The Generalized TTL Security Mechanism (GTSM)
draft-ietf-rtgwg-rfc3682bis-06.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 1, 2007.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

The use of a packet’s Time to Live (TTL) (IPv4) or Hop Limit (IPv6) to verify whether the packet originated within the same link has been used in many recent protocols. This document generalizes this technique. This document obsoletes RFC 3682.
Table of Contents

1. Introduction .................................................. 3
2. Assumptions Underlying GTSM ................................. 3
   2.1. GTSM Negotiation ........................................ 4
   2.2. Assumptions on Attack Sophistication .................. 4
3. GTSM Procedure ................................................. 5
4. Acknowledgments ................................................ 6
5. Security Considerations ....................................... 6
   5.1. TTL (Hop Limit) Spoofing ................................ 7
   5.2. Tunneled Packets .......................................... 7
      5.2.1. IP in IP ............................................. 8
      5.2.2. IP in MPLS .......................................... 8
   5.3. Multi-Hop Protocol Sessions ............................. 9
6. Applicability Statement ....................................... 10
7. IANA Considerations .......................................... 10
8. Changelog ....................................................... 10
   8.1. Changes between -05 and -06 ............................ 10
9. References ...................................................... 11
   9.1. Normative References .................................... 11
   9.2. Informative References .................................. 11
Appendix A. Multihop GTSM ....................................... 12
Authors’ Addresses ................................................. 12
Intellectual Property and Copyright Statements ............... 13
1. Introduction

The Generalized TTL Security Mechanism (GTSM) is designed to protect a router’s IP based control plane from CPU-utilization based attacks. In particular, while cryptographic techniques can protect the router-based infrastructure (e.g., BGP [RFC4271], [RFC4272]) from a wide variety of attacks, many attacks based on CPU overload can be prevented by the simple mechanism described in this document. Note that the same technique protects against other scarce-resource attacks involving a router’s CPU, such as attacks against processor-line card bandwidth.

GTSM is based on the fact that the vast majority of protocol peerings are established between routers that are adjacent [PEERING]. Thus most protocol peerings are either directly between connected interfaces or at the worst case, are between loopback and loopback, with static routes to loopbacks. Since TTL spoofing is considered nearly impossible, a mechanism based on an expected TTL value can provide a simple and reasonably robust defense from infrastructure attacks based on forged protocol packets from outside the network. Note, however, that GTSM is not a substitute for authentication mechanisms. In particular, it does not secure against insider on-the-wire attacks, such as packet spoofing or replay.

Finally, the GTSM mechanism is equally applicable to both TTL (IPv4) and Hop Limit (IPv6), and from the perspective of GTSM, TTL and Hop Limit have identical semantics. As a result, in the remainder of this document the term "TTL" is used to refer to both TTL or Hop Limit (as appropriate).

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Assumptions Underlying GTSM

GTSM is predicated upon the following assumptions:

1. The vast majority of protocol peerings are between adjacent routers [PEERING].

2. It is common practice for many service providers to ingress filter (deny) packets that have the provider’s loopback addresses as the source IP address.

3. Use of GTSM is OPTIONAL, and can be configured on a per-peer (group) basis.
4. The peer routers both implement GTSM.

5. The router supports a method to use separate resource pools (e.g., queues, processing quotas) for differently classified traffic.

Note that this document does not prescribe further restrictions that a router may apply to the packets not matching the GTSM filtering rules, such as dropping packets that do not match any configured protocol session and rate-limiting the rest. This document also does not suggest the actual means of resource separation, as those are hardware and implementation-specific.

The possibility of DoS attack prevention, however, is based on the assumption that packet classification and separation of their paths is done before they go through a scarce resource in the system. In practice, this means that, the closer GTSM processing is done to the line-rate hardware, the more resistant the system is to the DoS attacks.

2.1. GTSM Negotiation

This document assumes that, when used with existing protocols, GTSM will be manually configured between protocol peers. That is, no automatic GTSM capability negotiation, such as is envisioned by RFC 3392 [RFC3392] is assumed or defined.

If a new protocol is designed with built-in GTSM support, then it is recommended that procedures are always used for sending and validating received protocol packets (GTSM is always on, see for example [RFC2461]). If, however, dynamic negotiation of GTSM support is necessary, protocol messages used for such negotiation MUST be authenticated using other security mechanisms to prevent DoS attacks.

Also note that this specification does not offer a generic GTSM capability negotiation mechanism, so messages of the protocol augmented with the GTSM behavior will need to be used if dynamic negotiation is deemed necessary.

2.2. Assumptions on Attack Sophistication

Throughout this document, we assume that potential attackers have evolved in both sophistication and access to the point that they can send control traffic to a protocol session, and that this traffic appears to be valid control traffic (i.e., has the source/destination of configured peer routers).

