.2(1)E1.

Tool Used for Testing

- **Chiron** (an all-in-one IPv6 Pen-Testing Framework) running in a Linux Box (can be downloaded from www.secfu.net)
- **Wireshark/tcpdump** at both ends (attacker's and target's machine).
- Target's (victim's) OS did not matter during the tests.

RISC: Before We Start

Some Background Information Regarding the Processing of IPv6 Extension Headers
(or, ***what the RFCs say***)

Terminology

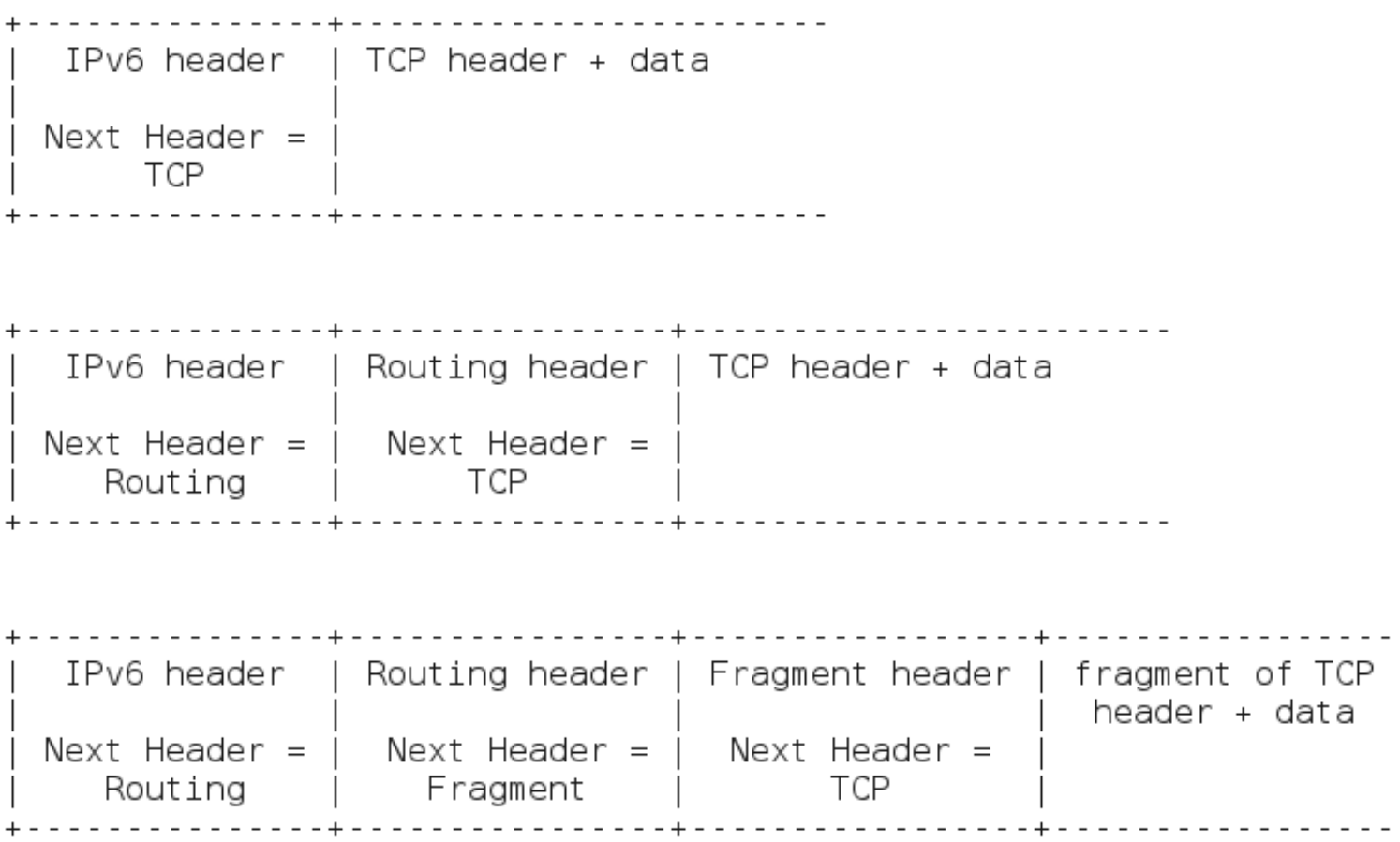
- ***node*** - a device that implements IPv6.
- Questions:
  - Is an IPv6 router a node?
  - Is an “IPv6 Ready” security device a node?

