Zero UDP checksum for IPv6

draft-ietf-6man-udpchecksums-05
draft-ietf-6man-udpzero-07

Gorry Fairhurst
P.F. Chimento
Marshall Eubanks

Outline

- Background
- Node Requirements
- Middlebox Requirements
- Usage Requirements
- Next Steps

Background

- Updates to IPv6 Specification (RFC2460)
- Received comments in IESG review
  - Restructuring documents
  - Making udpzero document aStandards Track Applicability Statement

- We are here because this now guides use of IPv6 UDP tunnels

Middlebox Requirements

1. **Middlebox implementations MUST allow forwarding of IPv6 UDP datagram with both a zero and standard UDP checksum**
2. A middlebox MAY configure a restricted set of specific port ranges that forward UDP datagrams with a zero UDP checksum. The middlebox MAY drop IPv6 datagrams with a zero UDP checksum that are outside a configured range
3. **When a middlebox forwards IPv6 UDP datagram with both a zero and standard UDP checksum, the middlebox MUST NOT maintain separate state for flows depending on the value of their UDP checksum field.** This requirement is necessary to enable a sender that always calculates a checksum to communicate via a middlebox with a remote endpoint that uses a zero UDP checksum

Usage Requirements

1. **UDP Tunnels that enable the use of zero UDP checksum MUST only enable this for a specific port or port-range.** This needs to be enabled at both the ingress and egress endpoints.
2. **An integrity mechanisms is always RECOMMENDED at the tunnel layer to ensure that corruption rates of the inner-most packet are not increased.** A mechanism that isolates the causes of corruption (e.g. identifying mis-delivery, IPv6 header corruption, tunnel header corruption) is expected to also provide additional information about the status of the tunnel (e.g. to suggest a security attack). A UDP Tunnel that encapsulates Internet Protocol (IPv4 or IPv6) packets MAY rely on the inner packet integrity checks, provided that the tunnel will not significantly increase the rate of corruption of the inner IP packet. If a significantly increased corruption rate can occur, then the tunnel MUST provide an additional integrity verification mechanism. Early detection is desirable to avoid wasting unneccessary computation/storage for packets that will subsequently be discarded.
3. A UDP Tunnel that supports use of a zero UDP checksum MUST be designed so that corruption of the tunnel header information does not result in accumulated state for the tunneling protocol.

Usage Requirements

5. A UDP Tunnel that encapsulates non-IP packets MUST have a CRC or other mechanism for checking packet integrity, unless the non-IP packet is specifically designed for transmission over lower layers that do not provide a packet integrity guarantee.
6. **A UDP Tunnel egress endpoint that supports a zero UDP checksum MUST also allow reception using a calculated UDP checksum.** The encapsulating ingress endpoint may choose to compute the UDP checksum, or may calculate this by default. **In either case, the egress endpoint MUST use the reception method specified in RFC2460 when the checksum field is not zero.**
7. **A UDP Tunnel with control feedback SHOULD be robust to changes in the network path.** The set of middleboxes on a path may vary during the life of an association. Endpoints need to discover paths with middleboxes that drop packets with a zero UDP checksum. **Therefore keep-alive messages SHOULD send datagrams with a zero UDP checksum. An endpoint that discovers an appreciable loss rate for keep-alive packets MAY terminate the tunnel.** Section 3.1.3 of RFC 5405 describes requirements for congestion control when using UDP-based tunnels.
8. A UDP Tunnel with control feedback that can fall-back to using UDP with a calculated RC 2460 checksum are expected to be more robust to changes in the network path. **Therefore keep-alive messages SHOULD include both UDP datagrams with a checksum and datagrams with a zero UDP checksum.** This will enable the remote endpoint to distinguish between a path failure and dropping of datagrams with a zero UDP checksum. Note that path validation need only be performed for each pair of tunnel endpoints, not for each tunnel context.

Next Steps

- New drafts to be submitted by end November
- New last calls, including a joint one with TSVWG
- Please review them then!

Spare Slides

Node Requirements

1. **An IPv6 sending node MAY use a calculated RFC 2460 checksum for all datagrams that it sends.** This explicitly permits an interface that supports checksum offloading to insert an updated UDP checksum value in all UDP datagrams that it forwards, however note that sending a calculated checksum requires the receiver to also perform the checksum calculation. Checksum offloading can normally be switched off for a particular interface to ensure that the datagrams are sent with a zero UDP checksum.
2. **IPv6 nodes SHOULD by default NOT allow the zero UDP checksum method for transmission.**
3. **IPv6 nodes MUST provide a way for the application/protocol to indicate the set of ports that will be enabled to send datagrams with a zero UDP checksum.** This may be implemented via a socket API call, or similar mechanism. It may also be implemented by enabling the method for a pre-assigned static port used by a specific tunnel protocol.
4. **IPv6 nodes MUST provide a method to allow an application/protocol to indicate that a particular UDP datagram requires a UDP checksum.** This needs to be allowed by the operating system at any time (e.g. to send keep-alive datagrams), not just when a socket is established.

Node Requirements

5. **The default IPv6 node receiver behaviour MUST discard all IPv6 packets carrying datagrams with a zero UDP checksum.**
6. **IPv6 nodes MUST provide a way for the application/protocol to indicate the set of ports that will be enabled to receive datagrams with a zero UDP checksum.** This may be implemented via a socket API call, or similar mechanism. It may also be implemented by enabling the method for a pre-assigned static port used by a specific tunnel protocol.
7. **RFC 2460 specifies that IPv6 nodes SHOULD log received datagrams with a zero UDP checksum.** This remains the case for any datagram received on a port that does not explicitly enable processing of a zero UDP checksum. A port for which the zero UDP checksum has been enabled MUST NOT log the datagram solely because the checksum value is zero.
8. **IPv6 nodes MAY separately identify received UDP datagrams that are discarded with a zero UDP checksum. It SHOULD NOT add these to the standard log, since the endpoint has not been verified.** This may be used to support other functions (such as a security policy).
9. **IPv6 nodes that receive ICMPv6 messages that refer to packets with a zero UDP checksum MUST provide appropriate checks concerning the consistency of the reported packet to verify that the reported packet actually originated from the node, before acting upon the information (e.g. validating the address and port numbers in the ICMPv6 message body).**