We also assume that each router in the path between the attacker and...
the victim protocol speaker decrements TTL properly (clearly, if either the path or the adjacent peer is compromised, then there are worse problems to worry about).

Since the vast majority of peerings are between adjacent routers, we can set the TTL on the protocol packets to 255 (the maximum possible for IP) and then reject any protocol packets that come in from configured peers which do NOT have an inbound TTL of 255.

GTSM can be disabled for applications such as route-servers and other large diameter multi-hop peerings. In the event that an the attack comes in from a compromised multi-hop peering, that peering can be shut down (a method to reduce exposure to multi-hop attacks is outlined below).

3. GTSM Procedure

If GTSM is not built into the protocol and used as an additional feature (e.g., for BGP, LDP, or MSDP), it SHOULD NOT be enabled by default.

If GTSM is enabled for a protocol session, the following steps are added to the IP packet sending and reception procedures:

Sending protocol packets:

The TTL field in all IP packets used for transmission of messages associated with GTSM-enabled protocol sessions MUST be set to 255. This also related error handling messages such as TCP RSTs or ICMP errors.

On some architectures, the TTL of control plane originated traffic is under some configurations decremented in the forwarding plane. The TTL of GTSM-enabled sessions MUST NOT be decremented.

Receiving protocol packets:

The GTSM packet identification step associates each received packet addressed to the router’s control plane with one of the following three trustworthiness categories:

+ Unknown: these are packets that cannot be associated with any registered GTSM-enabled session, and hence GTSM cannot make any judgement on the level of risk associated with them.
+ Trusted: these are packets that have been identified as belonging to one of the GTSM-enabled sessions, and their TTL values are within the expected range.

+ Dangerous: these are packets that have been identified as belonging to one of the GTSM-enabled sessions, but their TTL values are NOT within the expected range, and hence GTSM believes there is a risk that the packets have been spoofed.

The exact policies applied to packets of different classifications are not postulated in this document and are expected to be configurable. Configurability is likely necessary particular with the treatment of related messages such as ICMP errors and TCP RSTs. It should be noted that fragmentation may restrict the amount of information available to the classification.

However, by default, the implementations:

+ SHOULD ensure that packets classified as Dangerous do not compete for resources with packets classified as Trusted or Unknown.

+ MUST NOT drop (as part of GTSM processing) packets classified as Trusted or Unknown.

+ MAY drop packets classified as Dangerous.

4. Acknowledgments

The use of the TTL field to protect BGP originated with many different people, including Paul Traina and Jon Stewart. Ryan McDowell also suggested a similar idea. Steve Bellovin, Jay Borkenhagen, Randy Bush, Alfred Hoenes, Vern Paxson, Robert Raszuk and Alex Zinin also provided useful feedback on earlier versions of this document. David Ward provided insight on the generalization of the original BGP-specific idea. Alex Zinin and Alia Atlas provided significant amount of feedback for the newer versions of the document.

5. Security Considerations

GTSM is a simple procedure that protects single hop protocol sessions, except in those cases in which the peer has been compromised. In particular, it does not protect against the wide range of on-the-wire attacks; protection from these attacks requires
more rigorous security mechanisms.

5.1. TTL (Hop Limit) Spoofing

The approach described here is based on the observation that a TTL (or Hop Limit) value of 255 is non-trivial to spoof, since as the packet passes through routers towards the destination, the TTL is decremented by one. As a result, when a router receives a packet, it may not be able to determine if the packet’s IP address is valid, but it can determine how many router hops away it is (again, assuming none of the routers in the path are compromised in such a way that they would reset the packet’s TTL).

Note, however, that while engineering a packet’s TTL such that it has a particular value when sourced from an arbitrary location is difficult (but not impossible), engineering a TTL value of 255 from non-directly connected locations is not possible (again, assuming none of the directly connected neighbors are compromised, the packet hasn’t been tunneled to the decapsulator, and the intervening routers are operating in accordance with RFC 791 ([RFC0791]).

5.2. Tunneled Packets

The security of any tunneling technique depends heavily on authentication at the tunnel endpoints, as well as how the tunneled packets are protected in flight. Such mechanisms are, however, beyond the scope of this memo.

An exception to the observation that a packet with TTL of 255 is difficult to spoof may occur when a protocol packet is tunneled and the tunnel is not integrity-protected (i.e., the lower layer is compromised).

When the protocol packet is tunneled directly to the protocol peer (the protocol peer is the decapsulator), the GTSM provides no added protection as the security depends entirely on the integrity of the tunnel.

When the protocol packet is tunneled to the penultimate hop which then forwards the packet to a directly connected protocol peer, TTL is decremented as described below except in some myriad Bump-in-the-Wire (BITW) cases [BITW].