(Some of) the IPv6 Advantages

- ***Header Format Simplification****: Some IPv4 header fields have been dropped or made optional, to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.*
- ***Improved Support for Extensions and Options****: Changes in the way IP header options are encoded allows for more efficient forwarding, less stringent limits on the length of options, and greater flexibility for introducing new options in the future.*
- *In IPv6, optional internet-layer information is encoded in separate headers that may be placed between the IPv6 header and the upper-layer header in a packet. There are a small number of such extension headers, each identified by a distinct Next Header value. ...an IPv6 packet may carry zero, one, or more extension headers.*

Source: RFC 2460

IPv6 Datagram Chain



Source: RFC 2460

Order and Number of Occurrences of Ext. Headers

- *IPv6 nodes must accept and attempt to process extension headers in any order and occurring any number of times in the same packet, except for the Hop-by-Hop Options header which is restricted to appear immediately after an IPv6 header only. ...*
- *The Hop-by-Hop Options header, when present, MUST immediately follow the IPv6 header.*

Source: RFC 2460

Extension Headers Processing

- *With one exception, extension headers are not examined or processed by any node along a packet's delivery path, until the packet reaches the node (or each of the set of nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header.*
- *The contents and semantics of each extension header determine whether or not to proceed to the next header...*
- *...extension headers must be processed strictly in the order they appear in the packet.*

Source: RFC 2460

Unrecognised Extension Headers

- (if) *the Next Header value in the current header is unrecognized by the node, it should discard the packet and send an ICMP Parameter Problem message to the source of the packet, with an ICMP Code value of 1 ("unrecognized Next Header type encountered")...*

Source: RFC 2460

Recommended Order of Extension Headers

- *When more than one extension header is used in the same packet, it is recommended that those headers appear in the following order:*
  - *IPv6 header*
  - *Hop-by-Hop Options header*
  - *Destination Options header*
  - *Routing header*
  - *Fragment header*
  - *Authentication header*
  - *Encapsulating Security Payload header*
  - *Destination Options header*
  - *upper-layer header*

Source: RFC 2460

Number of Occurrences (again) and IPv6 Tunnelling

- *Each extension header should occur at most once, except for the Destination Options header which should occur at most twice (once before a Routing header and once before the upper-layer header).*
- *If the upper-layer header is another IPv6 header (in the case of IPv6 being tunnelled over or encapsulated in IPv6), it may be followed by its own extension headers, which are separately subject to the same ordering recommendations.*

Source: RFC 2460

IPv6 Extension Headers Processing

- *IPv6 nodes must accept and attempt to process extension headers in any order and occurring any number of times in the same packet, except for the Hop-by-Hop Options header which is restricted to appear immediately after an IPv6 header only.*
- *Nonetheless, it is strongly advised that sources of IPv6 packets adhere to the above recommended order ...*

Source: RFC 2460

Fragmenting an IPv6 Header Chain

- *The **Unfragmentable Part** consists of the IPv6 header plus any extension headers that must be processed by nodes en route to the destination, that is, all headers up to and including the Routing header if present, else the Hop-by-Hop Options header if present, else no extension headers.*
- *The **Fragmentable Part** consists of the rest of the packet, that is, any extension headers that need be processed only by the final destination node(s), plus the upper-layer header and data.*

Source: RFC 2460

Each Fragment is Composed Of

- The Unfragmentable Part of the original packet,...and the Next Header field of the last header of the Unfragmentable Part changed to 44.
- A Fragment header containing:
  - The Next Header value that identifies the first header of the Fragmentable Part of the original packet.