In IP-in-MPLS cases described below, the TTL is always decremented by at least one.
5.2.1. IP in IP

Protocol packets may be tunneled over IP directly to a protocol peer, or to a decapsulator (tunnel endpoint) that then forwards the packet to a directly connected protocol peer (e.g., in IP-in-IP [RFC2003], GRE [RFC2784], or various forms of IPv6-in-IPv4 [RFC4213]). These cases are depicted below.

Peer router ---------- Tunnel endpoint router and peer
TTL=255 [tunnel] [TTL=255 at ingress]
                [TTL=255 at egress]

Peer router ---------- Tunnel endpoint router ----- On-link peer
TTL=255 [tunnel] [TTL=255 at ingress] [TTL=254 at ingress]
                [TTL=254 at egress]

In the first case, in which the encapsulated packet is tunneled directly to the protocol peer, the encapsulated packet’s TTL can be set arbitrary value. In the second case, in which the encapsulated packet is tunneled to a decapsulator (tunnel endpoint) which then forwards it to a directly connected protocol peer, RFC 2003 specifies the following behavior:

When encapsulating a datagram, the TTL in the inner IP header is decremented by one if the tunneling is being done as part of forwarding the datagram; otherwise, the inner header TTL is not changed during encapsulation. If the resulting TTL in the inner IP header is 0, the datagram is discarded and an ICMP Time Exceeded message SHOULD be returned to the sender. An encapsulator MUST NOT encapsulate a datagram with TTL = 0.

Hence the inner IP packet header’s TTL, as seen by the decapsulator, can be set to an arbitrary value (in particular, 255). As a result, it may not be possible to deliver the protocol packet to the peer with a TTL of 255.

5.2.2. IP in MPLS

Protocol packets may also be tunneled over MPLS to a protocol peer which either the penultimate hop (when the penultimate hop popping (PHP) is employed [RFC3032]), or one hop beyond the penultimate hop. These cases are depicted below.
TTL handling for these cases is described in RFC 3032. RFC 3032 states that when the IP packet is first labeled:

... the TTL field of the label stack entry MUST BE set to the value of the IP TTL field. (If the IP TTL field needs to be decremented, as part of the IP processing, it is assumed that this has already been done.)

When the label is popped:

When a label is popped, and the resulting label stack is empty, then the value of the IP TTL field SHOULD BE replaced with the outgoing TTL value, as defined above. In IPv4 this also requires modification of the IP header checksum.

where the definition of "outgoing TTL" is:

The "incoming TTL" of a labeled packet is defined to be the value of the TTL field of the top label stack entry when the packet is received.

The "outgoing TTL" of a labeled packet is defined to be the larger of:

a) one less than the incoming TTL,
b) zero.

In either of these cases, the minimum value by which the TTL could be decremented would be one (the network operator prefers to hide its infrastructure by decrementing the TTL by the minimum number of LSP hops, one, rather than decrementing the TTL as it traverses its MPLS domain). As a result, the maximum TTL value at egress from the MPLS cloud is 254 (255-1), and as a result the check described in section 3 will fail.

5.3. Multi-Hop Protocol Sessions

While GTSM could possibly offer a slightly more limited security properties also when used with multi-hop protocol sessions (see Appendix A), we do not specify GTSM for multi-hop scenarios due to
simplicity, lack of deployment and implementation.

6. Applicability Statement

GTSM is only applicable to environments with inherently limited topologies (and is most effective in those cases where protocol peers are directly connected). In particular, its application should be limited to those cases in which protocol peers are directly connected.

Experimentation on GTSM’s applicability and security properties is needed in multi-hop scenarios. The multi-hop scenarios where GTSM might be applicable is expected to have the following characteristics: the topology between peers is fairly static and well known, and in which the intervening network (between the peers) is trusted.

7. IANA Considerations

This document requires no action from IANA.

8. Changelog

NOTE to the RFC-editor: please remove this section before publication.

8.1. Changes between -05 and -06

- Clarify the assumptions wrt. resource separation and protection based on comments from Alex Zinin.
- Rewrite the GTSM procedure based on text from Alex Zinin.
- Reduce TrustRadius and multi-hop scenarios to a mention in an Appendix.
- Describe TCP-RST, ICMP error and "related messages" handling.
- Update the tunneling security considerations text.
- Editorial updates (e.g., shortening the abstract).

9. References
9.1. Normative References


9.2. Informative References


Appendix A.  Multihop GTSM

NOTE: This is a non-normative part of the specification.

The main applicability of GTSM is for directly connected peers. GTSM could be used for non-directly connected sessions as well, where the recipient would check that the TTL is within "TrustRadius" (e.g., 1) of 255 instead of 255. As such deployment is expected to have a more limited applicability and different security implications, it is not specified in this document.

Authors’ Addresses

Vijay Gill
Email: vijay@umbc.edu

John Heasley
Email: heas@shrubbery.net

David Meyer
Email: dmm@1-4-5.net

Pekka Savola
Espoo
Finland
Email: psavola@funet.fi
Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).