Source: RFC 2460

Reassembling a Fragmented IPv6 Datagram

- *The **Unfragmentable Part** of the reassembled packet consists of all headers up to, but not including, the Fragment header of the first fragment packet (that is, the packet whose Fragment Offset is zero), with the following change(s):*
- *The **Next Header** field of the last header of the Unfragmentable Part is obtained from the Next Header field of the first fragment's Fragment header.*

Source: RFC 2460

Delay in the reception of the fragments

- *If insufficient fragments are received to complete reassembly of a packet within 60 seconds of the reception of the first-arriving fragment of that packet, reassembly of that packet must be abandoned and all the fragments that have been received for that packet must be discarded.*
- *If the first fragment (i.e., the one with a Fragment Offset of zero) has been received, an ICMP Time Exceeded -- Fragment Reassembly Time Exceeded message should be sent to the source of that fragment.*

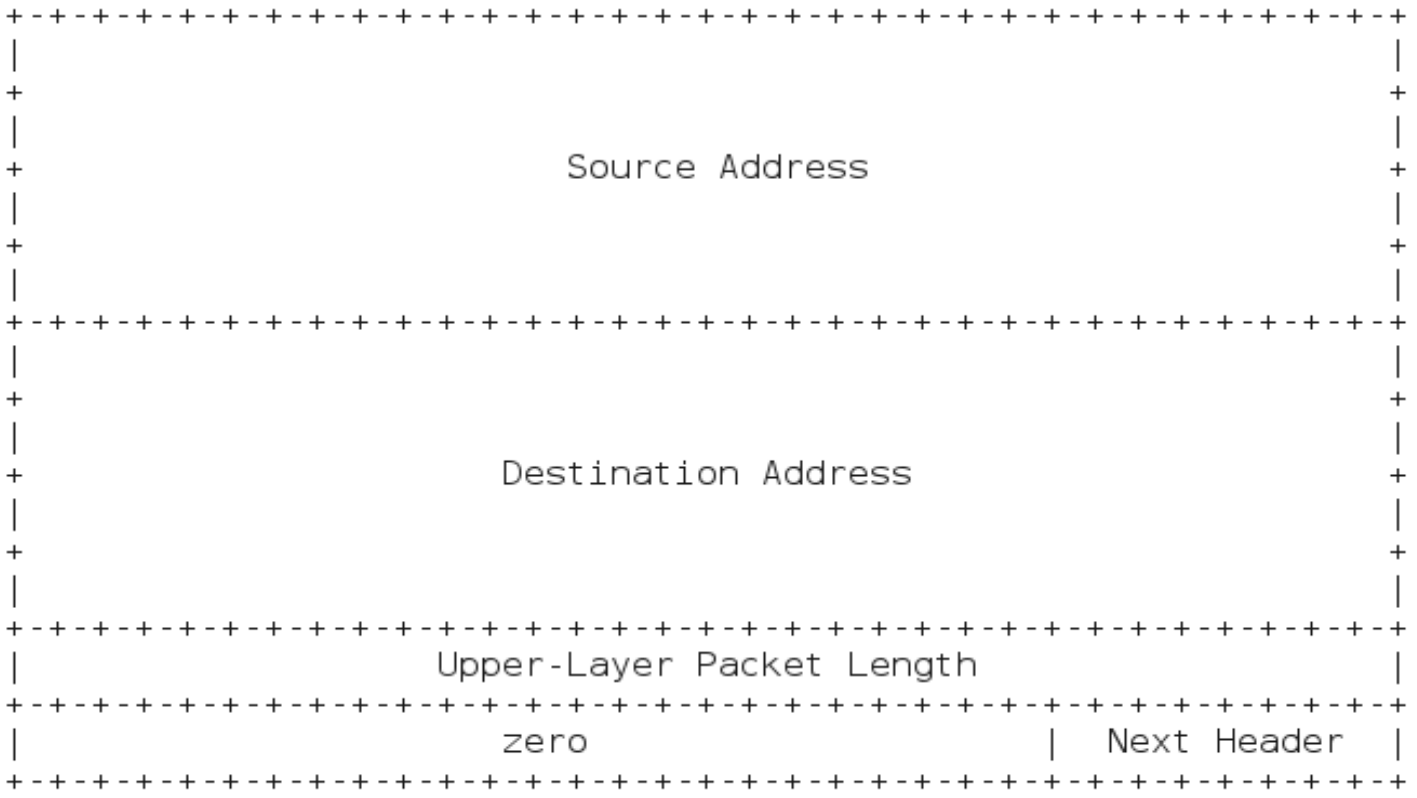
Source: RFC 2460

The following conditions are not considered errors:

- *The number and content of the headers preceding the Fragment header of different fragments of the same original packet may differ. Whatever headers are present, preceding the Fragment header in each fragment packet, are processed when the packets arrive, prior to queueing the fragments for reassembly. Only those headers in the Offset zero fragment packet are retained in the reassembled packet.*
- *The Next Header values in the Fragment headers of different fragments of the same original packet may differ. Only the value from the Offset zero fragment packet is used for reassembly.*

Source: RFC 2460

Upper-Layer Checksums



*The **Next Header** value in the pseudo-header identifies the upper-layer protocol (e.g., 6 for TCP, or 17 for UDP). It will differ from the Next Header value in the IPv6 header if there are extension headers between the IPv6 header and the upper-layer header.*

Source: RFC 2460

IPv6 Specification "Grey" Areas

- *The IPv6 Specification contains a number of areas where choices are available to packet originators that will result in packets that conform to the specification but are unlikely to be the result of a rational packet generation policy for legitimate traffic.*
- *The built-in flexibility of the IPv6 protocol may also lead to changes in the boundaries between legitimate and malicious traffic as identified by these rules.*

Source: RFC 4942

Processing at Middleboxes?

- [RFC2460] does not appear to take account the behavior of middleboxes and other non-final destinations that may be inspecting the packet, and thereby potentially limits the security protection of these boxes.
- In order to locate the transport header or other protocol data unit, the node has to parse the header chain.
- A middlebox cannot guarantee to be able to process header chains that may contain headers defined after the box was manufactured.

Source: RFC 4942

Extension Headers Clarification

- *Any forwarding node along an IPv6 packet's path:*
  - *SHOULD forward IPv6 packets regardless of any Extension Headers that are present.*
  - *They MUST recognise and deal appropriately with all standard IPv6 Extension headers.*
  - *They SHOULD NOT discard packets containing unrecognised extension headers.*

Source: RFC 7045

Implications of Oversized IPv6 Header Chains

- When a host fragments a IPv6 datagram, it MUST include the entire IPv6 header chain in the first fragment.
- A host that receives a First Fragment that does not satisfy the above-stated requirement SHOULD discard the packet and SHOULD send an ICMPv6 error message to the source address of the offending packet...
- An intermediate system (e.g., router or firewall) that receives an IPv6 First Fragment that does not satisfy the above-stated requirement MAY discard that packet, and it MAY send an ICMPv6 error message to the source address of the offending packet ...

Source: RFC 7112

Circumventing RA-Guard

- *IPv6 fragmentation introduces a key challenge for these mitigation or monitoring techniques, since it is trivial for an attacker to conceal his attack by fragmenting his packets into multiple fragments. This may limit or even eliminate the effectiveness of the aforementioned mitigation or monitoring techniques.*

Source: RFC 6980

Preventing the Circumvention of RA-Guard

- Nodes MUST silently ignore the following Neighbor Discovery and SEcure Neighbor Discovery messages if the packets carrying them include an IPv6 Fragmentation Header:
  - Neighbor Solicitation
  - Neighbor Advertisement
  - Router Solicitation
  - Router Advertisement
  - Redirect
  - Certification Path Solicitation

Source: RFC 6980

What IPv6 Capabilities were tested (in General)

- RFC Compliance.
  - Note: There are many cases when non-conforming behaviour is better from a security perspective.
- Fragmentation.
- IPv6 Extension Headers (including deprecated ones).
- Other security features (RA Guard, IDS capabilities), when and where supported.

Testing Scenarios

1. Default configuration (and allowing only **ICMPv6 Echo Request** messages).

- Test which IPv6 Extension Headers are allowed and which are not.
- Use arbitrary and mix combinations of the above

2. Allowing all the available IPv6 Extension Headers.

- Repeat the above tests.

3. By adding a “Default Allow” rule but also blocking specific TCP ports before this.

- Such a configuration it shouldn't be used by any means, but still, the blocked TCP ports should be protected by unauthorised access.

Testing the Support of Extension Headers

- IPv6 Fragment Header
  - Simple Fragmentation
  - Atomic Fragments
- Destination Options Header / Hop-by-Hop Extension Header
  - Unknown Options?
- IPv6 Routing Header
  - Type 0
  - Types 2-3
  - Unknown (non-existing) type
- IPv6/IP4 (as an Extension Header) – Tunnelling (more on this, later).
- Mobility Header.
- Fake (non-existing) header (to test RFC 7045).

Checking the Order and the Number of Occurrences

1. Repeat one IPv6 Extension Header Multiple Time

2. Mix Various IPv6 Extension Headers in a non-Recommended Order

3. Combine tests 1 and 2.

4. Increase the IPv6 Header Chain size (by combining methods of tests 1-3) and fragment it so as layer 4 header to appear at fragment 2, 3, 4, etc.

If one or more of the above pass through the device, check whether they can be used for circumventing firewalls/IDS/RA Guard.

Tunnelling IPv4/IPv6 in IPv6

- IPv6/IPv4 traffic tunnelled inside IPv6.
  - Can they filter such a traffic?
  - What if combined with additional IPv6 Extension Headers per IPv6 main Header or mixing / fragmented them (combined with tests 1-4 of previous slide)?

Other Attacks

- Flooding Attacks (combined with any of the above).
  - Can the devices handle them?
- Delay fragment attacks
  - What if fragments stored until all of them received?
  - What if they forwarded before all of them are received? How does filtering take place?

RISC: Firewall Testing

Cisco ASA 5505
running firmware 9.1(4)

Default Configuration: Fragmentation

- Check whether simple **fragmentation** is allowed: **YES**
- Varying the time interval between consecutive fragments:
  - 2 fragments with 5 sec in between, dropped.
  - 2 fragments with 3 sec in between, passed through.
  - 3 fragments with 2 sec in between, passed through.
  - 3 fragments with 3 sec in between, dropped.
- If all the **fragments** are **stored** before the last one is received: **YES**

CISCO ASA

Default Configuration: IPv6 Extension Headers Support

- Hop-by-Hop Options Header: **YES**
- IPv4 Header: NO
- IPv6 Header: NO
- Type 0/2/3/10 Routing Header: NO
- Fragment Extension Header (Atomic Fragment): **YES**
- Destination Options Header: **YES**
- Mobility Header: NO
- Fake Header: NO

CISCO ASA

Default Configuration: Additional Tests

- Sending the layer-4 protocol header at the 2nd, 3rd, 4th, etc. Fragment by fragmenting an “Options” Header. Dropped.
- Repeating the supported extension headers, two, three, or more times.
  - Hop-by-Hop is allowed only once and only if it is at the beginning (as it should).
  - Destination Options is allowed up to twice, as it should.
  - Fragment Ext. Header is allowed only once.
- Mixing the order of the supported extension headers: All of them are allowed only in the correct order.

CISCO ASA

Default Allow All Rule (and blocking a specific TCP port)

- All the known Extension Headers are allowed.
- Fake Header is also allowed.
- Type 0 Routing Header is still not allowed (kernel blocked?)
- When the packet is NOT fragmented, using a FAKE header **we can reach a TCP port that is not allowed**!

CISCO ASA

Cisco ASA: Conclusions

- Good IPv6 supported functionality (many known IPv6 Extension Headers, fragmentation, etc.).
  - It allows out-of-the-box only non-“risky” IPv6 Extension headers (not IPv4/IPv6/Routing Header/Unknown headers)..
- Operational issues may arise when fragments are slightly delayed.
- Not a 100% RFC compliant (especially when compliance circumvents security).
- Security-oriented:
  - Type-0 Routing header is blocked, even if everything else is allowed.
  - Layer-4 header in a fragment other than the 1st is not accepted.
  - Extension Headers are accepted only in the correct order and in the correct number of occurrences.
- Can be circumvented only if a Default Allow Rule is used.

Fortinet Fortigate 200B running v5.0, build0252 (GA Patch 5)

Default Configuration: Fragmentation

- Check whether simple **fragmentation** is allowed: **YES**
- Varying the time interval between consecutive fragments:
  - When delay >60 secs, packets are dropped. Sent back *ICMPv6 Time exceeded fragment reassembly time exceeded* message.
  - Tested for 2 fragments, 50 secs in between: Passed through.
  - Tested for 3 fragments, 22 secs in between: Passed through
- If all the **fragments** are **stored** before the last one is received: **YES**

Fortigate Fortinet

Default Configuration: IPv6 Extension Headers Support

- Hop-by-Hop Options Header: **YES**
- IPv4 Header: NO
- IPv6 Header: NO
- Type 0 Routing Header: NO
- Type 2/3/10 Routing Header: **YES**
- Fragment Extension Header (Atomic Fragment): **YES**
- Destination Options Header: **YES**
- Mobility Header: NO
- Fake Header: NO

Fortigate Fortinet

Default Configuration: Additional Tests

- Sending the layer-4 protocol header at the 2nd, 3rd, 4th, etc. Fragment by fragmenting an “Options” Header. They pass through. You can send the layer-4 header in the 32nd fragment.
- Can this be used to circumvent firewall (for instance, use TCP SYN scan against a closed port). **NO**
- For the supported Headers, repeat them two, three, and more times - Mix the order of the supported headers.
  - 0, 2x60, 2x44, 60 worked.
  - 2x60, 8 fragments, also worked.

Fortigate Fortinet

Default Allow All Rule (and blocking a specific TCP port)

- Using a Fake Header, **we can reach a TCP port that is not allowed**, either when the IPv6 datagram is fragmented or not.

Fortigate Fortinet

Flooding Attacks

- Possible, in theory, since:
  - Fragments are stored and not forwarded until all of them are received.
  - Fragments are retained (if not all of them have been received) until 60 seconds.
- We could NOT DoS it, but using a single machine, we increased the CPU load at about 20%-24%.
- We finally DoSed the attacker's machine!

Fortigate Fortinet

Fortigate Fortinet: Conclusions

- Very good IPv6 supported functionality (many known IPv6 Extension Headers, fragmentation, etc.). Supports out-of-the box the RFC 2460 Extension Headers.
- Still not a 100% RFC compliant.
- Type-0 Routing header is dropped by default.
- Delayed Fragments are stored and accepted up to 60 seconds.
  - Could be possibly DoSed.
- It allows Extension Headers no matter what the order or their number of occurrences are even in the default configuration.
- It allows layer-4 protocol header in a fragment other than the 1st (it cannot be circumvented though).
- Can be circumvented only if a Default Allow Rule is used.

Juniper SRX 100H2
running JunOS 12.1X46-DH.2

Default Configuration: Fragmentation

- Check whether simple **fragmentation** is allowed: **YES**
- Varying the time interval between consecutive fragments:
  - 5 fragments, 3 sec delay, one passed, rest dropped
  - 3 fragments, 1 sec delay, just the two passed
  - 2 fragments if 1.3 sec accepted, if 1.5 sec dropped
- If all the **fragments** are **stored** before the last one is received: **NO**

Juniper

Default Configuration: IPv6 Extension Headers Support

- Hop-by-Hop Options Header: **YES**
- IPv4 Header: NO
- IPv6 Header: NO
- Type 0 Routing Header: NO 'Parameter problem, erroneous header field encountered'
- Type 2/3/10 Routing Header: **YES**
- Fragment Extension Header (Atomic Fragment): **YES**
- Destination Options Header: **YES**
- Mobility Header: NO
- Fake Header: NO

Juniper

Default Configuration: Additional Tests

- Sending the layer-4 protocol header at the 2nd, 3rd, 4th, etc. fragment by fragmenting an “Options” Header. **Blocked**.
- For the supported Headers, repeat them two, three, and more times - Mix the order of the supported headers.
  - It strictly respects the number of occurrences.
  - It respects the order of the hop-by-hop header but not the other ones (e.g. Routing header is accepted at the end)

Juniper

Default Allow All Rule (and blocking a specific TCP port)

- Using a Fake Header, **we can reach a TCP port that is not allowed**, only when the IPv6 datagram is fragmented.

Juniper

Juniper: Conclusions

- Good IPv6 supported functionality (many known IPv6 Extension Headers, fragmentation, etc.).
- Not a 100% RFC compliant.
- Supports out-of-the box the RFC 2460 Extension Headers.
- Type-0 Routing header is dropped by default.
- Delayed fragments are not stored, and also accepted only for a few seconds
- It strictly respects the number of occurrences of the Extension Headers, but not the recommended order (except from the Hop-by-Hop).
- Can be circumvented only if a Default Allow Rule is used.

Checkpoint Gaia Release 77.10

(running on commodity hardware)

Default Configuration: Fragmentation

- Check whether simple **fragmentation** is allowed: **YES**
- Varying the time interval between consecutive fragments:
  - Two fragments accepted only when the inter-arrival time is about 0.5 sec – definitely do not pass through for 0.8 sec or more.
  - Five fragments do not pass through when the inter-arrival time is about 0.1 sec
- If all the fragments are stored before the last one is received.
  - Not possible to figure out whether fragments are stored before the last one is received due to the very small inter-arrival time. Probably yes.

Checkpoint

Default Configuration: IPv6 Extension Headers Support

- Hop-by-Hop Options Header: NO
- IPv4 Header: NO
- IPv6 Header: NO
- Type 0 Routing Header: NO 'Parameter problem, erroneous header field encountered'
- Type 2/3/10 Routing Header: NO
- Fragment Extension Header (Atomic Fragment): NO
- Destination Options Header: NO
- Mobility Header: NO
- Fake Header: NO

Checkpoint

Default Allow All Rule (and blocking a specific TCP port)

- ALL the known IPv6 Extension Headers are still dropped.
- However, unknown (non-existing) Extension Headers are allowed to pass through!?
- This is still true no matter how many Fake Headers are added (10 or more) or, if you fragmented them!
- If we send the layer-4 header at a fragment other than the 1st (by adding the Fake Header), the firewall is bypassed (**we can reach the otherwise blocked TCP port**).

Checkpoint

Allowing IPv6 Extension Headers

- We finally found a way to configure the allowance of IPv6 Extension Headers at Checkpoint.
- Not that easy!

Checkpoint

Allowing IPv6 Extension Headers (but No Default Allow Rule)

- Destination Options, Hop-by-Hop, Routing Header, Mobility Header are only allowed!
- Not IPv4, IPv6 or Fragment Extension Headers. Atomic Fragments are not allowed. IPv4 or IPv6 Tunnelling is not allowed either.
- When we use a Dest-Opt Header (or any other allowed header) and move layer 4 at a fragment other than the 1st: **Blocked.**
- If we mix several headers multiple times (for example 3 Hop-by-Hop, then 2 Destination Options, then 2 Hop-by-Hop), the packet passes through.
- Type 0 Routing Header is nevertheless blocked. Types 1 to 10 (non-existing) pass through.
- If you use Type 2 Routing Header, then the packet pass through even to an IPv6 Address that is otherwise explicitly blocked.

Checkpoint: Conclusions

- In it's default configuration, it actually eliminates any IPv6 Extension Headers functionality (except from fragmentation). The most paranoid default IPv6 configuration.
- Not that easy to configure it to allow some IPv6 Extension Headers. When you do, the correct order or the correct number of occurrences are not filtered (traffic passes through).
- Very low level of RFC compliance (by default).
- Very strict (less than 1 sec) accepted inter-arrival delay between fragments. Again, the most “paranoid”.
- Type-0 Routing header is nevertheless blocked.
- Can be circumvented only if a Default Allow Rule is used.

RISC: IDS/IPS Testing

Tipping Point, TOS Package 3.6.1.4036 and digital vaccine 3.2.0.8530

Default Configuration

- It doesn't store the fragments locally.
- It immediately forwards them as long as the layer-4 headers is in the 1st fragment.
- It drops a packet when layer-4 header is in the 2nd fragment without issuing an alert.
- What about if it is in parallel and not inline?

Default Configuration: More Findings

- When 10 or more Extension headers are used, it issues an alert.
- But it does not issue an alert if 9 Hop-by-Hop Ext Headers are used.
- Type 0 Routing Header is detected.
- Tunnelling is NOT detected.

How we can bypass Tipping Point

How we can bypass Tipping Point

- Details will be disclosed after a public patch will be available at www.insinuator.net and www.secfu.net. (sorry about that).
- However, we could make Tipping Point completely blind and fly under its radars no matter what kind of attack we launced :)
- Such “malformed” packets are accepted by Windows 7, Kali, Fedora 20 AND OpenBSD, but not from FreeBSD.

Responsibly Disclosured

- Tipping point was informed in 19th of February. A pcap file and some info were provided.
- The description of the attack (crafted IPv6 fragments) was also sent in 22nd of February.
- Tipping Point reaction:
  - They informed us they have fixed the issue.
  - They are going to release publicly the corresponding patch.

Confirming the Issue with a Layer-7 Attack

- XSS Attack:

"GET /index.php?asd=\"><script>alert(1)</script>"

- XSS is not blocked even in “aggressive” mode.
- It also works even when all HTTP traffic is blocked at the Tipping Point.

Cisco Catalyst 4948E running Cisco IOS Release 15.2(1)E1.

RA-Guard Protection

- Known Issue:
  - RA messages circumvent RA-Guard protection if the RA message is in the 2nd fragment, or later.
- What our tests showed:
  - RA in 2nd fragment, just a DestOpt in the 1st – blocked
  - RA in 2nd fragment (or later), DestOpt with data – passed

Confirmed

Other Ways to Circumvent RA-Guard Protection?

- At least two other ways were found (no fragmentation at this time):
  - Tunnel the traffic (IPv6 in IPv6).
  - Use a Fake (unknown) header:
- However, the target does not easily accept such a packet, unless:
  - IPv6 tunnelling (for some reason) has been implemented, or
  - (more possible) there is a new Extension Header, which is known to the target and not to the Switch (usually OS are updated more frequently and more easily than switches firmware).
- Hence, still an issue but probably not of a high risk.

Conclusions

- ***Cisco ASA*** appears to have the most “secure” out-of-the-box configuration, while preserving a very good IPv6 functionality.
- It is the only one that passes through traffic only when Extension Headers appear in the correct order and in the correct number of occurrences (typically not RFC-2460 compliant behaviour but definitely, more secure).
- Type-0 Routing Header is blocked no matter what other traffic is allowed (protecting you for potential misconfiguration).
- The very small accepted inter-arrival delay, although it may eliminates any security issues, it may create operational ones (at least in extreme cases).

Conclusions

- ***Fortigate Fortinet*** appears to have very good IPv6 functionality, but, potential issues can be that:
  - Delayed Fragments are stored and accepted up to 60 seconds. Could be possibly DoSed?
  - It allows Extension Headers no matter what the order or their number of occurrences are even in the default configuration.
  - It allows layer-4 protocol in a fragment other than the 1st (it cannot be circumvented though).

Conclusions

- ***Juniper*** also appears to have a security-oriented default configuration, quite similar to Cisco one but slightly less strict
- It can also encounter operational issues due to the small accepted inter-arrival delay between fragments.

Conclusions

- ***Checkpoint*** actually eliminates IPv6 functionality.
- Very low level of RFC compliance.
- Difficult to configure the usage of IPv6 Extension Headers.
- Almost “paranoid” accepted inter-arrival time between fragments (less than 1 sec). It may cause operational issues more easily than the others.

Tipping Point

- Yet another IPS that appears to have problems regarding examining not expected IPv6 traffic.
- It can be circumvented quite easily.
- Other ways of circumventing may still exist.

Conclusions

- Cisco Catalyst 4948E RA-Guard Evasion:
  - Known issues
  - Other issues also exist, but more difficult to exploit.

Future Work

- These were just a first bunch of tests / experiments.
- More thorough ones are required to further examine any other potential issues.
- The results of RISC project show that securing IPv6 is not as easy as are used from IPv4.
- Thorough knowledge of the protocol is required even from sys and network admins.
- Not to mention about vendors!!